Argument Schemes in Computer System Safety Engineering

TANGMING YUAN

Department of Computer Science
University of York
Deramore Lane
York YO10 5GH
UK
Email: tommy.yuan@cs.york.ac.uk

TIM KELLY

Department of Computer Science
University of York
Deramore Lane
York YO10 5GH
UK
Email: tim.kelly@cs.york.ac.uk

Abstract: In this paper, we argue that informal logic argument schemes have important roles to play in safety argument construction and reviewing process. Ten commonly used reasoning schemes in computer system safety domain are proposed. The role of informal logic dialogue games in system safety arguments reviewing is also discussed and our intended work in this area is proposed. It is anticipated that this work will contribute toward the development of safety arguments and help to move forward the interplay between research in informal logic and research in computer system safety engineering.

Resumé: Dans cet article nous soutenons que les schèmes de la logique non formelle jouent des rôles importants dans la construction d'arguments employés dans des systèmes de sécurité en informatique. Nous décrivons dix de ces schèmes de raisonnement qui sont couramment utilisés. Nous discutons du rôle des jeux de dialogue en logique non formelle joué dans ces systèmes, et de nos travaux prévus dans ce domaine. Nous anticipons que ces travaux contribueront à l'élaboration de ces types d'arguments et à l'interaction entre la recherche dans la logique non formelle et la recherche dans les systèmes de sécurité en informatique.

Keywords: argument schemes, dialectics, safety arguments, safety arguments reviewing

 Informal Logic, Vol. 31, No. 2 (2011), pp. 89-109

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 *off line* 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. The particular concern of this paper is the safety aspect of a computer system.

Unfortunately, the assessment of safety 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 safety 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, for example, IEC 61508 (International Electro-technical 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 safety. 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 achieve and demonstrate computer system safety. We firstly give a brief introduction of argument-based approach to computer system safety engineering. We then argue that 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 facilitating safety arguments reviewing process and our proposed work in this area.

2. Safety arguments

A recent approach for both achieving and demonstrating computer system safety is to adopt an argument-based approach (e.g. McDermid, 2001; Bishop and Bloomfield, 1998;

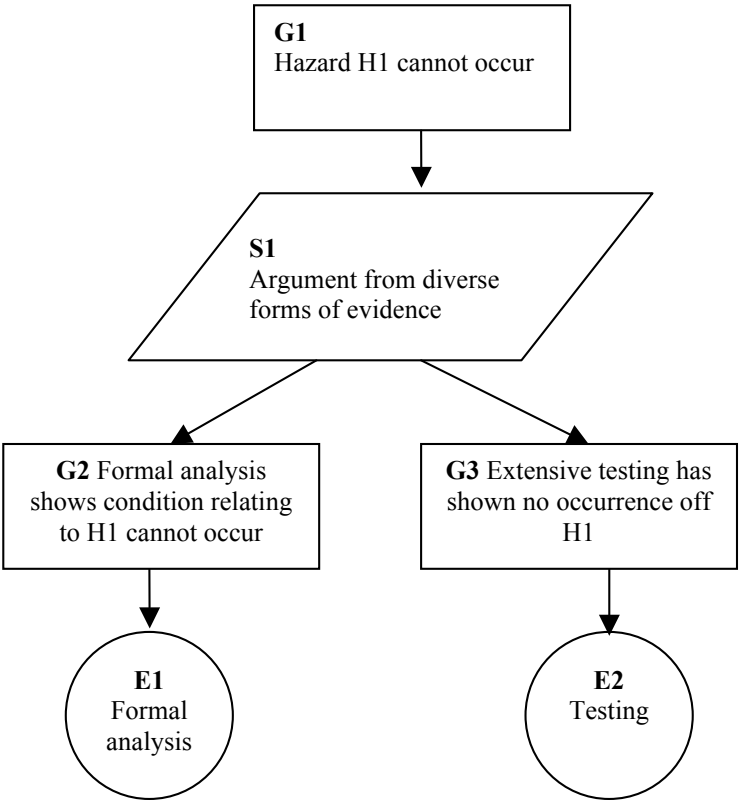

Figure 1 Example Use of Goal Structuring Notation

Kelly, 2007), aka *goal-based* approach. In an argument-based approach, the stakeholders first agree on the goals for which assurance is required (for example, this device must not harm people), then the developers produce specific claims (for example, the radiation delivered by this device will never exceed intensity level x) and an argument to justify the claims based on verifiable evidence (for example, 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*.

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. A traditional approach to assess argument is the *fallacy* approach, where arguments were categorized in terms of traditional

fallacies. A modern approach to argument assessment is the *scheme* 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: i) 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; ii) when assessing argument, each argument scheme provides a set of *critical questions* that can be used to examine the plausibility of an argument. The set of critical questions 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 (Walton, 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 critical questions (CQ) 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?

Both *fallacy* and *scheme* are useful approaches to evaluate arguments. The fallacy approach is a post activity to identify argument flaws. Argument schemes however can be used at both early argument construction stage and at later on review stage. Fallacy approach therefore focuses on detecting argument flaws while scheme approach focuses on preventing argument flaws (via scheme template) and revealing argument deficits (via

critical questions). Further, the *fallacy* approach suffers from the fact that many instances of traditional fallacies appear to be good arguments (cf. Walton et al., 2008). In the paper that follows, we focus on scheme approach to safety argument development. We are particularly interested in forms of reasoning in computer system safety domain. Informal logic, by no means, provides all domain specific schemes for computer system safety, 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 critical questions 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).

The scheme can be applied when there is knowledge of plausible hazards, e.g. identified by hazard analysis. To illustrate the scheme in action, we use the portable insulin pump system presented in (Sommerville 2007, p.196). The system is concerned with reading the blood sugar (glucose) sensor, computing the insulin requirements and controlling the micro pump which causes the insulin to be delivered automatically. Hazard analysis of the system reveals the following eight hazards: (1) insulin overdose, (2) insulin underdose, (3) power failure due to exhausted battery, (4) electrical interference with other medical equipment, (5) parts of machine break off in body, (6) infection caused by introduction of machine, (7) poor sensor and actuator contact, and (8) Allergic reaction to materials or insulin. To demonstrate that the insulin pump is acceptable safe to operate, a plausible strategy is to apply the scheme of *argument from hazard avoidance* and the argument instance will be as follows.

Premise1: Hazard 1 has been addressed.
Premise2: Hazard 2 has been addressed.
...
Premise 8: Hazard 8 has been addressed.
Conclusion: The insulin pump is acceptable safe to operate.

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 between 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.

The scheme can be applied in the context where the set of safety related functions for system *X* is known, e.g. identified by functional hazard analysis. An example instance argument applying the scheme is reproduced from (Kelly, 1999, p. 182) below.

Premise 1: Fuel management system operates safely.
Premise 2: Thrust reverser function operates safely.
Premise 3: Airframe communications function operates safely.
Premise 4: There are no hazardous interactions between functions.
Conclusion: Engine controller operates safely.

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 fault tree analysis* is proposed as follows.

Premise 1: Probability fault tree analysis 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 data sets.

The scheme can be applied wherever a fault tree for the condition exists, i.e. the skills for the construction and validation such a causal model are available. An example instantiation of the scheme is taken from (Kelly, 1999, p.254) as follow.

Premise 1: Probability fault tree analysis estimates probability of failure on demand to be 0.13×10^{-3} per annum.
Conclusion: probability of failure on demand is less than 0.001 per annum.

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 fourteen 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} to10^{-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 critical questions associated with this scheme that could be used to validate the strength of the argument. An example criticism made by Haddon-CAVE QC (2009) matching each critical question 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 critical questions be asked concerning the validity of the argument, H73 might not be improperly sentenced and the accident might be avoided.

The scheme can be applied when the source of past data exists and is accessible.

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 critical question, argument from authority (e.g. appeal to the analysers' experience) might be used to support the proof claim.

The scheme can be applied in the context of mathematical specification of the system and the availability of expertise and resources for formal verification. An example instance of the scheme of *argument from formal verification* drawn from the

insulin pump system described in section 3.1 above is given below.

Premise: Formal verification shows condition of exceeding *maxDose* will not occur**.**
Conclusion: The maximum single dose computed by the insulin pump will not exceed *maxDose***.**

3.6 Argument from verification testing

Verification testing aims to provide evidence that a program 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.

Verification testing is probably the most commonly used evidence as to whether the quality of a system matches its specifications. An example instance of the scheme of *argument from verification testing* drawn from the insulin pump system described in section 3.1 above is given below:

Premise: 400 verification tests show no single occurrence of exceeding *maxDose*.
Conclusion: The maximum single dose computed by the insulin pump will not exceed *maxDose*.

3.7 Argument from validation testing

Validation testing checks how the program 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 systems, 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.

The scheme can be applied in the context of the existence of an operational profile for the system. An example instance of the scheme of *argument from validation testing* drawn from the insulin pump system described in section 3.1 above is given below.

Premise: 1000 person/hour validation tests show the insulin pump operates safely.
Conclusion: The insulin pump is safe to operate.

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 program 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 program 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.

The scheme can be applied wherever there are at least two legs of argument supporting the same claim independently.

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).

The scheme can be applied in context of designed decisions to use redundant components in system X. An example instance of the scheme of *argument from redundancy* drawn from the insulin pump system described in section 3.1 above is given below.

Premise: Redundant batteries are used to ensure reliability.
Conclusion: The probability of power failure due to exhausted battery is low.

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?

The scheme can be applied in situations when standard processes are used. An example instance of the scheme of *argument from development process* drawn from the insulin pump system described in section 3.1 above is given below:

Premise: ASTM D1585 - Guide for Integrity Testing of Porous Medical Packages is used to carry out all the tests for insulin pump system.
Conclusion: All the tests of the insulin pump system are properly carried out.

These, then, are the commonly used argument schemes we have proposed for a computer system safety engineering domain. Many other schemes can also be proposed, e.g. *argument from component reuse*, *argument from developers' experience*. We are currently catering for such a repository of schemes.

Although the argument schemes proposed above are generalised from computer system safety engineering literature,

they may be applicable to non-computerised systems, e.g. the scheme of argument from hazard avoidance, argument from probabilistic fault tree analysis, argument from functional decomposition, argument from historical data and argument from redundancy. As argument schemes are patterns of *good* arguments, they can be used to instantiate safety arguments. Further, because critical questions associated with each scheme indicate implicit assumptions of an argument where points of attacks can be made, they are therefore the key tools for reviewing and evaluating the strength and validity of safety arguments. It is anticipated that such schemes will provide guidelines for computer safety engineers to represent and review 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). For a successful safety arguments reviewing, it is required 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 (e.g. 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. Next, the appropriateness of any 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.

It might also be necessary 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. Our current work caters for these concerns.

5. Conclusion

The argument-based approach to computer system safety engineering has been introduced. The approach has been widely ad-

opted in Europe, and increasingly world-side (e.g. Australia, 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 and provided concrete example uses of the proposed schemes in representing safety arguments. The critical questions of the proposed schemes provide crucial tools for computer system engineers 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 planned work in this area.

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 authors wish 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. & Dunne P. E., Argumentation in Artificial Intelligence. *Artificial Intelligence*, 171, pp. 619-641, 2007.

Bishop P.G. & Bloomfield, R.E, A Methodology for Safety Case Development. *Safety-critical Systems Symposium* (SSS 98), Birmingham, UK, 1998.

Bloomfield, R.E. & Littlewood, B., 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, 2003.

Dijkstra, E.W., Notes on Structured Programming. In Dahl, Dijkstra, Hoare, *Structured Programming*, Academic Press, 1972.

Greenwell, W.S., Holloway, C.M. & Knight, J.C., A Taxonomy of Fallacies in System Safety Arguments. *Proc. of the*

International Conference on Dependable Systems and Networks, Yokohama, Japan, 2005.

Haddon-CAVE QC, C., *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, 2009.

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

Harrison, K.J., Static Code analysis on the C-130J Hercules Safety-Critical Software. *Proc. of International Systems Safety Conference*, 1999.

International Electro-technical Commission, *Functional Safety of Electrical/ Electronic/ Programmable Electronic Safety-related Systems* (IEC 61508), 1998. (http://www.iec.org.ch)

Kelly, T.P., 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, 2007.

Kelly, T.P., *Arguing Safety – A Systematic Approach to Safety Case Management.* DPhil Thesis, Department of Computer Science, University of York, 1999.

Kelly, T.P, Are Safety Cases Working? *U.K. Safety Critical Systems Club Newsletter*, 17 (2): 31-33, 2008.

Littlewood, B. & Wright, D., 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*, 33 (5), pp. 347-365, 2007.

Mackenzie, J.D., Four Dialogue Systems. *Studia Logica: An International Journal for Symbolic Logic,* 49 (4), pp. 567-583, 1990.

McDermid, J. A., Software Safety: Where's the Evidence? Proc. of the 6th Australian Workshop on Industrial Experience with Safety Systems and Software, Australian Computer Society, 2001.

Perelman, C. & Olbrechts-Tyteca, L., *The New Rhetoric: a Treatise on Argumentation*. Notre Dame Press, 1969.

Rahwan, I. & McBurney, P., Argumentation Technology: Introduction to the Special Issue. *IEEE Intelligent Systems*, 22 (6), pp. 21-23, 2007

Reed, C. & Grasso, F., Recent Advances in Computational Models of Argument. *International Journal of Intelligent Systems*, 22(1), pp. 1-15, 2007.

RTCA Inc. and EUROCAE. *DO-178B: Software considerations in airborne systems and equipment certification*, 1992.

Sommerville, I., *Software Engineering*. Pearson Education, 2007.

UK Ministry of Defence, *Defence standard 00-55-the procurement of safety critical software in defence equipment*, 1997. (http://www.dstan.mod.uk/)

UK Ministry of Defence, *Defence Standard 00-56 (Issue 3), Safety Management Requirements for Defence Systems*, December 2004. (http://www.dstan.mod.uk/)

Weaver, R.A., Fenn, J. & Kelly, T.P., 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, 2003.

Weaver, R.A., McDermid, J. & Kelly, T.P., Software Safety Arguments: Towards a Systematic Categorisation of Evidence. *Proc. of the 20th International System Safety Conference (ISSC2002)*, Denver, Colorado, USA, System Safety Society, 2002.

Walton, D., *Argumentation Schemes for Presumptive Reasoning.* Mahwah, N.J., Erlbaum, 1996.

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

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

Walton, D., Argument Visualization Tools for Corroborative Evidence. *Proc. of the 2nd International Conference on Evidence Law and Forensic Science,* pp. 32-49, Beijing, 2009.

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

Walton, D., Reed, C. & Macagno, F., *Argumentation Schemes.* Cambridge University Press, 2008.

Yuan, T., Moore, D. and Grierson, A., A Human Computer Debating System and Its Dialogue Strategies. *International Journal of Intelligent Systems, Special Issue on Computational Models of Natural Argument*, 22 (1):133-156, 2007.

Yuan, T., Moore, D. and Grierson, A., A Human-Computer Dialogue System for Educational Debate, A Computational Dialectics Approach. *International Journal of Artificial Intelligence in Education* 18:3-26, 2008.

Yuan, T., Moore, D., Reed, C., Ravenscroft, A. and Maudet N., Informal Logic Dialogue Games in Human-Computer Dialogue. *Knowledge Engineering Review,* 26(3), 2011.