Principles for Systematic Development of an Assurance Case Template from ISO 26262

Thomas Chowdhury*, Chung-Wei Lin†, BaekGyu Kim†, Mark Lawford*, Shinichi Shiraishi†, and Alan Wassyng*

*McMaster Centre for Software Certification, McMaster University, Hamilton, ON, Canada
†Systems & Software Division, Toyota InfoTechnology Center, Mountain View, CA, USA
Emails: {chowdt2, lawford, wassyng}@mcmaster.ca, {cwlin, bkim, sshiraishi}@us.toyota-itc.com

***Abstract*—A failure in a critical system can cause death, injury, financial loss, and environmental damage. To develop safe and trustworthy systems, we need to plan the development and assessment of system functionality in advance. Assurance Cases are a generalization of Safety Cases, and are gaining momentum as a preferred way of demonstrating assurance of critical properties in complex software-intensive systems. To cope with the lack of standardized assurance structures, and to encourage safety assessment prior to development, we previously proposed the use of an assurance case template. The principles presented here can be used to build an assurance case template that complies with the functional safety standard, *ISO 26262* in a cost-effective way. In the future, such principles may lead to semi-automated development of these templates.**

1. Introduction

An assurance case is a popular method we can use to document system safety assurance [1]. According to Bloomfield *et al.*, *“An assurance case is a documented body of evidence that provides a convincing and valid argument that a specified set of critical claims about a system’s properties are adequately justified for a given application in a given environment”* [2]. The assurance case starts with a claim about the properties of the system of interest that is supported by a structure of sub-claims, eventually supported by evidence. This structure is easier to understand in a graphical format, and a popular notation is Goal Structuring Notation (GSN), developed by Kelly [3].

An assurance case template is a complete assurance case for a product-line, developed prior to building products of that product-line. An essential aspect of system safety assurance for safety critical systems is that it is necessary to plan and document, as early as possible, how and why the system will be developed and assessed. Such a template supports this approach. The template includes optional argument paths dependent on the specific product, kinds of required evidence, and acceptance criteria for the evidence [4]. A skeleton of an assurance case template is shown in Figure 1. The arrows in this diagram and other assurance case extracts in this paper are drawn from a parent to a sub-claim, which is popularized by GSN.

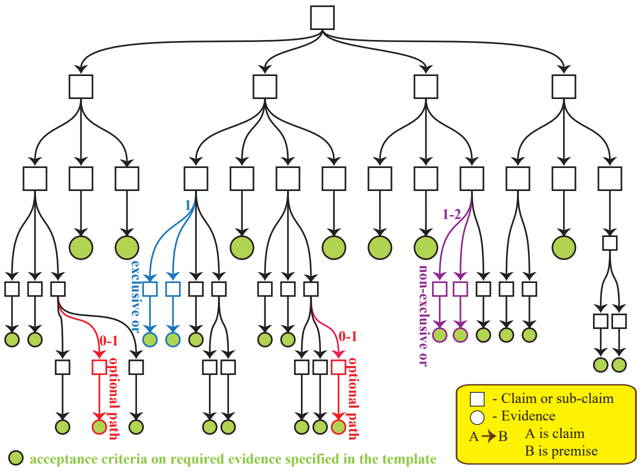
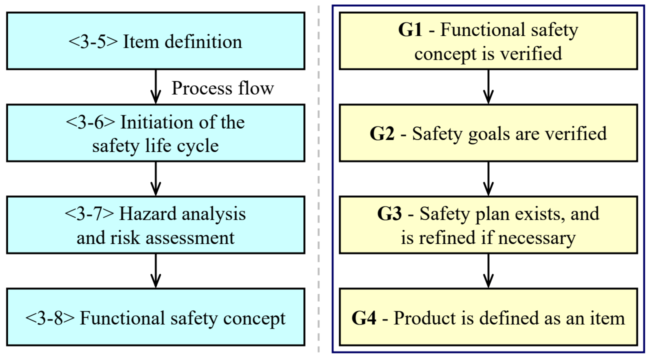


Figure 1. Assurance case template (modified from [4]).

The main contribution of this paper is to define principles we can use to develop an assurance case template that complies with a standard such as *ISO 26262 (Road vehicles—Functional safety* [5]). *ISO 26262* is the de facto standard for functional safety of automotive vehicles. It deals with the electrical and electronic components of automotive vehicles—including software. We intend that the contribution of this paper will eventually pave the way to semi-automated development of such templates.

2. Literature Review

There have been a number of works related to building assurance cases compliant with existing standards, or using assurance cases to represent implicit safety arguments in a standard. The first such attempt was made by Ankrum and Kromholz who targeted three standards in 2005 [6]. In recent years, Holloway [7] described initial attempts at classifying *DO-178C* (the de facto civil aviation standard)



<3-5> Item definition
Process flow
<3-6> Initiation of the safety life cycle
<3-7> Hazard analysis and risk assessment
<3-8> Functional safety concept
G1 - Functional safety concept is verified
G2 - Safety goals are verified
G3 - Safety plan exists, and is refined if necessary
G4 - Product is defined as an item

(a) Process steps in *ISO 26262* (b) Related claims

Figure 2. Illustration of the Flip-It principle.

and developed the start of a primary argument and the start of a confidence argument that illustrates the implicit arguments in the standard. Birch *et al.* [8] used assurance case fragments to show how to make a more product-focused safety case, still in compliance with the requirements of *ISO 26262*. Hocking *et al.* [9] constructed the overview of an assurance case that complies with *ISO 26262*, but it is not developed by converting the standard into an assurance case. Holloway [10] revisited the earlier work and revised the approach. This resulted in the development of *"four fundamental concepts"*: (1) where "correct" software does not contribute to an unsafe state, (2) permitting development flexibility, (3) use of confidence arguments to support the primary argument, and (4) describing the (perceived) argument in the standard before trying to evaluate the sufficiency of the case.

3. Building an Assurance Case Template

Similar to Holloway's quest for classification and fundamental concepts [7], [10], we have developed ten principles that we can use, and reuse, in development of an assurance case template, that complies with a standard such as *ISO 26262*. Due to the limitation of space, we select and present eight of them in this paper.

3.1. Principles for Constructing a Template

Principle 1: Modeling Standard: Most modern standards are complex enough that simply reading them (many times) is insufficient to develop a deep enough understanding to build assurance case templates. We use a number of models to aid us in our understanding of *ISO 26262*. These models include a *data flow* of processes and work products, a *conceptual model* of the standard, and a list of *consolidated work products*, showing what clauses contribute to the work product during development. The conceptual model will be useful in automating aspects of the template (such as checks on completeness, etc.) as well as understanding the standard. The consolidated work products are important, because we know that an assurance case is static, in that the argument must not be time dependent. Contents of work products will be used as evidence to support one or more claims.

Principle 2: Modeling System Variability: The assurance case template is designed for assuring products within a product family. It is thus important that we document variations in the product family. For example, in order to develop an effective assurance case template for Adaptive Cruise Control (ACC), it is important that we take into account all known variations of ACC that we are likely to build. As a start, we need to develop and document a *feature model* for the product family. After that, it really depends on whether this is the first time that the product is to be built, or whether the manufacturer has experience of building such products in the past.

Principle 3: Flip-It: Standards such as *ISO 26262* are predominantly process based, and the clauses in the standard usually dictate a process to be followed. An assurance case documents an argument, described through the use of claims, sub-claims, and evidence. Very often, a sequence of process steps in the standard will translate (directly) into an argument fragment, consisting of a path of claims and sub-claims. Taking Part 3 in *ISO 26262* as an exmaple, we find the clauses shown in Figure 2(a), where <p-s> indicates Part p, Section s in *ISO 26262*. Each one of these process steps can be transformed into an associated claim in an argument fragment. Specifically,

- Item definition → "Product is defined as an item", in compliance with <3-5>.
- Initiation of the safety life cycle → "Safety plan exists, and is refined if necessary" (item is not new), in compliance with <3-6>.
- Hazard analysis & risk assessment → "Safety goals are verified", in compliance with HARA (hazard and risk analysis) <3-7>.
- Functional safety concept → "Functional safety concept is verified", in compliance with <3-8>.

Now, if we examine these claims, we find that "Functional safety concept is verified" depends solely on the claim "Safety goals are verified". In other words, for process related claims that are directly dependent, we "flip" the order of the claims with respect to the process steps. The resulting claims for the process steps in Figure 2(a) are shown in Figure 2(b).

Principle 4: Conjunctive: In the above discussion of the Flip-It principle, we assert that each claim depends solely on the claim that supports it. If a claim depends on more than one other claim, then we apply the Conjunctive principle. For example, consider the determination of Automotive Safety Integrity Levels (ASILs) as described in *ISO 26262*. The relevant process clauses are shown in Figure 3(a). We find that the determination of the ASIL is dependent on all three steps: estimations of severity of potential harm, probability of exposure, and controllability of hazardous event. In this case, any claim we make about determining the ASIL, must be supported by three claims concerning severity, exposure, and controllability. Thus, we

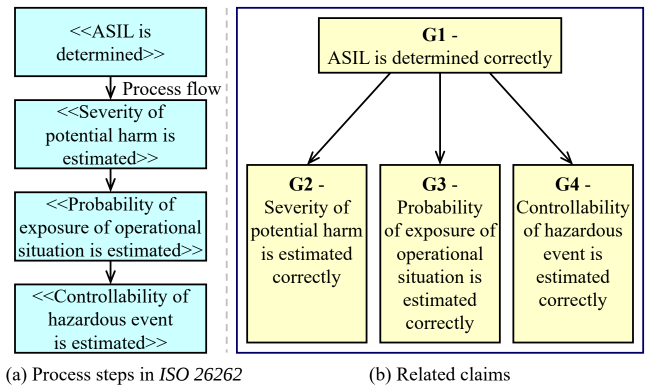

<<ASIL is determined>>
Process flow
<<Severity of potential harm is estimated>>
<<Probability of exposure of operational situation is estimated>>
<<Controllability of hazardous event is estimated>>
(a) Process steps in ISO 26262
G1 - ASIL is determined correctly
G2 - Severity of potential harm is estimated correctly
G3 - Probability of exposure of operational situation is estimated correctly
G4 - Controllability of hazardous event is estimated correctly
(b) Related claims

Figure 3. Illustration of the Conjunctive principle.

arrive at the argument fragment shown in Figure 3(b). The conjunctive claims together with their parent claim, form a unit in the argument structure. This entire unit may be "flipped" when applying the Flip-It principle. However, when the Flip-It principle is applied to a conjunctive claim, the order of claims within the conjunctive unit is not changed. Therefore, in Figure 3 for example, no matter how this conjunctive claim is used within an assurance case, G2, G3, and G4 will always be sub-claims of G1.

Principle 5: Optional Pattern: The standard may define alternatives depending on what happens in practice. For example, a component may be developed by the manufacturer, or it may be outsourced or bought off-the-shelf. In the case in which it is outsourced or bought off-the-shelf, we treat it as being developed by a supplier. As a result, there will be different argument paths dependent on whether a component is developed by the manufacturer or by a supplier. Part 8 in *ISO 26262*, for example, details how safety compliance is attained when a component is obtained from a supplier. In addition, a claim that the implementation complies with its requirements may be supported by mathematical analysis to this effect, and/or by testing. There is another example of an optional argument path, and it is shown in Figure 4. Testing will always be used, but mathematical analysis may not be. In general, we may need optional paths, exclusive-or paths, or non-exclusive-or paths. Similarly, since we are developing a template for a product family, different features in the family, and even different sensors, for example, will require different claims and evidence.

Principle 6: Evidence Specification: *ISO 26262* specifies desired attributes and characteristics for various artifacts. For example, Part 8, Clause 6 specifies attributes and characteristics for safety requirements. The safety requirements, themselves, will form part of the evidence for one or more claims. The attributes and characteristics specified in *ISO 26262* can then be used as acceptance criteria for the safety requirements in the appropriate evidence node. Taking an ACC as an example, there is a claim in the assurance case template that "Functional safety requirements of ACC are derived from correct safety goals with assigned ASIL,

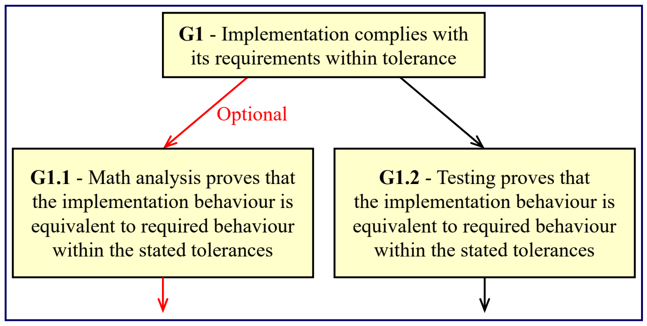

G1 - Implementation complies with its requirements within tolerance
Optional
G1.1 - Math analysis proves that the implementation behaviour is equivalent to required behaviour within the stated tolerances
G1.2 - Testing proves that the implementation behaviour is equivalent to required behaviour within the stated tolerances

Figure 4. Example of optional argument path.

allocated to preliminary architectural elements of ACC and verified". This is supported by an argument path consisting of sub-claims and appropriate evidence, where an evidence node related to the above claim specifies: (1) the kind of required evidence—the work product <4-6.5.1> and (2) the acceptance criteria for the evidence—the attributes and characteristics of safety requirements, specified in <8-6.4.2>.

Principle 7: Evidence Classification: We identify evidence associated with four types of claims—evidence for claims related to planning, process, verification, and expertise. The first three types are identified in *ISO 26262*. The final type is one that we observe often in assurance cases. The classification helps us to define the type of required evidence and the acceptance criteria for the evidence and organize our knowledge so that we can re-use (or slightly modify) appropriate acceptance criteria.

Principle 8: Completeness Arguments: One of the most difficult arguments we have to contend with in assurance, is one that depends on completeness. For example, the claim that "all hazards are mitigated" is important in arguing safety, since it is useful only if all hazards are identified. In such a case, we insist on adding a claim that explains why best effort was expended in determining that "no additional hazards were identified". The sub-claims that support such a claim must include: claims as to why the process was thorough enough to have discovered additional hazards, claims that the results of these investigations show conclusively that no additional hazards are likely to exist, and claims regarding the expertise of the people who conducted this investigation. This principle does not apply to a large percentage of the standard/assurance case, but its effect is large, indeed.

3.2. Principle Coverage in *ISO 26262*

The remaining principles are the Argument Options and Feature Options principles. Table 1 shows the percentage of *ISO 26262* covered by each principle. Our calculations are based on 349 core requirements (Parts 2–7), 519 total requirements(Parts 2–9, including core requirements), 118 work products, and 23 HARA requirements (Part 3 only). We calculated the covered percentage of each principle based on total requirements except principle 3. For example,

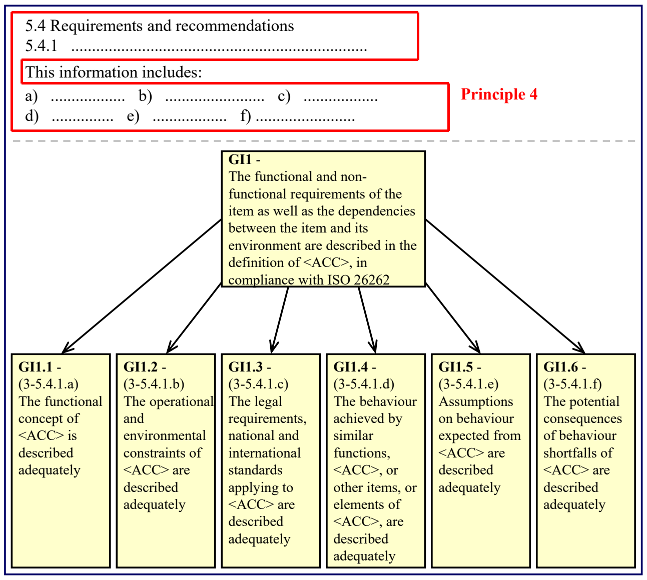

5.4 Requirements and recommendations
5.4.1 ......
This information includes:
a) ...... b) ...... c) ......
d) ...... e) ...... f) ......
Principle 4
GI1 - The functional and non-functional requirements of the item as well as the dependencies between the item and its environment are described in the definition of <ACC>, in compliance with ISO 26262
GI1.1 - (3-5.4.1.a) The functional concept of <ACC> is described adequately
GI1.2 - (3-5.4.1.b) The operational and environmental constraints of <ACC> are described adequately
GI1.3 - (3-5.4.1.c) The legal requirements, national and international standards applying to <ACC> are described adequately
GI1.4 - (3-5.4.1.d) The behaviour achieved by similar functions, <ACC>, or other items, or elements of <ACC>, are described adequately
GI1.5 - (3-5.4.1.e) Assumptions on behaviour expected from <ACC> are described adequately
GI1.6 - (3-5.4.1.f) The potential consequences of behaviour shortfalls of <ACC> are described adequately

Figure 5. Example of developing claims using the principles.

principle 4 is applicable to 192 out of 519 total requirements. So, the covered percentage of principle 4 is 37%. Principle 3 is applicable to 218 out of 349 core requirements. So the covered percentage of principle 3 is 63%. Note that we have not yet come across requirements in the standard that we cannot include in the assurance case template.

TABLE 1. COVERED PERCENTAGES OF *ISO 26262* CLAUSES.

Principle	Target	%
1	Total Requirements	100
2	Total Requirements	100
3	Core Requirements	63
4	Total Requirements	37
5	Total Requirements	9
6	Total Requirements	5
7	Work Products	75
8	HARA	1
9	Total Requirements	7
10	Total Requirements	7

3.3. Examples of Using the Principles

This section shows how we use the principles to develop an assurance case template based on *ISO 26262*. Principle 1 is obvious and will not be demonstrated in this section. We start by annotating major sections of the standard with the relevant principles for each section. We can take Part 3, Section 5.4.1 as an example. The top part of Figure 5 shows the specific sections on the relevant page of the standard with our annotations. The resulting extract from the assurance case template is shown in the bottom part of Figure 5. Another example is the annotation of Parts 6 and 8 shown in the top part of Figure 6. This extract describes characteristics and attributes of the safety requirements, so we annotate the extract to show that we will use Principle 6.

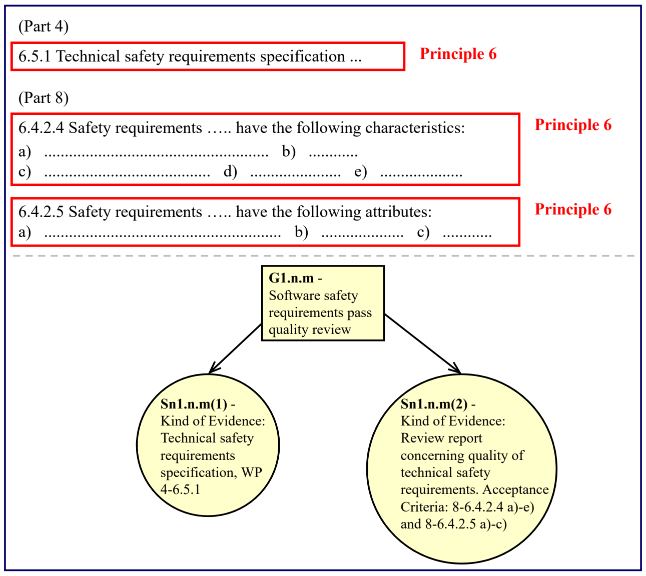

(Part 4)
6.5.1 Technical safety requirements specification ...
Principle 6
(Part 8)
6.4.2.4 Safety requirements ..... have the following characteristics:
a) ...... b) ......
c) ...... d) ...... e) ......
Principle 6
6.4.2.5 Safety requirements ..... have the following attributes:
a) ...... b) ...... c) ......
Principle 6
G1.n.m - Software safety requirements pass quality review
Sn1.n.m(1) - Kind of Evidence: Technical safety requirements specification, WP 4-6.5.1
Sn1.n.m(2) - Kind of Evidence: Review report concerning quality of technical safety requirements. Acceptance Criteria: 8-6.4.2.4 a)-e) and 8-6.4.2.5 a)-c)

Figure 6. Example of specifying evidence using Principle 6.

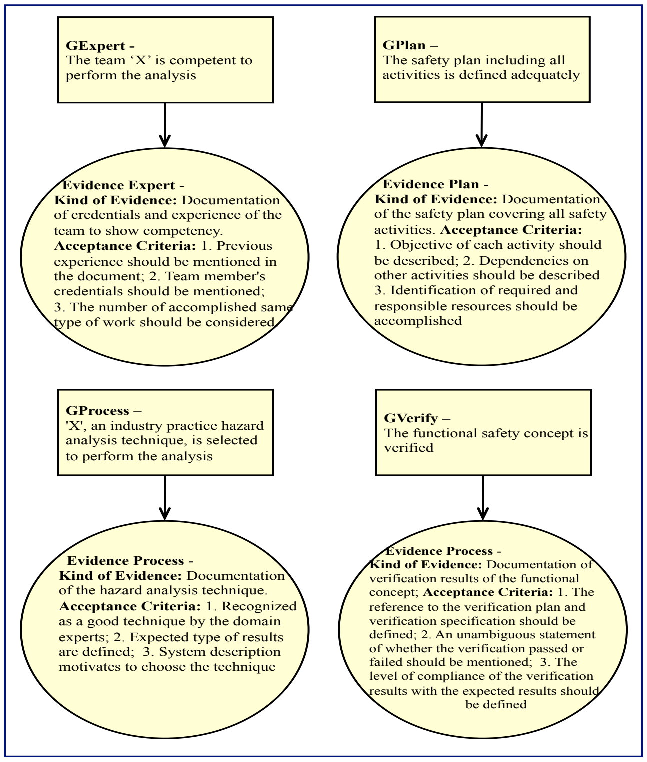

GExpert - The team 'X' is competent to perform the analysis
Evidence Expert - Kind of Evidence: Documentation of credentials and experience of the team to show competency. Acceptance Criteria: 1. Previous experience should be mentioned in the document; 2. Team member's credentials should be mentioned; 3. The number of accomplished same type of work should be considered
GPlan – The safety plan including all activities is defined adequately
Evidence Plan - Kind of Evidence: Documentation of the safety plan covering all safety activities. Acceptance Criteria: 1. Objective of each activity should be described; 2. Dependencies on other activities should be described 3. Identification of required and responsible resources should be accomplished
GProcess – 'X', an industry practice hazard analysis technique, is selected to perform the analysis
Evidence Process - Kind of Evidence: Documentation of the hazard analysis technique. Acceptance Criteria: 1. Recognized as a good technique by the domain experts; 2. Expected type of results are defined; 3. System description motivates to choose the technique
GVerify – The functional safety concept is verified
Evidence Process - Kind of Evidence: Documentation of verification results of the functional concept; Acceptance Criteria: 1. The reference to the verification plan and verification specification should be defined; 2. An unambiguous statement of whether the verification passed or failed should be mentioned; 3. The level of compliance of the verification results with the expected results should be defined

Figure 7. Example of specifying evidence using Principle 7.

The bottom part of Figure 6 shows the acceptance criteria in the evidence nodes, directly extracted from *ISO 26262*. Figure 7 represents an example of principle 7. Evidence of four types of claims define recommended documentation with appropriate acceptance criteria, so they can be reused later.

4. Conclusion

We have motivated the use of assurance case templates as a means of capturing the requirements of a complex standard such as *ISO 26262*. To aid this development, we presented eight principles that can be used to drive the transformation of clauses in the standard into claims and evidence in the assurance case template. We estimated the *coverage* of clauses in *ISO 26262* by each principle and illustrated how the principles may be applied through the use of simple examples taken directly from *ISO 26262*.

References

[1] J. Rushby, X. Xu, M. Rangarajan, and T. L. Weaver, "Understanding and evaluating assurance cases," SRI International, Tech. Rep., 2015.

[2] R. Bloomfield, P. Bishop, C. Jones, and P. Froome, "ASCAD, Adelard Safety Case Development Manual," *Adelard, 1998. ISBN 0-9533771-0*, vol. 5, 1998.

[3] T. P. Kelly, "Arguing Safety—A Systematic Approach to Safety Case Management," *Department of Computer Science, The University of York*, 1998.

[4] A. Wassyng, P. Joannou, M. Lawford, T. S. Maibaum, and N. K. Singh, "Chapter 13 New Standards for Trustworthy Cyber-Physical Systems," in *Trustworthy Cyber-Physical Systems Engineering*. CRC Press, 2016, pp. 337–368.

[5] ISO, "26262: Road vehicles—Functional safety," *International Standard ISO/FDIS*, vol. 26262, 2011.

[6] T. Ankrum and A. Kromholz, "Structured assurance cases: three common standards," in *IEEE International Symposium on High-Assurance Systems Engineering (HASE)*, 2005, pp. 99–108.

[7] C. M. Holloway, "Making the implicit explicit: Towards an assurance case for DO-178C," in *International System Safety Conference*, 2013.

[8] J. Birch, R. Rivett, I. Habli, B. Bradshaw, J. Botham, D. Higham, P. Jesty, H. Monkhouse, and R. Palin, "Safety cases and their role in ISO 26262 functional safety assessment," in *International Conference on Computer Safety, Reliability, and Security*. Springer, 2013, pp. 154–165.

[9] A. B. Hocking, J. Knight, M. A. Aiello, and S. Shiraishi, "Arguing software compliance with ISO 26262," in *IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW)*. IEEE, 2014, pp. 226–231.

[10] C. M. Holloway, "Explicate'78: Uncovering the Implicit Assurance Case in DO-178C," in *Safety-Critical Systems Symposium*, 2015.