Application of a Commercial Assurance Case Tool to Support Software Certification Services.

Luke Emmet <loe@adelard.com>
Sofia Guerra <aslg@adelard.com>

Adelard, Drysdale Building,
Northampton Square, London EC1V 0HB, UK

1 Introduction

Many industry sectors require a documented case that the system will meet its critical requirements; this documented case is often called an "assurance case". In the past, safety justifications tended to be *implicit* and *standards-based*—compliance to accepted practice was deemed to imply adequate safety. This approach works well in stable environments where best practice is supported by extensive experience, but in those sectors adopting fast moving technologies, a more explicit goal-based approach has been advocated, which can accommodate change and alternative strategies to achieve the same goal.

Goal-based approaches have been increasingly adopted in a range of industry sectors, where claims (or goals) are made about the system and arguments and evidence is presented to support that those goals are met. Goal-based approaches are more flexible as they focus directly on the critical requirements and they are more attuned to the ways in which sophisticated engineering arguments are actually made.

Our contention is that developing a software certificate is akin to the process used for system assurance argument. In fact, if software certification is taken as the "demonstration of the reliability, safety, or security of software systems in such a way that it can be checked by an independent authority" [7], there is no obvious clear distinction between software certification and the development of a safety case in a regulated sector (although for non-regulated sectors, there might not be the review by an independent authority).

Based on conceptual work by Toulmin [1], recent assurance and safety case notations have been developed, including Claims-Argument-Evidence [2], [3] and Goal Structuring Notation [6].

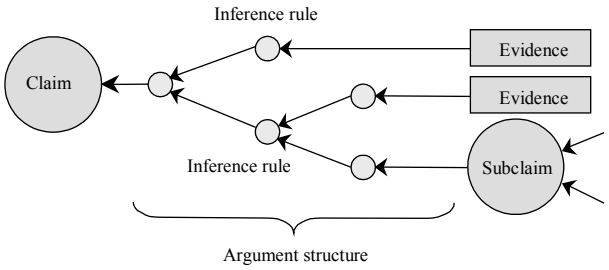


Figure 1: Generic argument structure

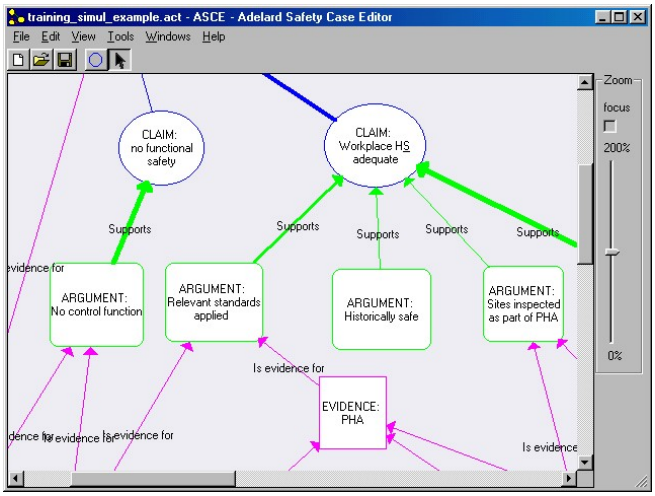
The components of the assurance cases are similar to those of a software certification process. They are:

- *Claims* about a property of the system or sub-system, such as functional properties, RAM (Reliability, Availability, Maintainability) properties, security or usability.
- *Argumentation* nodes that describe the approach to satisfying the claim and provide the rationale for how the collected evidence is to be used to support the claims being made about it. We distinguish between deterministic or analytical (e.g. formal proof, exhaustive testing), probabilistic (e.g. MTTF, reliability testing) and qualitative (e.g. compliance with QMS or with safety standards).

- Process and product *evidence* for the software components, including design documentation quality system documentation, testing evidence, COTS product evaluations, etc.

In Adelard we have been addressing the justification of software-based systems for 18 years. This work has included research and development into the nature, structure and content of the justification of software. The concepts we have developed have been especially used for justifying safety of software-based systems, but these same concepts have been used for reasoning about other system properties such as dependability, reliability and security [8].

2 The ASCE system



ASCE (The Assurance and Safety Case Environment) is a flexible graphical hypertext system used for the development, review and maintenance of assurance and safety cases and other structured technical documentation [4], [5]. It is a fully supported commercial[1] tool currently being used by a wide range of organisations for the development, and assurance of safety-related systems.

Its main technical aspects are as follows:

- Flexible and intuitive *graphical editor* supporting a range of notations, such as Claims-Arguments Evidence and GSN. Other notations can be supported through the use of domain-specific "schemas".
- Each node contains a *structured HTML* narrative field supporting embedded fine grained cross-references to other elements in the network, shared files and Internet resources.
- User *extensible plugins* to support specific applications using ASCE, and integration with other technologies, including
    - *Validated content services*—so-called Dynamic Narrative Regions (DNRs). This allows narrative to be generated from external sources and integrated into an ASCE node narrative. ASCE checksums all DNR content and enables such external dependencies to be validated to see if they have changed since last imported.
    - *Structural and content analysis services*. For example algorithms can be developed that identify all the evidence nodes that provide support for a selected claim. An impact analysis algorithm could identify all claims reliant on a particular piece of evidence.
    - *Powerful reporting capabilities*. Custom reports can be defined by using plugins, which augment the standard capabilities of exporting to a collection of HTML files or to a linear word processed document.
- Its *XML data structure* allows external analysis of ASCE files, and supports long-term use of the data within an ASCE network.
- Includes an *ASCE difference tool*, allowing the analyses of two versions of an ASCE file for detecting structural and narrative differences between them. The ASCE difference tool can be used to support review or audit activities.

[1] ASCE is free for non commercial teaching and academic research purposes.

3 Software Certification

The ASCE system as it is currently designed can be readily configured to support software certification processes. Examples of scenarios of use are described in the following subsections.

3.1 Development of software certification case structure

The overall certification argument is defined in terms of claims being made for the system and its component subsystems. ASCE supports the development and clear presentation of the strategy used for justifying the software certification, showing how these claims will be met (and are progressively met as part of the certification process) and the evidence that has been identified to support the claims. The argument presented will ultimately be audited or reviewed (see Section 3.3 below).

3.2 Evidence integration

As the certification process evolves, the argumentation strategies may be developed and the evidence that supports the claims will become available. In some cases, the argumentation strategy may need to be modified if the evidence is stronger or weaker than originally envisaged.

One particular kind of evidence may be a certification case for subsystem elements. These can be integrated as evidence nodes in the overall certification case using DNR elements in the node narratives. This supports the notion of certificate hierarchies.

Overall, the diverse supporting evidence will be integrated using a number of methods:

- Narrative entered directly into the ASCE file.
- Hyperlinks to external files containing the evidence.
- DNR elements for critical evidence (e.g. testing evidence may need to be integrated, showing the actual status of the test results). This provides the strongest level of validation to external information sources.

A particular DNR might be developed that allows the certification case to particularly refer to the full configuration of a software item. This might require a simple integration with the configuration management tool that is being used.

3.3 Auditing and review

The certification case can be audited by reviewing the contents of the ASCE file. Given that external evidence is directly linked in, it is possible for the auditor to graphically view, and ultimately follow chains of dependencies from the claims being made down to the underlying evidence. The ASCE difference tool might be used to review differences since the previous version. The ASCE tool also supports clear notation for making which nodes have been audited and completed or accepted. The auditor can easily keep track of what aspects of the arguments have been accepted by looking at the nodes' annotations of the top-level graphical argument.

3.4 Certificate publication and revocation

Although the certification case will be considered the formal controlled information artefact, it may be required (e.g. for contractual purposes) to publish a paper document as a formal deliverable. In this case, the standard reporting tools would be used to create such a publication.

Before publication, the software certificate manager would validate all the DNR elements in the case. This would determine whether the underlying evidence had changed since it was last checked.

- *Unchanged* - if the certification case has been reviewed and audited, the certificate is still valid and can be published against the particular build.

- *Changed* - the underlying evidence has changed. Therefore the certification case must be reviewed to determine whether the top-level claims are still valid. The ASCE impact analysis algorithms could be used to help the reviewer determine which claims depend on the evidence changed and might need to be revisited.

4 Future directions

Given the extensible architecture of the tool, there are a number of aspects of use that can be supported with additional notations and plugins. These include:

- **Formal modelling and reasoning engines** – we have already implemented notations and algorithms (e.g. to support Fault trees) that formally analyse the structure of the representation used and propagate information across it. We are actively investigating integration with additional reasoning engines, although we anticipate there may be difficulty in faithfully in implementing the full underlying semantics of the reasoning engine.

- **Notation enhancements** – each core notation schemas in ASCE has been tuned for a particular purpose. One of our active areas of development is to enhance existing notations and develop new ones to support particular forms of analysis, and to distinguish particular kinds of evidence used (e.g. analytical, probabilistic and qualitative). Depending on the schema definition it is possible for different display rules to visually amplify this information and present it as decoration on the main display (e.g. as “traffic lights”).

- **Modular cases and integration** – as larger arguments are developed it is necessary to refactor the argument and modularise it to allow for different elements to be maintained by different stakeholders and to evolve separately. Existing facilities such as DNRs allow for changes to be managed across the interfaces between such modules. However, more work is needed in the future to explore the issues arising from such use in a way that supports pragmatic usage but at the same time provides a useful degree of automated integration as the collection of interdependent modules evolve.

5 Conclusions

Assurance and safety cases are mature concepts that already have wide use to support the assurance of dependable systems. Such concepts can be applied to software certification, in that claims need to be established and backed up by evidence about the product or its development process.

There are a number of open issues that are relevant for both assurance cases and software certification. In [8] we summarised some of the research areas we have been working, which included the use of formality and models to support the validation of the assurance case, the relationship to standards and how the safety case goal-based approach can be deployed in other application areas.

ASCE is a commercial tool used to develop and manage assurance and safety cases, and in this paper we have outlined how it could be used to support the development and management of software certificates. Together with the ASCE user community, we are actively developing the product and its associated notations and plugins. Our aim is to support an increasingly wide range of application scenarios associated with heterogeneous information management and the demonstration of systems assurance.

5 References

[1] Toulmin, S.E. (1958) The Uses of Argument, Cambridge University Press, Cambridge, England.

[2] Bishop, P. & Bloomfield, R. A Methodology for Safety Case Development, Safety-Critical Systems Symposium, Birmingham, UK, Feb 1998

[3] Adelard (1998) ASCAD—The Adelard Safety Case Development Manual ISBN 0 9533771 0 5

[4] ASCE home page: http://www.adelard.com/software/asce

[5] Luke Emmet & George Cleland, Graphical Notations, Narratives and Persuasion: a Pliant Systems Approach to Hypertext Tool Design, in Proceedings of ACM Hypertext 2002 (HT'02), June 11-15, 2002, College Park, Maryland, USA.

[6] Kelly, T. Arguing Safety A Systematic Approach to Managing Safety Cases (1998). PhD Thesis, available at http://www.cs.york.ac.uk/ftpdir/reports/YCST-99-05.ps.gz

[7] Call for Papers - ASE Workshop on Software Certificate Management (SoftCeMent)

[8] P.G. Bishop, Robin Bloomfield and Sofia Guerra. The future of goal-based assurance cases. In Proceedings of Workshop on Assurance Cases. Supplemental Volume of the 2004 International Conference on Dependable Systems and Networks, pp. 390-395, Florence, Italy, June 2004.