Argument-based approach to computer system safety engineering

Tangming Yuan* and Tim Kelly

Department of Computer Science,
University of York,
Deramore Lane, York YO10 5GH, UK
Fax: +44-(0)-1904-432708
E-mail: Tommy.Yuan@cs.york.ac.uk
E-mail: Tim.Kelly@cs.york.ac.uk
*Corresponding author

Abstract: Safety case development is not a post-development activity, rather it should occur throughout the system development lifecycle. The key components in a safety case are safety arguments. Too often, safety arguments are constructed without proper reasoning. Inappropriate reasoning in safety arguments could undermine a system's safety claims, which in turn contributes to safety-related failures of the system. To address this, we argue that informal logic argument schemes have important roles to play in safety arguments construction and review process. Ten commonly used reasoning schemes in computer system safety domain are proposed against the safety engineering literature. The role of informal logic dialogue games in computer system safety arguments reviewing is also discussed and a dialectical model for safety argument review is proposed. It is anticipated that this work will contribute toward the development of computer system safety arguments, and help to move forward the interplay between research in informal logic and research in computer system safety engineering.

Keywords: safety arguments; argument schemes; dialectics; safety arguments reviewing.

Reference to this paper should be made as follows: Yuan, T. and Kelly, T. (xxxx) 'Argument-based approach to computer system safety engineering', *Int. J. Critical Computer-Based Systems*, Vol. X, No. Y, pp.000–000.

Biographical notes: Tangming Yuan is a Teaching Fellow in Software Engineering within the Department of Computer Science at the University of York. His research interests include argument and dialogue, and their applications to human-computer dialogue, agent communication and computer system dependability argument.

Tim Kelly is a Senior Lecturer in the Department of Computer Science at the University of York. His research interests include safety case management, software safety analysis and justification, software architecture safety, certification of adaptive and learning systems, and the dependability of 'systems of systems'. He has supervised a number of research projects in these areas with funding and support from Airbus, BAE Systems, Data Systems and Solutions, DTI, EPSRC, ERA Technology, Ministry of Defence, QinetiQ and Rolls-Royce. He has published over 140 papers on high integrity systems development and justification.

1 Introduction

As society's dependence on computer systems continues to increase, the importance of computer systems' dependability increases accordingly. The term 'dependability' when applied to computer system, can be considered as a property of the system that equates to its trustworthiness – the degree of user confidence that the system will produce the consequences for which it was designed, and no adverse effects in its intended environment. A dependable computer system typically exhibits one or more of the following quality attributes: availability, reliability, safety and security. For example, for an online railway tickets booking system to be justifiably deemed dependable, we would expect that it achieves a fairly consistent high standard availability (the system is not often 'offline' or 'clogged up' with too many users), reliability (the system does not fail, carries out bookings accurately and in a timely fashion), and security (the system can protect itself from accidental or deliberate external attacks, e.g., prevent unauthorised access of personal booking details). For a safety critical system, e.g., a railway signal control system, it is deemed dependable if the system does not damage people or the system's environment or severe economic loss even if the system fails.

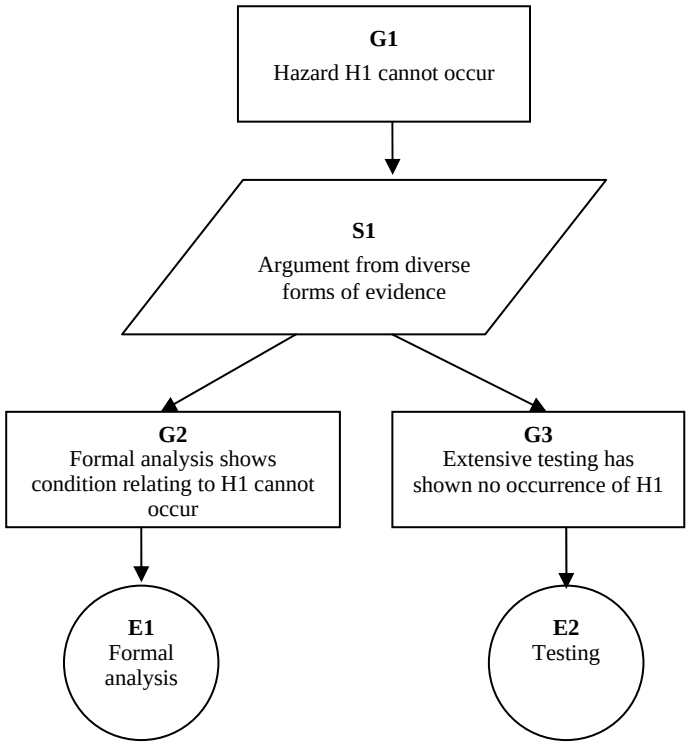
Unfortunately, the assessment of dependability of computer systems has long been acknowledged to be difficult (cf. Bloomfield and Littlewood, 2003; Littlewood and Wright, 2007; Weaver et al., 2002, 2003). The field of computer system engineering suffers from a pervasive lack of product evidence about the incidence and severity of system failures due to systematic nature of system failures (Weaver et al., 2002, 2003). Most computer system standards, e.g., IEC 61508 (International Electrotechnical Commission, 1998), DO178B (RTCA Inc. and EUROCAE, 1992), DS 00-55 (UK Ministry of Defence, 1997) for the development of safety critical systems, recommend a set of techniques and identify processes for different safety integrity levels or development assurance levels. The fundamental assumption underlying such a process-based approach is that both the developer and the assessor accept that, by following the process of applying these techniques, the system achieves the required level of dependability. There is however some evidence that the assumption does not always hold (Harrison, 1999). Indeed, it is not possible to demonstrate a direct causal relationship between the use of prescribed processes and high levels of safety. It is possible to conceive of situations where the prescribed processes have been followed, but there remain software contributions to hazards which are not sufficiently controlled.

In this paper that follows, we discuss a more recent, argument-based approach to computer system dependability in general and to computer system safety in particular. We firstly give a brief introduction of argument-based approach to computer system safety engineering. We then argue that informal logic argument schemes have an important role to play in representing and reviewing of safety arguments, and propose ten schemes that are commonly used in computer system safety domain. We finally discuss the role of dialectics in safety case development and proposed a dialectical model in facilitating safety arguments reviewing process.

2 Safety arguments

A recent approach for both achieving and demonstrating computer system dependability is to adopt an argument-based approach (e.g., McDermid, 2001; Bishop and Bloomfield, 1998; Kelly, 2007), a.k.a. *goal-based* approach. We adopt the phrase *argument-based* in the remainder of this paper on the ground that *argument* is the most commonly used concept in the area of both safety case development and informal logic. In an argument-based approach, the stakeholders first agree on the goals for which assurance is required (e.g., this device must not harm people), then the developers produce specific claims (e.g., the radiation delivered by this device will never exceed intensity level x) and an argument to justify the claims based on verifiable evidence (e.g., there is a mechanical interlock on the beam intensity and here is evidence, derived from extensive testing, that it works). This approach involves the construction, negotiation and assessment of valid and coherent arguments of dependability, and the selection of techniques to collect evidence supporting dependability claims. Commercial tools have been developed for this purpose, e.g., the *Adelard Safety Case Editor* (ASCE) supports Kelly's (1999) *goal structuring notation* (GSN) and the claim-argument-evidence (CAE) style arguments. Figure 1 shows an example use of the core components of the GSN notation. A rectangular box represents a claim or a sub-claim, a diamond represents the arguing strategy, and a circular represents a piece of evidence. The argument in Figure 1 shows that in order to achieve the *G1* both legs of evidence are collected and the arguing strategy is *from diverse forms of evidence.*

Figure 1 Example use of GSN



While the majority of these existing notations offer convenient tools for engineers to develop safety arguments, they offer little assistance for someone to challenge and critique the assumptions made. Too often, safety arguments are constructed with inappropriate reasoning, e.g., using the wrong reasons, drawing the wrong conclusion and/or omission of key evidence (cf. Greenwell et al., 2005). Inappropriate reasoning in a system's safety argument could undermine the system's safety claims, which in turn contributes to a safety-related failure of the system. To address this, we need to extend the existing tools, methods and standards for computer system safety engineering to facilitate *good* argument, and to promote a new computer system safety engineering literacy. The sections follows discuss our proposed extensions on the use of argument schemes and dialectics in representing and reviewing safety arguments.

3 Safety argument schemes

We would like to represent *good* arguments in the first place. We would also wish to be able, with reasonable objectivity, to assess whether a given argument is sufficiently convincing, or whether it hides any weaknesses that should be of concern. One way to assess argument is the *fallacy approach*, where arguments were categorised in terms of traditional fallacies. The approach however suffers from the fact that many instances of traditional fallacies appear to be good arguments (cf. Walton et al., 2008). A modern approach to argument assessment is the *argument schemes* approach. Unlike fallacies, schemes are not understood as in principle poor forms of arguments, but as general structures which may convey good reasoning (cf. Walton et al., 2008; Perelman and Olbrechts-Tyteca, 1969; Walton, 1996). Two devices are provided by schemes:

1 when constructing arguments, they provide a repertory of forms of argument (scheme) to be considered, and a template prompting for the pieces that are needed, i.e., the premises and conclusion

2 when assessing argument, each argument scheme provides a set of *critical questions* (CQs) that can be used to examine the plausibility of an argument.

The set of CQs indicates points of weaknesses (assumptions) of an argument where doubts can be placed, and challenges and attacks can be made against. For example, consider Walton's (1997, p.210) analysis of the scheme of *argument from expert opinion*, a special form of *argument from authority*:

- *Major premise:* Source E is an expert in subject domain D containing proposition A.
- *Minor premise:* E asserts that proposition A (in domain D) is true (false).
- *Conclusion:* A may plausibly be taken to be true (false).

There are six basic CQs matching the appeal to expert opinion, as indicated in Walton (1997, p.223):

CQ1 *Expertise question:* How credible is E as an expert source?

CQ2 *Field question:* Is E an expert in the field that A is in?

CQ3 *Opinion question:* What did E assert that implies A?

CQ4 *Trustworthiness question:* Is E personally reliable as a source?

CQ5 *Consistency question:* Is A consistent with what other experts assert?

CQ6 *Backup evidence question:* Is E's assertion based on evidence?

Informal logic, by no means, provides all domain specific schemes for the computer system safety engineering, but knowledge gained from the study of informal logic argument schemes can help to generalise domain specific forms of reasoning. Against computer system safety engineering literature, ten argument schemes are proposed as follows:

3.1 Argument from hazard avoidance

A safety case is 'reasonable confirmation that risks are managed to as low as reasonably practical (ALARP)' [Haddon-CAVE QC, (2009), p.544]. The first step in constructing a safety case is hazard identification and assessment. System safety requirements are then generated to define the defences against the hazard and how the hazard will be managed if it arises. Following Kelly (1999), the scheme of *argument from hazard avoidance* is proposed as follows:

- *Premise:* All identified hazards for system X are addressed.
- *Conclusion:* System X is acceptably safe.
- CQ1. How complete is the list of identified hazards?
- CQ2. How accurate is each of identified hazards?
- CQ3. Are the hazards adequately controlled?

The scheme is usually used at the top level of a safety argument (Kelly, 1999). Further lower level arguments need to be produced to argue that each of hazards has been managed to *ALARP*. CQ1 and CQ2 concern the completeness and correctness of the list of identified hazard. To respond to these questions, engineers experience and the use of standard techniques [e.g., hazard and operability studies (*HAZOPS*)] can be sought to justify the confidence of the hazard identification and assessment. CQ3 concerns whether each identified hazard has been controlled properly.

Failing to ask the CQs will lead to undiscovered hazards and inappropriate handling of a hazard. For example, in September 1993, a Lufthansa Airbus A320 landed at Warsaw airport in a thunderstorm. Upon landing, the brakes on the computer-controlled brake system did not function for about nine seconds. The aircraft ran off the end of the runway, collided with an earth bank and started to burn. A safety feature on the aircraft had stopped the deployment of the braking system because this can be dangerous if the plane is in the air. The braking system will be deployed only when the plan lands on both wheels. However, when crosswinds generated during a thunderstorm caused the plane to tilt, the plane landed on one rather than two wheels. The cause of this accident is therefore an undiscovered hazard leading to a system specification error (cf. Sommerville, 2007).

3.2 Argument from functional decomposition

Often, safety critical systems are too large and/or too complex to be addressed as a whole. One way to simplify this is to decompose the system (e.g., via the divide and conquer technique) into sub-components which are hopefully easier to be addressed. Following Kelly (1999), the scheme of argument from functional decomposition is proposed as follows:

- *Premise 1:* All safety-related functions of system X are safe.
- *Premise 2:* There are no hazardous interactions between functions.
- *Conclusion:* System X is acceptably safe.
- CQ1: Is the list of safety-related functions complete?

Again, this scheme is usually used at the higher level of a safety argument (Kelly, 1999). Further lower level arguments need to be constructed to support the safety claim of each safety-related function and each interaction among all the functions. CQ1 concerns the completeness of the list of safety-related functions. To respond to this question, the experience of the creator of the list of safety-related functions and the use of technique like function hazard analysis can be appealed to justify the confidence of the function decomposition.

3.3 Argument from probabilistic fault tree analysis

Fault tree analysis (FTA) method is widely accepted in safety engineering to quantitatively determine the probability of a safety hazard (Kelly, 1999). A FTA starts with a hazard at the root of the tree and then identifies the states that can lead to that hazard, continues until reaching the root causes of the hazard. Boolean logic can be used to link a series of lower level states. When numerical probabilities have been provided for the root causes of the hazard and probability analysis has been possible, a quantitative claim can be put forward regarding the probability of the hazard. The scheme of *argument from probabilistic FTA* is proposed as follows.

- *Premise 1:* Probability FTA for hazard *X* indicates the probability of occurring of *X* is *p*.
- *Conclusion:* the probability of occurring of *X* is *p*.
- CQ1. Is the tree an accurate representation of the causes of hazard *X*?
- CQ2. Are root causes of the fault tree independent?
- CQ3. Is the failure probability for each root cause accurately represented?
- CQ4. Does historical evidence support the fault tree quantitative result?

If the fault tree is not valid, the claims derived from the tree will not stand. CQ1 is therefore concerned with this. Should there be a common failure mode between the root causes; the probability calculation would not be mathematically valid. CQ2 is therefore designed to cater for this. The quality of reliability estimate for the basic event is also very important; CQ3 is envisaged to question the reliability of the basic events. Also,

component level fault trees can sometimes miss higher level system effects, so a fourth CQ is recommended to compare the estimate from the fault tree with historical datasets.

3.4 Argument from historical data

In 2006, the RAF NIMROD MR2 aircraft XV230 suffered a catastrophic mid-air fire while it was on a routine mission in Afghanistan, leading to the total loss of the aircraft and the death of all 14 members on board. The subsequent enquiry (Haddon-Cave QC, 2009) showed that the leaked fuel is the source for the fire and that the hazard (H73-fuel system leakage) documented in its safety case was improperly sentenced. H73 was addressed by being classified as *improbable* based on its in-service accident database. To analyse this type of reasoning, the scheme of *argument from historical data* is proposed below.

- *Minor premise:* Source *S* shows the probability of the past occurrence of an event *E* is X.
- *Major premise:* The potential occurrence of event *E* is *X* which is qualified as *P* according to certain standards.
- *Conclusion:* The potential occurrence of event *E* is *P*.

An example instance of the scheme of argument from historical data from the Nimrod inquiry (Haddon-CAVE QC, 2009) is outlined below.

- *Minor premise:* In-service data shows that the past occurrence of fuel system leakage is within the range of 10^{-6} to 10^{-7}.
- *Major premise:* The potential occurrence of fuel system leakage is 10^{-6} to 10^{-7} which is qualified as *improbable*.
- *Conclusion:* the potential for fuel system leakage is *improbable*.

The following are the proposed CQs associated with this scheme which could be used to validate the strength of the argument. An example criticism made by Haddon-CAVE QC (2009) matching each CQ is given in brackets in italics.

CQ1 *Source question:* how credible is *S* as a source?

[e.g., a counter argument – "Incident database might not capture all minor fuel leak" (p.278)].

CQ2 *Consistency question:* Is *S* consistent with other sources?

[e.g., a counter-argument – "the maintenance personnel from the Nimrod Servicing Group estimated that the probability of fuel coupling leaks was far more frequent than the MRA4 generic data suggested" (p.280)].

CQ3 *Qualification question:* Does *X* really mean *P*? Or is the warrant backed with sufficient evidence or standard?

[e.g., someone might argue that 10^{-6} is *probable* although it is very low. As Haddon-CAVE QC puts "quantitative risk assessment is an art not a science. There is no substitute for engineering judgment" (p.546)].

CQ4 *Shadow of the future question:* How likely the past occurrence of event *E* represents its potential occurrence?

[e.g., a counter-argument – "The fact that something has not happened in the past is no guarantee that it will not happen in the future" (p.546)].

CQ5 *Modification question:* Have there been any changes that invalidate past historical experience? (e.g., Nimrod underwent modifications to permit in-flight refuelling).

Should the CQs be asked concerning the validity of the argument, H73 might not be improperly sentenced and the accident might be avoided.

3.5 Argument from formal verification

The application of formal methods to the development of safety-critical software has been advocated by a number of standards (e.g., DO-178B), and mandated by at least one (UK Defence Standard 00-56). Formal methods, as the ultimate static verification technique, use mathematical arguments that the implementation of a software system is consistent with its specification. The most optimal conclusion drawn from formal verification is that the targeted implementation conforms to its specification. It does not necessarily conclude the final system is safe because the question remains as to whether the formal model used is a sufficiently accurate representation of the reality of the implemented system although the model may be mathematically consistent. The Warsaw accident discussed in Section 3.1 above is an example of accident caused by a safety specification error. The scheme of argument from formal verification is proposed as follows:

- *Premise:* Formal verification shows system or component *X* conforming to its safety specification.
- *Conclusion:* System or component *X* conforms to its safety specification.
- CQ1. Is the formal verification properly performed?

Because formal verification is usually complex and error prone, CQ1 might be asked to check the correctness of the proof. To respond to this CQ, argument from authority (e.g., appeal to the analysers' experience) might be used to support the proof claim.

3.6 Argument from verification testing

Verification testing aims to provide evidence that a programme or component meets its safety specification. Similar to formal verification, verification testing does not necessarily conclude the final system is safe because the question remains as to whether the safety specification is a sufficiently accurate representation of the real needs of users of the system. The scheme for *argument from verification testing* is proposed as follows:

- *Premise:* Testing shows system or component *X* meets its safety specification.
- *Conclusion:* System or component *X* meets its safety specification.
- *CQ1*. How rigorous is the testing?
- *CQ2*. Is the testing properly performed?

By its very nature, exhaustive testing is usually not possible (Dijkstra, 1972). Various adequacy measures however can be placed to judge the confidence of the claim, e.g., specification coverage where equivalent partition and boundary value analysis are used to design the tests, and structural coverage where tests are designed to achieve statement, branch, condition, modified condition and decision coverage (MC/DC), multiple-condition and data flow coverage. Different industrial standards have different requirements for this, e.g., DO-178B standard mandates the use of MC/DC. CQ1 is concerned with such adequacy. CQ2 might also be asked to validate whether the tests have been properly exercised. To respond to this question, one might use argument from authority (e.g., appeal to the testers' experience, the use best practice tools, techniques and methods during testing) to support the testing claim.

3.7 Argument from validation testing

Validation testing checks how the programme works under its operational condition. The scheme for *argument from validation testing* is proposed as follows:

- *Premise:* Validation testing shows the system or component *X* is safe.
- *Conclusion:* System or component *X* is safe.
- CQ1. How accurately does the operational profile reflect the real use of the system?
- CQ2. Is the testing statistically significant?
- CQ3. Is the testing properly performed?

Often, operational profile is based on experience of other system which may not reflect the real use of the system, CQ1 is concerned with this. Further, it is important to generate a reasonably large and statistically significant number of failures to be confident that the reliability measurement is accurate. CQ2 is therefore designed to deal with this. CQ3 concerns whether the test is properly exercised. To respond to this question, one might use the argument from authority (e.g., appeal to the testers' experience) to support the testing claim.

It is rarely liable to conclude that the system is safe given the incompleteness nature of testing. Testing therefore simply provides some evidence, which is used together with other evidence, e.g., formal verification, to make a judgement about the system safety. This is discussed in the following section.

3.8 Argument from diverse forms of evidence

Diverse arguments involve using several items of evidence support the same safety claim whilst each argument/evidence can support the safety claim individually. The purpose of using diverse arguments is to increase confidence of a safety claim (Bloomfield and Littlewood, 2003; Littlewood and Wright, 2007; Weaver et al., 2002) where complementary items of evidence corroborate each other (Walton, 2009). This is demanded by a number of safety standards, e.g., UK Defence Standard 00-55. The scheme of *argument from diverse forms of evidence* is proposed as follows:

- *Premise:* For evidence e_1, $e_2 \ldots e_n$, each supports the safety claim *C*.
- *Conclusion:* Safety claim *C*.
- CQ1: Are e_1, $e_2 \ldots e_n$ independent to each other?

The key question associated with this scheme is concerned with the independence of individual item of evidence. Weaver et al. (2002) argue that items of evidence intended to support the same safety goal in complementary ways must be independent. Independence may be undermined if they all rely on the same incorrect base data, e.g., a programme control flow graph. They further argue that independence can be either conceptual or mechanistic. Conceptually different approaches are based on different underlying theories. For example, testing and static analysis are conceptually different approaches to developing evidence where one involves running the programme and the other does not. Mechanistically different approaches implement the same underlying theory in different ways. For example, code reviews done by human reviewer in comparison to by automated code analyser. As a general rule conceptual independence is more significant than mechanistic independence. The argument in Figure 1 shows a diverse argument with two items of conceptually independent evidence.

3.9 Argument from redundancy

A common form of improving computer systems dependability is the use of redundant components through the existence of either back-up or checking components. The redundant components implement a common specification into a number of versions (version 1, 2…*n*). There are two related forms of redundancy. A most commonly used form is to use the redundant components in parallel for some critical function, and their outputs are then compared using a voting system. Inconsistent outputs or outputs that are not produced in time are rejected, thus ensuring correct behaviour. Another form is to use the redundant components in sequence rather than in parallel. This form includes a test to check that a component has executed correctly, and alternative code that allows system to backup and repeat the computation if the test detects a failure. This form of argument is proposed as follows:

- *Premise:* Replicas of component (or system) X are used to ensure reliability.
- *Conclusion:* Component (or system) X is reliable.
- CQ1. Is the common specification of redundant components correct?
- CQ2. Is each of the replicas designed differently?

The key factor undermines the confidence of redundancy claim is the common failure of the redundant components. One cause of the common mode of failure is that the redundant components are designed against the common erroneous specification. CQ1 is concerned with this. The components should also be designed and built by different manufactures or by different development teams using different programming languages, platforms and algorithms to reduce the chance of such common mode failure. CQ2 is concerned with such independence of the redundant components. The enquiry of the Ariane 5 launcher accident reveals that the backup software used was a copy and behaved

in exactly the same way. Should CQ2 be asked, the Ariane accident might be prevented (Sommerville, 2007).

3.10 Argument from development process

In manufacturing, high quality product follows stringent development process (Sommerville, 2007). However, software quality does not necessarily follow standard development process (Harrison, 1999) as software is designed rather than manufactured. Factors such as individual skills and experience, novelty of the application and time/cost constraint, affect product quality irrespective of the process used. Development process (e.g., defensive design, stringent review) therefore does not provide direct evidence to the desired product quality attributes. It merely together with other conditions, e.g., the applicability of the process (in CQ1 below), developers' experience (in CQ2 below) and other constraints (in CQ3 below), provides an item of evidence that the development activities are properly carried out. Argument appealing to development process can be used to support another argument with direct evidence to the desired product quality, e.g., *argument from testing*. The scheme of *argument from development process* is proposed as follows:

- *Premise:* Standard process is used to carry out activity *A*.
- *Conclusion:* Activity *A* is properly carried out.
- CQ1. Is the process applicable to this situation?
- CQ2. Are the developers who carried out the activity sufficiently experienced?
- CQ3. Is there any time/cost constraint when carry out this activity?

These, then, are the commonly used argument schemes we have proposed for a computer system safety engineering domain. The argument schemes proposed above are patterns of *good* safety arguments generalised from safety engineering literature, they are in line with the current and previous research in *argument patterns* (e.g., Kelly, 1999; Kelly and McDermid, 1998; Hawkins and Kelly, 2010). The distinctive and core feature of argument schemes is that each scheme contains of a set of CQs. The CQs are envisaged to indicate implicit assumptions of an argument where points of attacks can be made. The set of CQs associated with each argument is not necessarily complete and could evolve as experience is gained (e.g., after major incidents). CQs are the key tools for reviewing and evaluating the strength and validity of safety arguments. Safety argument reviewing will be discussed in the following section.

4 Dialectical aspects

Safety arguments, serving as part of the system development process, are constructed, reviewed, negotiated and assessed by the developer and assessors (and/or relevant stakeholders) (Bishop and Bloomfield, 1998; Kelly, 2007). It is worth noting that safety arguments development and reviewing is a not a post-development activity, rather it should occur throughout the system development lifecycle. For a successful safety arguments reviewing, it requires both someone to develop and defend the safety

arguments and someone to challenge and critique the assumptions made. Too often, the latter part is missing (Kelly, 2008). The need of dialectics has also been reinforced by the recent issue of Defence Standard 00-56, as quoted below:

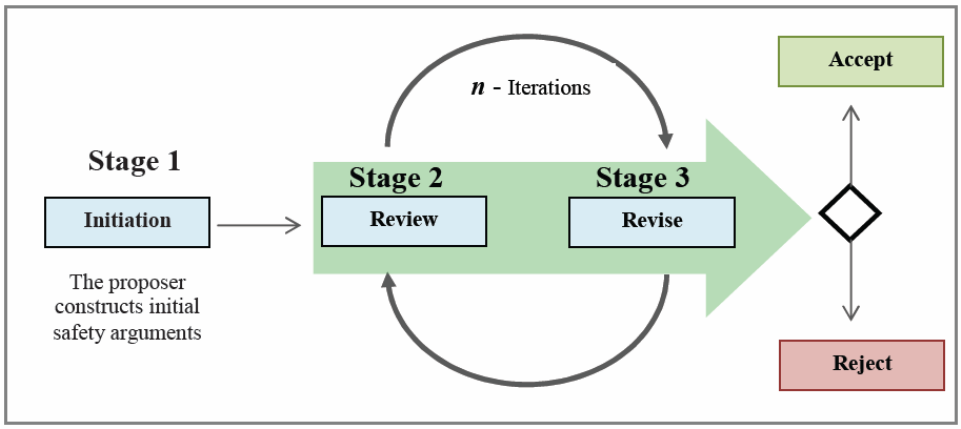
> "9.5.6 Throughout the life of the system, the evidence and arguments in the Safety Case should be challenged in an attempt to refute them. Evidence that is discovered with the potential to undermine a previously accepted argument is referred to as counter-evidence. The process of searching for potential counter-evidence as well as the processes of recording, analysing and acting upon counterevidence are an important part of a robust Safety Management System and should be documented in the Safety Case."

The importance of dialectal aspect in safety argument seems clear. To ensure fair and reasonable criticisms and responses taking place, a suitable dialogue model is needed to manage the interaction process as it unfolds (e.g., Walton and Krabbe, 1995). Studies of dialectics in the area of informal logic are of potential value here. *Dialectics* is seen as the branch of philosophy attempting to build models for *fair and reasonable dialogue* (Walton and Krabbe, 1995). A common approach within dialectics is to construct *dialogue games* (e.g., Hamblin, 1970; Walton and Krabbe, 1995; Walton, 1998; Mackenzie, 1990). A dialogue game can be seen as a prescriptive set of rules, regulating the participants as they make moves in the dialogues. These rules legislate as to permissible sequences of moves, and also as to the effect of moves on participants' *commitment stores*, conceived as records of statements made or accepted. Such dialogue games have received much recent interest from people working in human computer dialogue and in artificial intelligence (cf. Bench-Capon and Dunne, 2007; Reed and Grasso, 2007; Rahwan and McBurney, 2007; Yuan et al., 2007, 2008).

There are however many normative dialectical systems that have been proposed in the area of informal logic (e.g., Walton and Krabbe, 1995; Mackenzie, 1990). It is therefore necessary to select or develop a suitable dialectical model against the requirements for safety arguments reviewing process. A dialectical model for safety argument review has been proposed. The overall review process can be represented as shown in Figure 2 below. The process encompasses three distinct phases: the initiation, review and revise. It starts with an initiation of a proposal of safety arguments followed by reviews conducted by independent reviewers. The proposer then responds and revises the initial proposal in light of the criticisms made by the reviewers. The revised version of the safety arguments will be further reviewed until reviewers reject or accept the arguments or the proposer withdraws. The number of iterations n can be defined by mutual agreement of all participants if they wish to set such a limitation. However, the ultimate aim of the iterations is to reach a position where the safety arguments are mutually accepted by both proposer and reviewers.

For the initiation stage, a set of move types are designed for the proposer to construct initial safety argument. These move types are in line with GSN syntax including goal (claim), strategy, evidence, context, assumption and justification. An additional set of move types are available for the review and revise stage. These include counterargument, question, challenge, resolution demand, withdraw and accept.

Figure 2 Safety argument reviewing process (see online version for colours)



Each party (i.e., the reviewer and proposer) maintains a commitment store. The following commitment rules are envisaged applicable to both participants:

1 Initial commitment, CR_0: Initially, the commitment stores of both participants are empty.

2 Acceptance, CR_A: Accepting an argument P causes P being added to the speaker's commitment store.

3 Withdrawals, CR_W: Withdrawing an argument P causes P being removed from the speaker's commitment store if it is there.

4 Arguments, CR_S: Making an argument P results in P being added to the speaker's commitment store, unless the preceding event is a challenge, in which case it will be handled according to the defence rule below.

5 Defence, CR_Y: If an argument P is made as an answer for a challenge of Q, then P and P implies Q are added to the speaker's commitment store.

Further, both parties must follow the dialogue rules listed below:

1 R_{FROM}: A participant is allowed to make multiple moves in one turn, until he/she voluntarily pass the initiative.

2 $R_{REPSTAT}$: A speaker may not make an argument if he/she has already committed to it except for answering a question or a challenge.

3 R_{ACCEPT}: A speaker is not allowed to accept an argument P unless:
   a his opponent is committed to P
   b the preceding event is a question from his opponent regarding P.

4 R_{CHALL}: A challenge P must be responded by:
   a withdrawing P
   b making an argument supporting P.

5 R_{QUEST}: A question must be responded by:
   a providing an answer
   b Yes/No answer for a bipolar question.

6 $R_{RESOLUTION}$: A resolution demand must be responded by:
   a withdrawing one of the offending conjuncts
   b making an argument justifying the inconsistent situation.

7 $R_{LEGALCHALL}$: A challenge may not be used unless P is:
   a in the opponent's commitment store
   b not in the challenger's commitment store.

This, then, is the dialectical model we have proposed for safety argument review. Next, the appropriateness of the proposed dialogue model needs to be established. To enable human participants (e.g., safety engineers and assessor) to operationalise such dialogue model, computer support is required, e.g., to properly record the interaction history and commitments as assurance evidence. The proposed experimental work required for this, aimed at iteratively building a computational realisation of the model and establishing whether the model can be readily to provide good service to the safety reviewing process. A fully functional prototype operationalising the proposed model has been built to facilitate our initial usability evaluation of the implementation and the dialogue model. Four safety engineering and two human-computer interaction experts from University of York participated in the evaluations. Details of the evaluations are documented in Wan (2010). The evaluations at this stage do not reveal problems related to the model though it provides two valuable suggestions for the computational implementation of the model, e.g., the evaluators are in favour of seeing the visual links between dialogue contributions; the evaluators also prefer more assistants in making a move.

5 Conclusions

The argument-based approach to computer system safety engineering has been introduced. The approach has been widely adopted in Europe, and increasingly world-side (e.g., Australia and Japan) particularly in safety case development (cf. Haddon-CAVE QC, 2009). The argument and safety case approach to software safety certification is being adopted in a wide number of domains (including defence, automotive, medical, and rail). We have proposed a number of argument schemes in computer system safety domain. The proposed schemes are useful for computer system engineers to construct and other stakeholders to review and evaluate safety arguments. We have also argued the usefulness of dialectical models in facilitating safety arguments review and discussed our work in seeking such a dialogue model.

There are several ways to carry this work forward. Our immediate work is to conduct a refinement of computational implementation of our model and using which to conduct further user studies. We are also planning to provide users with some forms of support in making strategic moves during the arguments reviewing process. A desirable means is to provide users with a software agent who can offer strategic advice when required. A pre-requisite for such an agent is a set of appropriate strategic heuristics. The strategic knowledge is essential for an agent to provide useful advice. There has been considerable

research into the development of suitable computational strategies [c.f. a review in Yuan et al. (2011)]. It is, however, necessary to develop strategies against the specific requirements for a software arguing assistant. To determine the appropriateness of the proposed strategies, further studies will be required, aimed at testing whether the strategy is readily to be adopted by a software agent to provide valuable strategic advice. Regarding the argument schemes, as well as the ten proposed schemes, many other argument schemes can be proposed, e.g., *argument from component reuse*, *argument from developers' experience*. We are currently catering for such a repository of schemes and investigating means to incorporate schemes into our computational tool for use in safety arguments construction and review.

Much remains to be done, but the potential pay-off in terms of expanding computer system safety engineering is enormous. Further, there is great scope for an interesting and fruitful interplay between research within informal logic on the argument schemes and the dialogue models per se, and research on their utilisation in computer system safety engineering. The hope is that this paper will move this interplay forward.

Acknowledgements

The author wishes to convey sincere appreciation to Dr. David Moore from Leeds Metropolitan University for his valuable comments on the early draft of this paper and to Dr. Katrina Attwood from University of York for her time to discuss the initial idea of this paper.

References

Bench-Capon, T.J.M. and Dunne, P.E. (2007) 'Argumentation in artificial intelligence', *Artificial Intelligence*, Vol. 171, Nos. 10–15, pp.619–641.

Bishop, P.G. and Bloomfield, R.E. (1998) 'A methodology for safety case development', *Safety critical Systems Symposium (SSS 98)*, Birmingham, UK.

Bloomfield, R.E. and Littlewood, B. (2003) 'Multi-legged arguments: the impact of diversity upon confidence in dependability arguments', *Proc. of Dependable Systems and Networks (DSN)*, pp.25–34, IEEE Computer Society.

Dijkstra, E.W. (1972) 'Chapter I: notes on structured programming', in Dahl, O.J., Dijkstra, E.W. and Hoare, C.A.R. (Eds.): *Structured Programming*, Academic Press Ltd., London.

Greenwell, W.S., Holloway, C.M. and Knight, J.C. (2005) 'A taxonomy of fallacies in system safety arguments', *Proc. of the International Conference on Dependable Systems and Networks*, Yokohama, Japan.

Haddon-CAVE QC, C. (2009) *The Nimrod Review – An Independent Review into the Broader Issues Surrounding the Loss of the RAF Nimrod MR2 Aircraft XV230 in Afghanistan in 2006*, Printed in the UK by the Stationery Office Limited.

Hamblin, C. (1970) *Fallacies*, Methuen, London.

Harrison, K.J. (1999) 'Static code analysis on the C-130J Hercules Safety-Critical Software', *Proc. of the 17th International Systems Safety Conference*, Orlando, Florida.

Hawkins, R. and Kelly, T. (2010) 'A structured approach to selecting and justifying software safety evidence, *Proc. of the 5th IET International System Safety Conference*, Manchester, UK.

International Electrotechnical Commission (1998) *Functional Safety of Electrical/Electronic/Programmable Electronic Safety-related Systems (IEC 61508)*, available at http://www.iec.ch/ (accessed on 20 May 2011).

Kelly, T.P. (1999) 'Arguing safety – a systematic approach to safety case management', DPhil Thesis, Department of Computer Science, University of York.

Kelly, T.P. (2007) 'Reviewing assurance arguments – a step-by-step approach', *Proc. of Workshop on Assurance Cases for Security – The Metrics Challenge, Dependable Systems and Networks (DSN)*, Edinburgh, UK.

Kelly, T.P. (2008) 'Are safety cases working?', *UK Safety Critical Systems Club Newsletter*, Vol. 17, No. 2, pp.31–33.

Kelly, T.P. and McDermid, J.A. (1998) 'Safety case patterns – reusing successful arguments', *Proc. of IEE Colloquium on Understanding Patterns and Their Application to System Engineering*, London, UK.

Littlewood, B. and Wright, D. (2007) 'The use of multi-legged arguments to increase confidence in safety claims for software-based systems: a study based on a BBN of an idealised example', *IEEE Trans Software Engineering*, Vol. 33, No. 5, pp.347–365.

Mackenzie, J.D. (1990) 'Four dialogue systems', *Studia Logica: An International Journal for Symbolic Logic*, Vol. 49, No. 4, pp.567–583.

McDermid, J.A. (2001) 'Software safety: where's the evidence?', *Proc. of the 6th Australian Workshop on Industrial Experience with Safety Systems and Software*, Australian Computer Society.

Perelman, C. and Olbrechts-Tyteca, L. (1969) *The New Rhetoric: A Treatise on Argumentation*, Notre Dame Press, University of Notre Dame, Indiana.

Rahwan, I. and McBurney, P. (2007) 'Argumentation technology: introduction to the special issue,' *IEEE Intelligent Systems*, Vol. 22, No. 6, pp.21–23.

Reed, C. and Grasso, F. (2007) 'Recent advances in computational models of argument', *International Journal of Intelligent Systems*, Vol. 22, No. 1, pp.1–15.

RTCA Inc. and EUROCAE (1992) *DO-178B: Software Considerations in Airborne Systems and Equipment Certification*, Washington, D.C.

Sommerville, I. (2007) *Software Engineering*, Pearson Education, Harlow.

UK Ministry of Defence (1997) 'Defence standard 00-55-the procurement of safety critical software in defence equipment', available at http://www.dstan.mod.uk/ (accessed on 20 May 2011).

UK Ministry of Defence (2004) 'Defence Standard 00-56 (Issue 3), Safety Management Requirements for Defence Systems, December, available at http://www.dstan.mod.uk/ (accessed on 20 May 2011).

Walton, D. (1996) *Argumentation Schemes for Presumptive Reasoning*, Erlbaum, Mahwah, NJ.

Walton, D. (1997) *Appeal to Expert Opinion*, Penn State Press, University Park, PA.

Walton, D. (1998) *The New Dialectic: Conversational Contexts of Argument*, University of Toronto Press, Toronto.

Walton, D. (2009) 'Argument visualization tools for corroborative evidence', *Proc. of the 2nd International Conference on Evidence Law and Forensic Science*, pp.32–49, Beijing.

Walton, D. and Krabbe, E. (1995) *Commitment in Dialogue: Basic Concept of Interpersonal Reasoning*, State University of New York Press, Albany NY.

Walton, D., Reed, C. and Macagno, F. (2008) *Argumentation Schemes*, Cambridge University Press, Cambridge.

Wan, F. (2010) 'Computer-assisted argument review, a dialectics approach', Master dissertation, University of York.

Weaver, R.A., Fenn, J. and Kelly, T.P. (2003) 'A pragmatic approach to reasoning about the assurance of safety arguments', *Proc. of 8th Australian Workshop on Safety Critical Systems and Software (SCS'03)*, Canberra, Australia.

Weaver, R.A., McDermid, J. and Kelly, T.P. (2002) 'Software safety arguments: towards a systematic categorisation of evidence', *Proc. of the 20th International System Safety Conference (ISSC2002)*, System Safety Society, Denver, Colorado, USA.

Yuan, T., Moore, D. and Grierson, A. (2007) 'A human computer debating system and its dialogue strategies', *International Journal of Intelligent Systems, Special Issue on Computational Models of Natural Argument*, Vol. 22, No. 1, pp.133–156.

Yuan, T., Moore, D. and Grierson, A. (2008) 'A human-computer dialogue system for educational debate, a computational dialectics approach', *International Journal of Artificial Intelligence in Education*, Vol. 18, No. 1, pp.3–26.

Yuan, T., Moore, D., Reed, C., Ravenscroft, A. and Maudet, N. (2011) 'Informal logic dialogue games in human-computer dialogue', *Knowledge Engineering Review*, Vol. 26, No. 3, pp.159–174.