Extending the Safety Case Concept to Address Dependability

Georgios Despotou, BEng (Hons), MSc, MIEE; Tim Kelly, MA, PhD; Department of Computer Science, University of York, YO10 5DD, UK

{georgios.despotou, tim.kelly}@cs.york.ac.uk

Keywords: Dependability, Safety Case, Argument, Quality Attributes, Trade-offs, System Reasoning

Abstract

A safety case is a well-reasoned argument, supported by evidence that a system is acceptably safe to operate in a particular context. For many, evolving a safety case in step with the design has proved to be an effective means of identifying and addressing safety concerns during a system's lifecycle. However, ultimately safety cases address only one system attribute - safety. Increasingly, the idea of extending the well-established concept of the safety case to address wider dependability concerns is being discussed. Attempting to address all dependability attributes can result in competing objectives. As a consequence, there are trade-offs among the dependability attributes that need to be resolved in order to achieve the optimum dependability characteristics for the system. Furthermore, the balance of these trade-offs can depend heavily upon the context in which the system operates.

In this paper we examine the suitability of extending existing methodologies and concepts from safety case development practice to address the wider concerns of dependability arguments. We will discuss existing approaches to managing trade-offs between competing design objectives and explain how trade-offs may be supported within the Goal Structuring Notation (GSN) framework. In particular we examine how trade-off resolution during the evolution of the dependability objectives, contributes to establishing a final dependability argument.

Introduction

It is now commonplace that developers of safety critical systems are required to produce a corresponding safety case – communicating an argument, supported by evidence that a system is acceptably safe to operate. Examples of application domains for such systems include the defence and railway sectors. In such domains the description of requirements for the safety case product, as well as the processes for development, are described in detail by the respective safety standards such as the U.K. Defence Standard 00-55 (ref. 1) and the U.K. Health and Safety Executive Railway Safety Case Regulations (ref. 2). Having existed as a core requirement of such standards for many years, safety case development practice has significantly matured over the last decade and is now widespread in safety engineering. Today, safety cases are well supported by development methodologies and notations such as the Goal Structuring Notation (GSN) – presented later in the paper.

In overall terms a safety case should provide an argument accompanied by evidence that all safety concerns and risks have been correctly identified and mitigated. However, a safety case ultimately addresses only one attribute – safety. In addition to safety, there are also other important system attributes that need to be addressed, such as maintainability and security. Dependability is a system property that is perceived to encompass these wider attributes. In abstract there is an agreement concerning the overall notion of dependability. However in practice there can be differences on how it is interpreted based on the developers' background and domain of application. In the next section the most common definitions of dependability are presented, along with the one adopted for this research, explaining in brief what it means for the system lifecycle.

The idea of reasoning about dependability is inspired by safety cases. It is common observation that providing an argument about a system's dependability can increase the assurance of the developers and the users, about the operation of a system. However shifting the focus from only one attribute (i.e. safety) and attempting to address all attributes equally, poses two main obstacles that need to be overcome; the evolution of this type of argument, and the resolution of conflicts between evolving (into specification) attribute objectives.

Dependability

Dependability is an emergent property having many viewpoints. Dependability has been described as *"the system's characteristic that justifies placing one's reliance on it"* (ref. 10). In order to place reliance on a system, the system needs to satisfy certain attributes such as reliability, availability and integrity. Whilst currently there is no consensus on the attributes that define dependability, many agree that is a composite term. For example, Laprie describes dependability as: *"the system property that integrates such attributes as reliability, safety, confidentiality, integrity and maintainability, is achieved by means of fault tolerance, fault prevention, fault removal, and fault recognition and it may be compromised by threats, faults and errors"* (ref. 10).

A system can have required attributes (e.g. performance, flexibility) that are not explicitly included in Laprie's definition. Furthermore in Laprie's view, all of dependability's attributes have as their epicentre the 'science of faults'. According to the context of the system, attributes can be related differently with each other and this will have an impact on the system's development. For example, there can be applications when reliability and safety are synonymous and others where they are clearly not. Prasad captured the dynamic and multi-attribute element of dependability in her definition (that has also been adopted in our research): *"We define dependability as a variable sized vector of attributes describing overlapping desiderata, chosen subjectively, in accordance to the stakeholders' particular requirements"* (ref. 12).

As seen, dependability is a system property that consists of different attributes, and can have different viewpoints in respect to the (subjective interests of the) different system stakeholders. In the following sections we will explain some fundamental concepts of the safety case practice, and how extending it will introduce the concept of dependability case. In addition, we will describe the impact the characteristics of dependability have on the process of extending several fundamental concepts from the safety development practice, in order to establish a dependability argument.

Safety Cases

The concept of the 'safety case' has already been adopted across many industries (including defence, aerospace and railways). Studying the safety standards relating to these sectors, it is possible to identify a number of definitions of the safety case. Although some of them can slightly differ than others, the core concept is that a safety case should communicate a clear, comprehensive and defensible argument that a system is acceptably safe to operate in a particular context. The definition of the safety case underlines some important aspects that need to be considered when referring to a safety case. Above all, the safety case exists to communicate an argument. It is used to demonstrate how someone can reasonably conclude that a system is acceptably safe from the evidence available. A safety case is a device for communicating ideas and information, usually to a third party (e.g. a regulator). In order to do this convincingly, it must be as clear as possible. The system to which a safety case refers can be anything from a network of pipes or a software configuration to a set of operating procedures. The concept is not limited to consideration of conventional engineering design. Absolute safety is an unobtainable goal. Safety cases are there to convince someone that the system is acceptably safe (safe enough when compared against some definition or notion of tolerable risk). Finally, context-free safety is impossible to argue. Almost any system can be unsafe if used in an appropriate or unexpected manner. It is part of the job of the safety case to define the context within which safety is to be argued. Safety cases consist of three principal elements: Requirements, argument and evidence as shown in Figure 1.

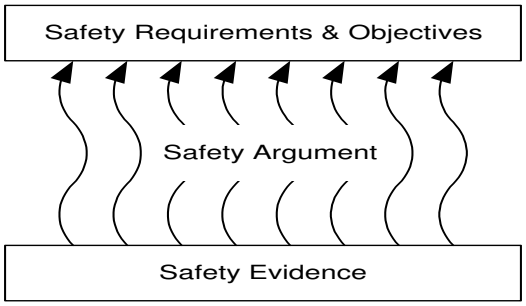

Figure 1 – The Role of Safety Argumentation

The safety argument is that which communicates the relationship between the evidence and objectives. Both argument and evidence are crucial elements of the safety case that must go hand-in-hand. Argument without supporting evidence is unfounded, and therefore unconvincing. Evidence without argument is unexplained – can be unclear that safety objectives have been satisfied. Establishing a safety argument and creating a safety case is not a single process that takes place at a single defined point of the system's lifecycle. Safety standards, such as the U.K. Defence Standard 00-56 (ref. 15) and the Ship Safety Management Handbook JSP430 (ref. 16), now require that safety case development be treated as an evolutionary activity that is integrated with the rest of the design and safety lifecycle. For example, Defence Standard 00-56 states the following:

> *"The safety case should be initiated at the earliest possible stage in the safety Programme so that hazards are identified and dealt with while the opportunities for their exclusion exist".*

In addition, the standard specifies that at least three versions of the safety case should be constructed:

- **Preliminary safety case** – After definition and review of the system requirements specification.
- **Interim safety case** – After initial system design and preliminary validation activities.
- **Operational safety case** – Prior to in-service use, including complete evidence of requirements satisfaction

At each stage of the evolution of the safety case, the safety argument is expressed in terms of what is known about the system being developed. At the early stages of project development the safety argument is limited to presenting high-level objectives, as design and safety knowledge increases during the project these objectives (and the corresponding argument) can be expressed in increasingly tangible and specific terms (as depicted in fig.2).

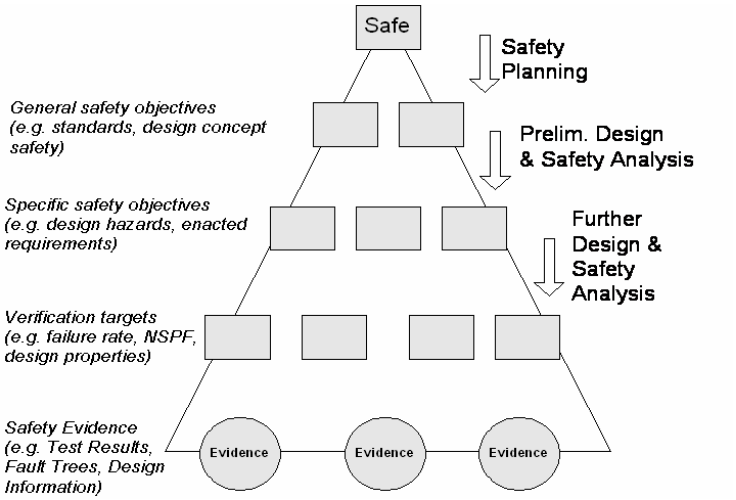

Figure 2 – Evolution of an Argument

Conventionally the safety case is thought of as a report (containing information about the system, its use, the hazards identified, description of hazard mitigations, and an overview of the evidence that all hazards sufficiently addressed). Whilst the report-oriented view is helpful, when adopting such a view it is often possible to lose sight of the logical chain of reasoning (the safety argument) that should be running through the safety case. Safety arguments are most typically communicated in existing safety cases through free text. Although it is possible to communicate clear arguments in the textual narrative of a report, many arguments expressed this way can be poorly expressed and difficult to comprehend. An example of a poorly expressed argument from a real safety case is shown in figure 3.

For hazards associated with warnings, the assumptions of [7] Section 3.4 associated with the requirement to present a warning when no equipment failure has occurred are carried forward. In particular, with respect to hazard 17 in section 5.7 [4] that for test operation, operating limits will need to be introduced to protect against the hazard, whilst further data is gathered to determine the extent of the problem.

Figure 3 – A Poorly Expressed Safety Argument

In the context of developing, agreeing, and maintaining the safety arguments within the safety case, the biggest problem with the use of free text is in ensuring that all stakeholders involved share the same understanding of the argument. Without a clear and shared understanding of the argument, safety case management is often an inefficient and ill-defined activity. The next section describes a structured technique – the Goal Structuring Notation (GSN) – developed to address the problems of clearly expressing and presenting safety arguments.

The Goal Structuring Notation (GSN)

The Goal Structuring Notation (GSN) (ref. 3)- a graphical argumentation notation - explicitly represents the individual elements of any safety argument (requirements, claims, evidence and context) and (perhaps more significantly) the relationships that exist between these elements (i.e. how individual requirements are supported by specific claims, how claims are supported by evidence and the assumed context that is defined for the argument). The main symbols of the notation are shown in Figure 4 (with example instances of each concept). The principal purpose of a goal structure is to show how goals (claims about the system) are successively broken down into sub-goals until a point is reached where claims can be supported by direct reference to available evidence (solutions). As part of this decomposition, using the GSN it is also possible to make clear the argument strategies adopted (e.g. adopting a quantitative or qualitative approach), the rationale for the approach (assumptions, justifications) and the context in which goals are stated (e.g. the system scope or the assumed operational role.

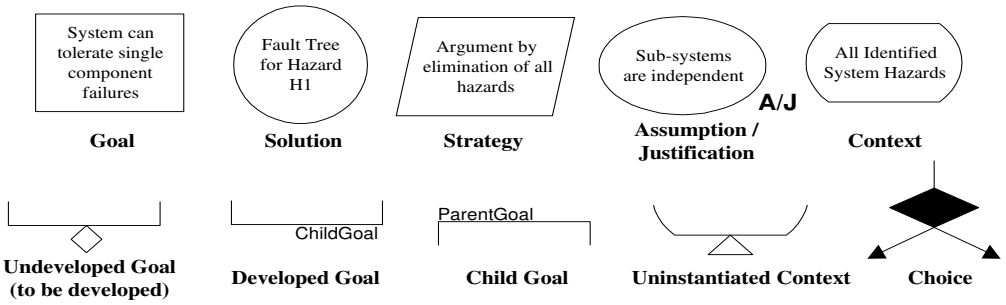

Figure 4 – Principal Elements of the Goal Structuring Notation

Figure 5 shows the usage of GSN elements by presenting an example of a goal structure.

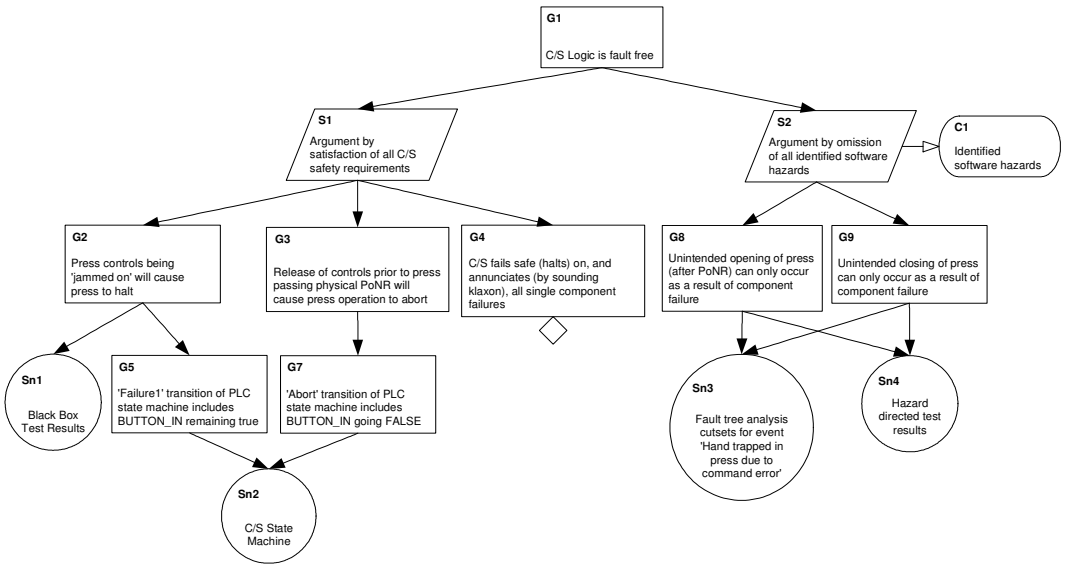

Figure 5 – An Example Goal Structure

GSN has been widely adopted by safety-critical industries for the presentation of safety arguments within safety cases (particularly within the UK). Being a mature and widely accepted notation, GSN has been suggested by many as a possible basis for helping to structure dependability cases. In the following sections, we further define the concept of the dependability case and explore the challenges faced in extending the application of GSN to representing multi-attribute dependability arguments.

Introducing the Concept of Dependability Cases

Although the dependability case is a relatively new and untested concept, the idea of developing 'cases' for system attributes other than safety is not new. Maintainability cases are, for example, a requirement of the U.K. Defence Standard 00-40 (ref. 4). Similarly, the Information Technology Security Evaluation Criteria (ITSEC) (ref. 17) require an explanation of why a system has met its required security level – implicitly this is a demand for a 'security case'.

Defence Standard 00-40 defines the reliability and maintainability (R&M) case as *"a reasoned audible argument created to support the contention that a defined system satisfies the R&M requirements"*. The US Naval Research Laboratory do not explicitly mention the term security case; however they have previously worked on the development and evaluation of *" a map of an argument that mission critical information is adequately protected by a system in its larger environment"* (ref. 6).

Table 1 shows typical objectives, arguments and evidence required when reasoning about different dependability attributes.

	Objective	Typical Argument	Typical Evidence
Safety	System is adequately safe	Hazard mitigation argument	Hazard Analysis, Causal analysis
Reliability	System meets reliability requirements	Enough redundancy, resilient components	Testing / Simulation, Markov analysis
Maintainability	System meets maintainability requirements	Modular cohesive design, plug and play devices, ease of replacing components	Expert opinion, simulation.
Security	Mission critical information is adequately protected	Assets protection argument	Access control, policies, MOATs

Table 1 – Objectives, Arguments and Evidence for the Dependability Attributes

It is possible to simply define a dependability case as a clear and defensible argument, that a system is acceptably dependable in a certain operational context. Whilst being correct, such a definition can mask the practical difficulties surrounding the development of dependability case (e.g. even in defining 'acceptable dependability'. We will illustrate these difficulties by means of an example approach. Using GSN, attempting to argue that a system is dependable in relation to its attributes might result in a goal structure such as that shown in figure 6.

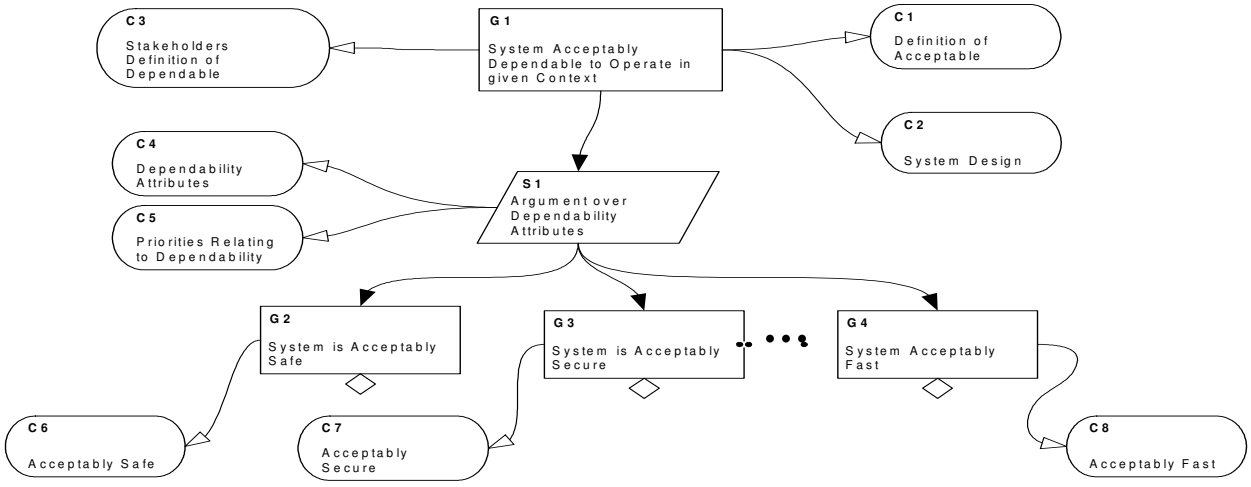

Figure 6 – A Possible Dependability Goal Structure

Figure 6 shows an attempt to establish a dependability argument simply by 'combining' all of the different attribute arguments. However, this argument fails to address the problem that the dependability attributes are non-orthogonal and can be interrelated to each other. Attempting to address all dependability attributes can result in competing objectives. For example, it is a common problem for availability to be in conflict with safety (e.g. in civil aviation). As a consequence, there are trade-offs among the dependability attributes that need to be resolved in order to achieve the optimum dependability characteristics for the system. Conflict may arise explicitly during specification stage (i.e. two directly conflicting requirements), or implicitly during design (i.e. a chosen design achieves one attribute's criteria but compromises another). When a conflict is encountered during the development of the system/argument then the developers need to make a trade-off – representing the best possible compromise between the conflicting objectives. The different attributes are heterogeneous and have different measures (such as safety risk, security risk and mean time to failure MTTF) and make use of different techniques (as illustrated in Table 1). As a consequence it is very difficult a common basis of comparison. The problems with the dependability goal structure can be summarised in two main categories:

- Representing the interrelationships within a dependability argument.
  As explained the attributes of dependability can be interrelated according to the design and implementation of the system. The dependability goal structure shown in figure 6 does not provide a means of capturing the relationships between goals addressing different dependability attributes (e.g. indicating where goals are in competition).

- Supporting the process of making trade-offs within the evolution of the dependability case
  Conflicts appear in many different types in the different stages of the evolution of a system. The outcome of a trade-off process should indicate the best possible choice throughout the development of the system. GSN can be used to capture the result

The following two sections discuss possible approaches to addressing these two problems.

A Modular Approach to Constructing Dependability Cases?

As explained, figure 6 depicts a dependability argument by arguing over the dependability attributes. Dependability cannot be expressed in a single metric (ref. 12). The definition of what is acceptably dependable (C1) does not automatically provide target objectives for each of the attributes. Instead, each of the attributes has different levels of acceptability that need to be defined (C6, C7 and C8). In addition, since the dependability attributes are interrelated and can lead to conflicts, defining the acceptability levels of each of the attributes has to be done with respect to the acceptable levels for the other attributes. For example, a system may be acceptably safe with respect to the level of security it offers and the performance it provides. An analogy can be drawn with the existing principle of ALARP (As Low As Reasonably Practical) principle in the safety domain – providing a means of trading off safety against cost. The international safety standard IEC61508 states that *"ALARP defines the tolerable risk as that risk where additional spending on risk reduction would be in disproportion to the actually obtainable reduction of risk"* (ref. 18). Similarly in order to reason effectively about dependability, we need to argue about each of the attributes in the context of the rest. Another important observation is that in order to find a satisfactory solution to the 'child' goals (G2, G3, and G4), a process for reconciling conflict and achieving trade-offs between the different attributes is required. Without a trade-off process, incrementally evolving a dependability argument in a similar manner to that practiced in safety case development (as illustrated in Figure 2) would be extremely difficult. Any change to the argument concerning one of the attributes has the potentially to affect other the arguments for other attributes.

In order to manage complex *safety* cases – in which there are complex relationships between arguments – the principles of compositional, modular, safety cases have already been established (Ref. 19). In this approach, the safety case for an overall system can be divided into a number of modules – containing the separate arguments and evidence for different aspects of system safety. For example, for a complex avionics platform the overall safety case can be reasoned about as the composition of separate arguments for each of the separate avionics subsystems. However, as described above for the dependability case, these arguments cannot be reasoned about in isolation. For example, it may only be possible to argue about the safety of one avionics subsystem in the context of assumed safe behaviour of another. To help manage the relationships that exist between safety case modules the concept of modular safety case interfaces (defining clearly the objectives, evidence, and assumed context of the case together

with any dependencies on other cases) and safety case contracts (recording how the dependencies between safety cases are resolved) have been defined. It is these principles of modularising an overall argument into separate but interrelated arguments that we believe offers a way forward in representing dependability cases.

As an illustration, Figure 7 presents an abstract overview of the dependability case architecture viewed in terms of separate modules of argument and evidence. In contrast to Figure 6, although separate arguments are presented for individual dependability attributes, each of the arguments is clearly set in the context of a trade-off argument that will separately argue about the resolution of trade-offs between conflicting dependability objectives. In addition to relating dependability arguments by means of the trade-off argument it is also possible that there may be more direct relationships that need to be captured between the arguments where they directly support one another. For example, a reliability argument may support a safety argument or there may be arguments and evidence that can be used to support both the safety *and* security argument. Such relationships have not been illustrated in the figure.

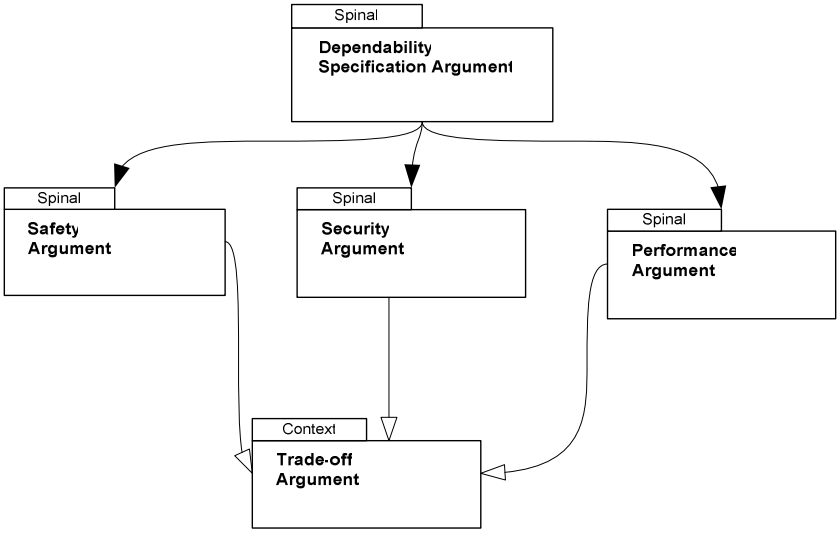

Figure 7 – A Modular View of a Dependability Argument

Modular arguments have the potential to represent a multi-attribute dependability argument and provide the basis for a dependability case that explicitly represents and reasons about the relationship between attributes. Exploring this potential is a continuing objective of our work.

Trade-off Studies

As a consequence of the fact that dependability addresses many (non-orthogonal) system attributes, trade-offs need to be made in order to resolve possible competing objectives, according to the overall best interest of the system's stakeholders. The appearance of trade-offs is a natural phenomenon during the development of a system. However, in the context of dependability, trade-offs become explicit as attributes can directly be in conflict with each other. The trade-off process needs to be incorporated within the dependability case lifecycle. A structured and systematic approach is required to making trade-offs in order to provide a compelling trade-off argument for use within the overall dependability case. Several methodologies already exist that could provide a basis for resolving a conflict and deciding a trade-off, such as the Architectural Trade-offs Analysis Method (ATAM) (ref. 11), the Multiple Criteria Decision Analysis (MCDA) (ref. 12), the SEI Cost Benefit Analysis Method (ref. 13) and the Analytical Hierarchy Process (AHP) (ref. 14). In this section, we provide an overview of three of these techniques (AHP, ATAM and CBAM) and discuss their suitability for helping manage the process of making tradeoffs between dependability attributes.

The Analytical Hierarchy Process (AHP) is a systematic method for comparing a list of objectives or alternatives. The objectives are set out along the two axes of a matrix and the stakeholders complete the matrix with values indicating the *relative* importance of the x, y objectives (1 = of equal importance, 9 = very strongly more important than), see Figure 8. For example, the matrix shown in Figure 8 reveals that the objective 1 (O1) is considered to be

strongly more important than objective 2 (O2) (shown by the value 6 in the cell O1,O2). By normalising the resultant table it is possible to produce a set of values that represent the overall importance of each of the objectives. The AHP is a purely numerical approach in selecting candidate objectives and could be prove to be useful in reasoning about conflicting dependability objectives, but there are two significant drawbacks. Firstly, the numbers do not provide the rationale and the criteria based on which they were assigned (making it difficult to present a compelling trade-off argument). The second drawback of the process is that the process cannot reflect qualitatively the associations between the different dependability attributes. (AHP assumes objectives are orthogonal).

	O1	O2	O3	O4
O1	1	1/6	¼	1/7
O2	6	1	3	5
O3	4	1/3	1	3
O4	7	1/5	1/3	1

Figure 8 –AHP Table

The Architectural Trade-off Analysis Method (ATAM) is a method that adopts a qualitative approach achieving trade-offs. Figure 8 presents the inputs and outputs of the ATAM.

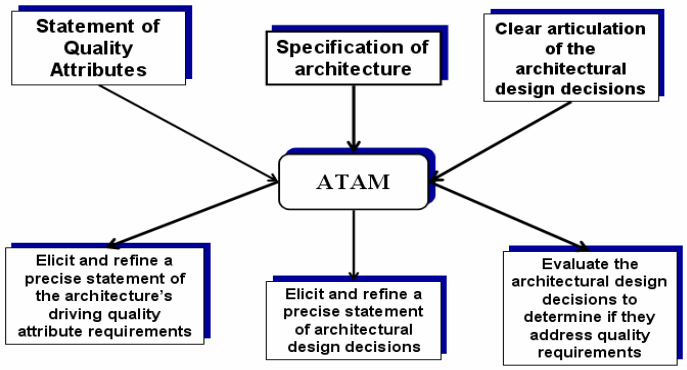

Figure 8 – ATAM Inputs and Outputs (ref. 11)

Among the important advantages of ATAM is that it provides a qualitative framework for evaluating the different design options. The stakeholders of a system derive a statement of desired quality attributes based on scenarios describing the use of the system. Each of the quality attributes required to be implemented is described in terms of stimuli for the architecture, how it will be implemented and what will be the impact of implemented measures. The ATAM allows prioritisation of one attribute with respect to another based on the utility of the system. Utility is the *overall 'goodness'* of the system, as expressed subjectively by its stakeholders. The prioritisation of different possible architectural strategies is complemented by the Cost Benefit Analysis Method (CBAM). This method provides concrete criteria for selecting the scenarios of most importance for the system. In CBAM the different scenarios are evaluated against the effort that is required to implement them (ultimately in monetary terms) and the benefit from their implementation (described in terms of utility gain).

ATAM provides a promising approach to reasoning about the trade-offs necessary within the dependability case. It facilitates rationale capture and provides means of determining a relative prioritisation between the attributes. Moreover, if combined with CBAM, it can provide a complete framework for making trade-offs including consideration of costs.

Summary

Dependability is a system property consisting of many different system attributes such as safety, security, reliability and maintainability. It is now widespread practice that any safety-critical system development should be accompanied by a 'safety case' consisting of a structured argument, supported by evidence. However, such a case typically only addresses safety in isolation of the other dependability attributes. In this paper, we have explored how existing concepts from safety case development (particularly, the presentation of arguments and incremental evolution of the case) could be extended to address the wider concerns of dependability case development. We have highlighted two key difficulties faced. Firstly, arguments concerning different dependability attributes are heavily interrelated. Such relationships must be captured and presented within the dependability case. Secondly, within dependability case evolution there must exist processes to handle the trade-offs that will inevitably encountered when attempting to satisfy multiple (potentially conflicting) objectives. The paper has highlighted possible

approaches to help address each of these difficulties. However, substantial (ongoing) work remains in establishing a dependability case development framework of equal maturity to that of current safety case practice.

References

1. Ministry of Defence, Directorate of Standardisation, Defence Standard, 00-55. Requirements of Safety Related Software in Defence Systems, 1997.
2. Health and Safety Executive. Railway Safety Case Regulations – Guidance on Regulations, HSE Books, 2000.
3. Kelly, T. P., Arguing Safety – A Systematic Approach to Managing Safety Cases. PhD Thesis, Department of Computer Science, University of York, 1998.
4. Ministry of Defence, Directorate of Standardisation, Defence Standard, 00-40. Reliability and Maintainability (R&M), 1999.
5. Kienzle M. D. Practical Computer Security Analysis. PhD Thesis. School of Engineering and Applied Science, University of Virginia, 1998.
6. Moore A., Strohmayer B. Visual NRM User's Manual. Centre for High Assurance Computing, Naval Reasearch Laboratory, Washington. Report: NRL/FR5540-00-9950, 2000.
7. McDermid A., Fenelon P. An Integrated Toolset for Software Safety Analysis. Journal of Systems and Software, July 1993.
8. Maxion A. R. Olszewski T R. Improving Software Robustness with Dependability Cases. 28th International Symposium on Fault-Tolerant Computing, Munich 1998.
9. Froome P., Jones C. Developing Advisory Software to Comply with IEC-61508. Prepared by Adelard for the Health and Safety Executive. ISBN: 0717623041, 2001.
10. Avizienis A., Lapries J., Randell B. Dependability of Computer Systems: Fundamental Concepts, Terminology and Examples. Workshop on Robot Dependability: Technological Challenges of Dependable Robots in Human Environments, Seoul 2001.
11. Kazman R., Klein M., Clements P. ATAM: Method for Architecture Evaluating the Quality Attributes of a Software Architecture. Technical Report CMU/SEI-200-TR004. Software Engineering Institute, Carnegie Mellon University, 2000.
12. Prasad D., Dependable Systems Integration using Measurement Theory and Decision Analysis. PhD Thesis, Department of Computer Science, University of York, UK, 1998.
13. Kazman R., Asundi J., Klein M., Making Architecture Design Decisions: An Economic Approach. SEI-2002-TR-035. Software Engineering Institute, Carnegie Mellon University, 2002.
14. Castillo L., Early FP Estimation and the Analytic Hierarchy Process.

    http://www.dpo.it/resources/papers.htm
15. U.K. Ministry of Defence, Directorate of Standardisation, Defence Standard, 00-56. Safety Management Requirements for Defence Systems. Ministry of Defence, 1996.
16. U.K. Ministry of Defence. JSP 430 – Ship Safety Management System Handbook. Ministry of Defence, 1996.
17. European Community advisory group Seniors Ocals Group Information Systems Security. Information Technology Security Evaluation Criteria. Department of Trade and Industry, United Kingdom, June 1991. Version 1.2.
18. International Electrotechnical Commission. IEC 61508 Functional Safety of Electrical/Electronic /Programmable Electronic Safety Related Systems. The IEC, 1998.
19. Kelly T., Managing Complex Safety Cases. Presented at the 11th Safety Critical Systems Symposium (SSS'03),February 2003 (Proceedings published by Springer-Verlag)

Biographies

Georgios Despotou BEng (Hons), MSc, MIEE. Research Student, Department of Computer Science, University of York, York, YO10 5DD, United Kingdom.
Phone: +44 1904 432792, Fax: +44 1904 432708
e-mail: georgios.despotou@cs.york.ac.uk

Georgios Despotou is a research student in the high integrity systems engineering (HISE) group, under the defence and aerospace research partnership (DARP). He was awarded in 2001 with a BEng (Hons) in the joint degree of Computer Systems Engineering from the University of Hull and in 2003 a Masters in Science in Software Engineering from the University of York, where he currently is a PhD student. Current research activities involve Dependability Cases and reasoning about Systems of Systems architectures.

Dr Tim Kelly MA, PhD. Lecturer, Department of Computer Science, University of York, York, YO10 5DD, United Kingdom.
Phone: +44 1904 432764, Fax: +44 1904 432708
e-mail: tim.kelly@cs.york.ac.uk

Dr Tim Kelly is a Lecturer in software and safety engineering within the Department of Computer Science at the University of York. He is also Deputy Director of the Rolls-Royce Systems and Software Engineering University Technology Centre (UTC) at York. His expertise lies predominantly in the areas of safety case development and management. His doctoral research focussed upon safety argument presentation, maintenance, and reuse using the Goal Structuring Notation (GSN). Tim has provided extensive consultative and facilitative support in the production of acceptable safety cases for companies from the medical, aerospace, railways and power generation sectors. Before commencing his work in the field of safety engineering, Tim graduated with first class honours in Computer Science from the University of Cambridge. He has published a number of papers on safety case development in international journals and conferences and has been an invited panel speaker on software safety issues.