Argument and Computation, 2015
Vol. 6, No. 2, 130–148, http://dx.doi.org/10.1080/19462166.2014.927921

Computer-assisted safety argument review – a dialectics approach

Tangming Yuan[a]*, Tim Kelly[a] and Tianhua Xu[b]

[a]*Department of Computer Science, University of York, Heslington, York YO10 5GH, UK;* [b]*The State Key Laboratory of Rail Traffic Control and Safety, Beijing Jiaotong University, Beijing 100044, People's Republic of China*

(*Received 7 March 2013; accepted 21 May 2014*)

There has been increasing use of argument-based approaches in the development of safety-critical systems. Within this approach, a safety case plays a key role in the system development life cycle. The key components in a safety case are safety arguments, which are designated to demonstrate that the system is acceptably safe. Inappropriate reasoning in safety arguments could undermine a system's safety claims which in turn contribute to safety-related failures of the system. The review of safety arguments is therefore a crucial step in the development of safety-critical systems. Reviews are conducted using dialogues where elements of the argument and their relations are proposed and scrutinised. This paper investigates an approach of conducting argument review using dialectical models. After studying five established dialectical models with varying strengths and drawbacks, a new dialectical model specially designed to support persuasion and information-seeking dialogues has been proposed to suit the requirements of argument review. An argument review prototype system was then iteratively developed. It adopted the model and aims to conduct argument review dialogues in a structured manner. User-based evaluations of the system suggest the usefulness of the dialectics approach to safety argument review. The evaluation also sheds light on the future development of such an application.

Keywords: safety-critical systems; safety case; safety arguments; dialogue games; argument review

1. Introduction

Human society has become increasingly dependent on computer-based systems. As technology advances, microprocessors and software that run on them have found their way into the hearts of products which so many of us routinely use as part of our daily lives. The presence of microprocessor-based electronic control units in devices which people trust their lives to, such as the braking systems of cars and radiation therapy machines in hospitals, justifies the importance of safety as a first and foremost requirement in the engineering of these crucial systems. These systems are called *safety-critical systems* – any system where failure could result in loss of life, significant property damage, or damage to the environment. They are deployed in a wide range of sectors and industries, such as high-speed rail in the transport sector and nuclear power plants in the energy sector.

These systems have high *dependability* requirements. That is, they are frequently subjected to industrial, national, and international regulations which demand compliance to certain rules or procedures in their design, deployment, operation, and decommission process, the attainment of one or more minimum standards in areas such as security, reliability, availability, or safety. The construction of a safety case or functionally equivalent documentation is mandated

*Corresponding author. Email: tommy.yuan@york.ac.uk in many standards used to guide the development of software for safety-critical systems, such as DS 00-55 (UK Ministry of Defence [UK MoD], 1997) and Part 3 of IEC 61508 (International Electrotechnical Commission [IEC], 2010). A safety case is defined by Bishop and Bloomfield (1998) as: 'A documented body of evidence that provides a convincing and valid argument that a system is adequately safe for a given application in a given environment'.

There are at least three different approaches to constructing such a justification. The first is the product-based testing approach. The assessment of whether a system satisfies its dependability requirements, however, cannot be conclusively based on the testing of the final product, since product-based testing alone is not exhaustive and only proves the existence of faults and flaws, not the absence of them, as noted by Dijkstra (1972). The second is the process-based approach where safety justification tends to be *implicit* and *standards-based*, i.e. compliance to accepted practice was deemed to imply adequate safety (Bishop, Bloomfield, & Guerra, 2004). The process-based approach used by standards such as CMM, ISO 9001, and ISO/IEC 15504 describes frameworks of software engineering processes as a set of linked activities to create software (Wang & Bryant, 2002). Developers follow the software engineering process, and the documentation and other evidence they produce are evaluated by regulatory authorities against a capability scale or according to a specific capability determination method, to have the resulting software recognised for achieving the required levels of capabilities, such as safety integrity levels (SILs) or development assurance levels (DALs).

However, this process-based approach is problematic because it is not possible to demonstrate a direct causal relationship between the use of prescribed processes and high levels of safety. The evidence produced by the developers does not necessarily give a quantitative demonstration that the desired SIL or DAL has been achieved, and factors such as difficulty in detecting software failure in accidents and commercial sensitivity of failure data make it difficult to determine the software's operational levels of safety (Weaver, Fenn, & Kelly, 2003). So even if software is produced for a SIL or DAL process, it would be difficult to assess whether the required level of safety has been attained until it goes into operation. A further difficulty with the process-based approach is that although it may work well in a stable environment where best practice was supported by extensive experience, it cannot accommodate change and alternative strategies to achieve the same goal.

Over the past two decades, there has been a trend towards an explicit *argument-based* approach which is the third approach to safety justification (Bishop & Bloomfield, 1995, 1998; Kelly, 1999, 2007; Kelly & Weaver, 2004; McDermid, 1991, 2001; Yuan & Kelly, 2012). The approach is to support how sophisticated engineering arguments are actually made. For example, by assuring a safety argument within a safety case, this approach aims to demonstrate clearly how the safety requirements are fulfilled by the presented evidence, and thus derive confidence in the software's dependability. A key strength of this approach is to make the set of arguments explicit and able to be inspected, and this should increase confidence that the form of argument and its conclusion are both sound. Unlike the process-based approach, the argument-based approach is ready to accommodate fast-moving technologies, e.g. by accepting evidence not prescribed in the standards. Despite that there has not been much progress on developing practice in certification, there has been increasing discussion on the adequacy of argument-based approaches (Leveson, 2011; Wassyng, Maibaum, Lawford, & Bherer, 2011) and the adequacy of confidence (Hawkins, Kelly, Knight, & Graydon, 2011). The interest of this paper is the argument approach to the development of a safety case.

The rest of the paper is organised as follows. First, we briefly review the recent development in safety arguments in Section 2. We then argue for dialectics in safety argument review and propose requirements for a suitable safety argument review model (SARM) in Section 3. A safety argument review model that suits the requirements of argument review is proposed in Section 4. A system that operates the review model is constructed and discussed in Section 5. Section 6 contains the user- and expert-based evaluation of the system and the argument review model. We finally conclude the paper and discuss our planned future work.

2. Safety arguments

A safety argument is part of a safety case that combines a body of evidence, showing that the evidence is sufficient to demonstrate the claim that the system is acceptably safe. To effectively assure the safety claims of a system, designers have to be able to communicate their arguments, along with the evidence which supports them, to reviewing peers and approving authorities in a clear, concise, and efficient manner. In a simple way, arguments could be described in free text; however, this approach has major drawbacks such as the lax and varying structure in many natural languages as well as the difficulties in maintaining the quality of writing once the number of contributors increases. Not only do these factors make the reviewing process very inefficient but, more importantly, they make it difficult to ensure all the stakeholders understand an argument in the same way (Kelly & Weaver, 2004).

Graphical notations are often deployed to represent arguments in a more structured and transparent manner (e.g. Buckingham Shum, 2008; Gordon & Walton, 2006; Reed & Rowe, 2004). In the safety-critical domain, there are two established, commonly used notations – the Goal Structuring Notation (GSN) proposed by the University of York (Kelly, 1999; Kelly & Weaver, 2004) and the Claims Argument Evidence proposed by Adelard LLP (Emmet & Cleland, 2002). The GSN has been adopted by an increasing number of companies in safety-critical industries and government agencies, such as the London Underground and the UK MoD, as a standard presentation scheme for arguments within safety cases (cf. Object Management Group, 2010). For example, 75% of UK military aircraft have a safety case with safety arguments expressed in GSN. This paper will use GSN to represent arguments graphically unless stated otherwise.

The GSN uses standardised symbols to represent an argument:

- individual constituent elements (claims, evidence, and context)
- relationships between elements (e.g. how claims are supported by evidence).

In GSN, claims in an argument are shown as *Goals* (rectangles). They are often broken down into sub-goals further down a hierarchy. Alternatively they may be supported by evidence, presented in the GSN as *Solutions* (circles), and for an argument to be robust, all sub-goals must eventually be supported by solutions at the bottom. *Strategies* adopted (especially when breaking down goals) are shown in parallelograms, and they are related to argument schemes (Yuan & Kelly, 2011). *Contexts* in which goals are stated appear as bubbles resembling racetracks. If necessary, ovals are used in GSN to denote *Assumptions* and *Justifications*. They can be distinguished by an ‘A’ or ‘J’ at the lower right of the symbol. Two types of links are used to connect the constituent elements. A line with a solid arrowhead, representing a *Supported by* relation, declares an inferential or evidential relationship. Permitted connections are goal-to-goal, goal-to-strategy, goal-to-solution, strategy-to-goal. A line with a hollow arrowhead represents an *In context* of relation, which declares a contextual relationship. Permitted connections are goal-to-context, goal-to-assumption, goal-to-justification, strategy-to-context, strategy-to-assumption, and strategy-to-justification (Kelly, 1999; Kelly & Weaver, 2004). An example use of the key components of the GSN is shown in Figure 1. The argument shows that in order to achieve the $G1$ both legs of evidence are collected and the arguing strategy is *from diverse forms of evidence*. For complex systems with many arguments, modular approaches (Kelly, 2005) have been used to aid with argument abstraction and composition. There has been substantial experience of using graphical

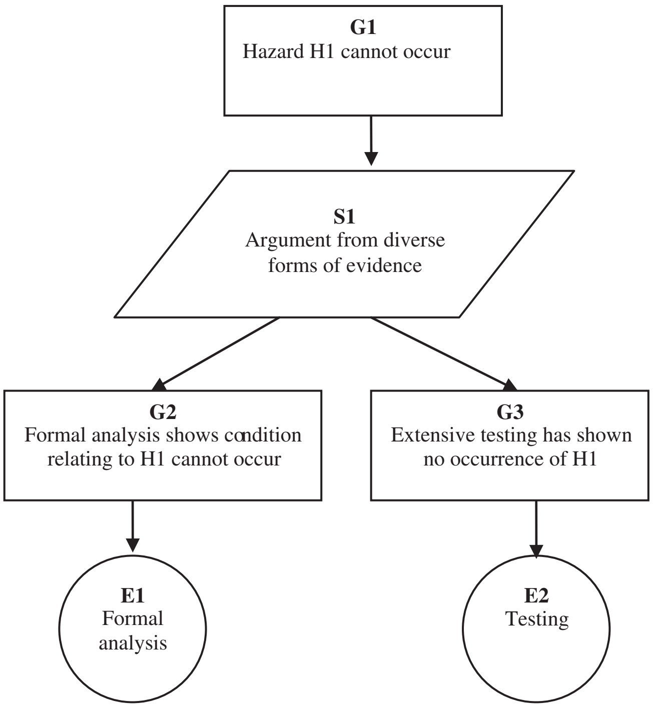

G1
Hazard H1 cannot occur
S1
Argument from diverse forms of evidence
G2
Formal analysis shows condition relating to H1 cannot occur
G3
Extensive testing has shown no occurrence of H1
E1
Formal analysis
E2
Testing

Figure 1. Example use of goal structuring notation.

GSN-based arguments for a wide range of scales of safety argument (from single programmable devices to whole aircraft safety cases).

3. Safety argument review

Arguments are by their nature subjective, and their robustness is not self-evident (e.g. confirmation bias (cf. Leveson, 2011)). To increase the soundness of the arguments, a review element is necessary for the assurance of safety cases. The process of documentation of arguments makes the case much more transparent and easier to review (Hawkins, Habli, Kelly, & McDermid, 2013). A review normally involves two parties: the proposing party, typically the system engineer, who asserts and defends the safety case, and the assessing party, e.g. an independent safety assessor who is or represents the certification authority, and scrutinises and attacks the arguments to uncover any vulnerability. The objective of a review is for the two parties to form a mutual acceptance of their subjective positions (Kelly, 2007).

Safety argument development and reviewing is a not a post-development activity, rather it should occur throughout the different stages of the system development life cycle. Typically at the end of the design stage, the certification authority must be convinced by the designer that the presented safety case is sound before giving approval to commence implementation. Another review by external, possibly regulatory authorities may also take place before the system is formally put into service. But internal reviews, performed by the peers or immediate superiors of the proposer, usually happen at an earlier stage with a higher frequency, not only to achieve a higher efficiency but also to lower the cost of correcting defects in a design by uncovering them earlier in the life cycle.

A successful safety argument review requires both someone to develop and defend the safety arguments and someone to challenge and critique the assumptions made (Kelly, 2008). The need for dialogue interactions has also been reinforced by the recent issue of Defence Standard 00-56 (UK MoD, 2004), 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 counter-evidence are an important part of a robust Safety Management System and should be documented in the Safety Case.

The importance of dialogue interactions in safety argument development seems clear. To understand how dialogues for multiple purposes can be conducted in a systematic, structured, and efficient manner, it is necessary to look at the *dialogue models* that have been developed in the area of dialectics within informal logic. Dialectics, as defined in the *Oxford English Dictionary*, is 'the art of critical examination into the truth of an opinion; the investigation of truth by discussion'. It can be seen as the branch of philosophy attempting to build models for 'good' dialogue. A dialectical system, sometimes called a dialogue game or dialectical model, is defined as 'a rule governed structure for organised conversation where there is an exchange of arguments between two parties reasoning together on a turn-taking basis aimed at a collective goal' (Walton, 1998). A dialectical system contains the following fundamental elements (cf. Hamblin, 1971; Walton & Krabbe, 1995):

- *A commitment store:* records the comments and commitments made by the participants during the dialogue;
- *A set of available move types:* specifies the moves permissible during a dialogue;
- *A set of dialogue rules:* prevents illegal events from happening during a dialogue;
- *A set of commitment rules:* specifies the effect of different move types on the participants' commitment store;
- *Winning and losing rules (optional).*

There are a few advantages for the adoption of dialectical models in argument review. For example, a dialectical model can help the identification and aid with the removal of fallacious arguments and common errors (Walton, 1998), and thus provide a platform for the argument proposer and reviewer to reason in a fair and reasonable manner. The set of move types provided by the dialectical model (e.g. question, challenge, and counter-argument) can help the argument reviewer to prepare for the type of criticisms that are likely to be made against an argument. Using a dialectical model can compensate to some extent for the formlessness of free review text by regulating the moves permitted in a dialogue, such that the orderliness of the dialogue depends less on the participants' language capabilities. It also improves accountability by providing facilities for recording dialogue history and commitments for assurance purposes. Finally, the rules set by dialectical systems could be enforced automatically using computers, so a computerised dialogue system could conduct organised dialogues with improved efficiency (Yuan, Moore, & Grierson, 2007).

A variety of dialectical models has been developed to suit dialogues serving different purposes under different circumstances. To determine which dialectical model is suitable for argument review, it is necessary to first understand which types of dialogues are present in argument reviews, and then consider a selection of models which are designed to support these dialogues. According to Walton & Krabbe (1995), dialogues between humans can be classified into six primary types, according to factors including the overall purpose of the dialogue, individual participants' objectives, and the information held by them initially. In practice, however, dialogues may contain attributes of multiple primary types. Generally speaking, the six primary types of dialogues are:

- *Information-seeking dialogues*, in which one participant seeks answers for one or more questions from another participant, who is presumed to have the required answers;
- *Inquiry dialogues*, in which participants, none of whom start off knowing the answers, collaborate to answer one or more questions;
- *Persuasion dialogues*, in which one participant seeks to persuade another to accept a statement which the latter is not currently endorsing;
- *Negotiation dialogues*, in which participants have a limited pool of resources to divide among themselves, and each seeks to maximise his or her share of resources according to their individual conflicting objectives;
- *Deliberation dialogues*, in which participants collaborate to decide what action or course of action is to be adopted in some situation they share or are willing to discuss whether they share a responsibility to decide the course of action; and finally
- *Eristic dialogues*, in which participants seek to vent perceived grievances, sometimes as a substitute for physical conflicts.

According to Krabbe (2000), persuasion, negotiation, and eristic dialogues are considered argumentative, whereas deliberation, inquiry, and information-seeking dialogues are considered non-argumentative, despite containing reasoning elements.

Knowing the general properties such as objectives of different dialogue types, it is then possible to try to identify the kinds of dialogues which are present in argument reviews by matching the characteristics of argument review to the qualities afforded by each category of dialogues. As stated in Section 3, the goal of the argument review is to have both the proposer and the reviewer come to a shared position on the argument's strength, by which the proposer tries to convince the reviewer that the presented argument is sound, while the reviewer may also try to convince the proposer to accept a counter-argument or the existence of certain vulnerabilities and limitations. Since both sides start off having conflicting opinions on the presented argument and try actively to persuade the other side, persuasion dialogues could be seen as instrumental during argument reviews, especially at the argument criticisms stage. Other types of dialogue, such as information-seeking and inquiry dialogues, could also take place since the reviewer may have a need to seek additional information regarding the argument from the proposer as well as a need to verify if what the argument conveyed wholly represents what the proposer is obliged to make known.

To effectively assist the conduct of dialogues between participants of argument review, a dialectical model therefore has to satisfy the requirements (cf. Yuan, Moore, & Grierson, 2003) listed below.

- *Support persuasion dialogues.* Persuasion dialogues seek to resolve the initial conflict in opinions between parties. This characteristic coincides well with argument review's goal of achieving mutual acceptance on the strength of an argument.
- *Support information-seeking dialogues*. This ensures that the parties can request and receive additional information from each other, especially when making attempts to deepen the understanding of presented arguments and clarify any implications in statements.
- *Asymmetric.* Asymmetric means the dialogue participants use different sets of moves and follow different sets of rules. During the argument construction and review process, the proposer and the reviewer have different roles where the proposer is to propose arguments and respond to criticisms and the reviewer is merely to criticise the arguments.
- *Sufficient room for opinion expression.* In an argument review, both the proposer and reviewer may have the need to present claims (including counter-claims), evidence of varying lengths

and complexity, or to address any matters of concern. The move types and the dialogue rules provided by the model should provide sufficient room for strategy formation and the expression of these views.

- *Support the identification and removal of fallacious arguments and common errors*. Errors and mistakes in reasoning affect the soundness of arguments resulting from it. Inappropriate reasoning in safety arguments could undermine a system's safety claims which in turn contribute to safety-related failures of the system. The model should have rules that support the identification and aid in the removal of fallacious arguments and common errors.
- *High usability, low cognitive load on user*. The model and its rules should be expressible in simple and concise language, so users familiar with that language can easily understand them and use them to assist their dialogues without first having to go through extensive dedicated training.

4. A model for safety argument review

After studying a sample of established dialectical models (Mackenzie, 1979; Prakken, 2005; Ravenscroft & Pilkington, 2000; Walton & Krabbe, 1995; Yuan, Moore, & Grierson, 2008) against the criteria specified in Section 3, we had a choice of selecting one of these five models wholesale as the underlying framework for an argument review dialogue, or to try and formulate a new model by combining selected elements from existing models which can justifiably contribute to the fulfilment of the requirements set in Section 3. Eventually the latter approach was taken. This is because each of the reviewed models has been perceived to contain one or more disadvantages, as shown in Table 1 (please note that the DC of (Mackenzie, 1979) and the DE of (Yuan et al., 2008) are arranged in the same column as they are of the same style), which damages their suitability for use in argument review or poses a knowledge barrier to participants without specialist training. Detailed critiques of these models can be found in Wan (2010). These shortcomings could not be compensated for by technical means and will substantially handicap the review process. On a side note, participants may also require additional time to familiarise themselves with longer or more complex rules of certain models.

The approach of building a new model by combining useful elements from existing models has the advantage of making use of the strengths of several established models in assisting the types of dialogue they were designed to support, while simultaneously making up the shortcomings they individually pose as an argument review framework. A safety argument review model, namely SARM, is proposed. The overall review process of SARM is represented in Figure 2. The process encompasses three distinct phases: initiation, review, and revision. 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

Table 1. Dialogue model assessment summary.

Models\ perceived disadvantages	Prakken	Walton and Krabbe PPD	Ravenscroft and Pilkington	Mackenzie DC and Yuan et al. DE
Insufficient information-seeking or inquiry capabilities required by argument review	✓	✓	✓	✓
Asymmetric	✓	✓		✓
Higher knowledge barrier due to specialised locutions in move types		✓		
Lower usability arising from length and complexity of rules		✓		✓

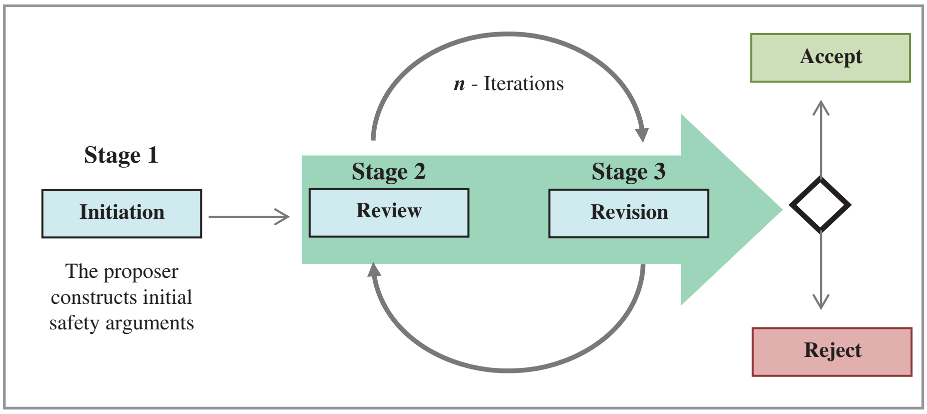


Figure 2. Safety argument review process.

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. The full description of the dialogue model is as follows.

The proposer maintains a commitment store as the safety case model. The safety case model is initially empty and only the proposer is allowed to add or remove elements from the model. SARM also maintains a review model which contains all the interaction sequences made by the reviewer and proposer alternately. Only the reviewer is allowed to add a new instance of interaction sequence to the model while both the reviewer and the proposer are allowed to view and expand an existing interaction sequence by appending fresh moves.

A participant is allowed to make one or more moves in every turn, until he or she voluntarily ends his or her turn and passes on the initiative. For the initiation stage, the proposer uses different forms of the *Argument Element* move type to construct the initial safety argument. These forms are in line with GSN syntax including goal (claim), strategy, evidence, context, assumption, and justification, which are elements of a safety argument. Following Bench-Capon (1998), each move type is expressed in an extended pre- and post-condition form as follows.

Claim (P)
Description: A GSN goal that presents the overall goal of the argument as well as the sub-goals that support the top goal. P is an atomic proposition.
Supported form: *I assert a claim P*.
GSN symbol: ▭
Precondition: Proposer has control of the dialogue.
Post-condition: Proposer is committed to P. If P is made as response to a review, the correspondent link *SupportedBy* is also added to the commitment store.
Permitted GSN connections: context, assumption, justification, strategy, claim, evidence.
Permitted review options: counter-argument, challenge, question, query-ref, comment.

Strategy (S)
Description: A GSN strategy that presents a brief description of the argument approach, e.g. argument from hazard avoidance (Yuan & Kelly, 2011).
Supported form: *I assert a strategy S*.

GSN symbol: ▱
Precondition: Proposer has control of the dialogue.
Post-condition: Proposer is committed to S.
Permitted GSN connections: claim, context, assumption, justification.
Permitted review options: challenge, question, query-ref, comment.

Evidence (E)
Description: A GSN solution presents a reference (stated as a non-phrase) to an evidence item or items that provide support for a particular claim.
Supported form: *I present evidence E.*
GSN symbol: ○
Precondition: Proposer has control of the dialogue.
Post-condition: Proposer is committed to E.
Permitted GSN connections: claim
Possible review options: challenge, question, query-ref, comment.

Context (C)
Description: presents a contextual artefact. This can be a non-phrase style of reference to contextual information, or a statement.
Supported form: *I define context P.*
GSN symbol: ▭
Precondition: Proposer has control of the dialogue.
Post-condition: Proposer is committed to A. If A is made in response to a review, the correspondent link *InContextOf* is also added to the commitment store.
Permitted GSN connections: claim and strategy.
Permitted review options: question, query-ref, and comment.

Assumption (A)
Description: A GSN assumption that presents an intentionally unsubstantiated statement.
Supported form: *I assume A.*
GSN symbol: ⬭A
Precondition: Proposer has control of the dialogue.
Post-condition: Proposer is committed to A. If A is made in response to a review, the correspondent link *InContextOf* is also added to the commitment store.
Permitted GSN connections: claim and strategy.
Permitted review options: question, query-ref, comment.

Justification (J)
Description: A GSN assumption that presents a statement of rationale for inclusion.
Supported form: *I justify J.*
GSN symbol: ⬭J
Precondition: Proposer has control of the dialogue.
Post-condition: Proposer is committed to J. If A is made in response to a review, the correspondent link *InContextOf* is also added to the commitment store.
Permitted GSN connections: claim, strategy.
Permitted review options: question, query-ref, comment.

SupportedBy (R)
Description: A GSN link that presents an inferential relationship between goals in an argument or evidential relationships between a goal and the evidence used to substantiate it.

Supported form: *I assert R*.
GSN Symbol: ⟶
Precondition: Proposer has control of the dialogue.
Post-condition: Proposer is committed to R.
Permitted GSN connections: goal-to-goal, goal-to-strategy, goal-to-solution, strategy-to-goal.
Permitted opponent responses: challenge, question, query-ref, comment.

InContextOf (R)
Description: A GSN link that declares a contextual relationship.
Supported form: *I declare R*.
GSN Symbol: —▷
Precondition: Proposer has control of the dialogue.
Post-condition: Proposer is committed to J.
Permitted GSN connections: goal-to-context, goal-to-assumption, goal-to-justification, strategy-to-context, strategy-to-assumption, and strategy-to-justification.
Permitted review options: challenge, question, query-ref, comment.

The following set of move types is designed for the review stage. These move types include counter-argument, resolution demand, challenge, question, query-ref, and comment. All the moves at this stage will be added to the review model. If the move is a reply to an existing interaction sequence, then the move is appended to the tail of the sequence; otherwise a new interaction sequence (e.g. containing the current move and the argument element the current move is replying to) is created and added to the review model.

Counter-argument (Q)
Description: An argument that is opposite to the one being proposed say P.
Supported form: I argue that Q
Visual representation: the symbol (⬚) attached to the argument being countered.
Precondition: Reviewer has control of the dialogue. P is part of the safety case model.
Post-condition: Move added to the review model.
Permitted proposer responses: withdrawal (P), a claim or justification that defends P.

Resolution demand (P and Q)
Description: Demand the opponent to resolve two or more of his commitments which the speaker perceives to be conflicting.
Supported form: Please resolve (P and Q).
Visual representation: the symbol ↕ attached to each of the conflicting argument elements.
Precondition: Reviewer has control of the dialogue. P and Q are part of the safety case model.
Post-condition: move added to the review model.
Permitted proposer responses: withdrawal (P) or withdrawal (Q), or making a claim that justifies the inconsistent situation.

Challenge (P)
Description: The speaker casts his strong doubt towards P explicitly.
Supported Form: *Why P?*
Visual representation: the symbol (!) attached to the argument element being challenged.
Precondition: Reviewer has control of the dialogue. P is part of the safety case model.
Post-condition: Move added to the review model.
Permitted opponent responses: justification (J), claim (R), withdraw (P).

Question (P)
Description: A bipolar question that casts doubt on particular point, or persuades the acceptance of a particular point.
Supported Form: Is it the case that P?, or Isn't it the case that P?
Visual representation: the symbol (?) attached to an argument element being questioned.
Precondition: P is part of the safety case model. Reviewer has control of the dialogue.
Post-condition: Move added to the review model.
Permitted opponent responses: claim (P), claim (not P), withdrawal (P).

Query-ref
Description: A non-binary question that seeks information, e.g. what? how?
Supported form: *Please clarify…*
Visual representation: the symbol (?) attached to an argument element being queried.
Precondition: Reviewer has control of the dialogue.
Post-condition: Move added to the review model.
Permitted proposer responses: claim, context, assumption.

Comment
Description: a remark expressing an opinion or reaction.
Supported form: *I issue a comment Q*
Visual representation: the symbol (*) attached to the element being commented.
Pre-condition: Reviewer has control of the dialogue.
Post-condition: Move added to the review model.
Permitted proposer responses: claim, strategy, justification, context, assumption, solution, comment.

At the end of each of the proposer's turns, all the elements in the review model must be responded to, and the syntax of the safety case model (that is a connected diagraph with each path ending at least one item of evidence) must be maintained. In addition to the set of move types for the initiation stage, a further move type-withdrawal is available for the revision stage. For each move at this stage, if the move is responding to an existing interaction sequence in the review model, then it is appended to the tail of the sequence. Otherwise the move does not affect the review model.

Withdrawal (P)

Description: Abandon the speaker's commitment to P or express a lack of knowledge about a subject P, of the speaker's own accord or as an answer to a Question, Challenge, or Resolution Demand.

Supported Form: *I withdraw P*, *I am not sure if P*, *I do not know about P*.

Precondition: Proposer has control of the dialogue.

Post-condition: P and its associated links are removed from the proposer's store if they are there. Move added to the review model if it is responding to an existing interaction sequence in the review model.

Most move types (i.e. claim, challenge, resolution demand, question, withdrawal) are drawn from the DC of (Mackenzie, 1979) and the DE of (Yuan et al., 2007, 2008). These two models share a relatively long development history, and the Yuan et al. DE model is an example of the DC model's latest evolution. The usage of an actively developed model has the advantage of enjoying an increased familiarity in the computing industry (Maudet & Moore, 2001; Yuan, Moore, Reed, Ravenscroft, & Maudet, 2011), so training time is reduced for system safety engineers who are potential users. Its disadvantages of lacking information-seeking abilities and complex rules can be overcome by introducing moves from other models, and automatic enforcement of the rules by the software can remove the user need to understand the rules beforehand. In order to meet the expressive adequacy criteria of a suitable model for safety argument review, four new move types, namely various forms of 'Argument Element', non-binary 'Question', 'Counterargument', and 'Comment' are introduced.

'Argument Element', as the replacement of the original DC and DE 'Statement' move, is used to reflect different elements in an GSN argument, i.e. goal (claim), strategy, evidence, context, assumption, and justification. The 'Question' move type from the original DC and DE models was one of the largest obstacles in information-seeking, as only yes–no questions are allowed. In the proposed model, this has been expanded to include query-ref questions which are more open-ended, as well as binary questions. These information-seeking moves have been introduced from Ravenscroft and Pilkington's (2000) Model. The 'counter-argument' move, though simply being an argument, is specially designed for the reviewer to make an argument with an opposite conclusion. A similar definition is made in (Prakken, 2000). The 'comment' move can be used by the reviewer to make a remark that does not associate with any of the existing move types. The 'comment' move was strongly suggested by the participants of the user evaluations (see Section 6).

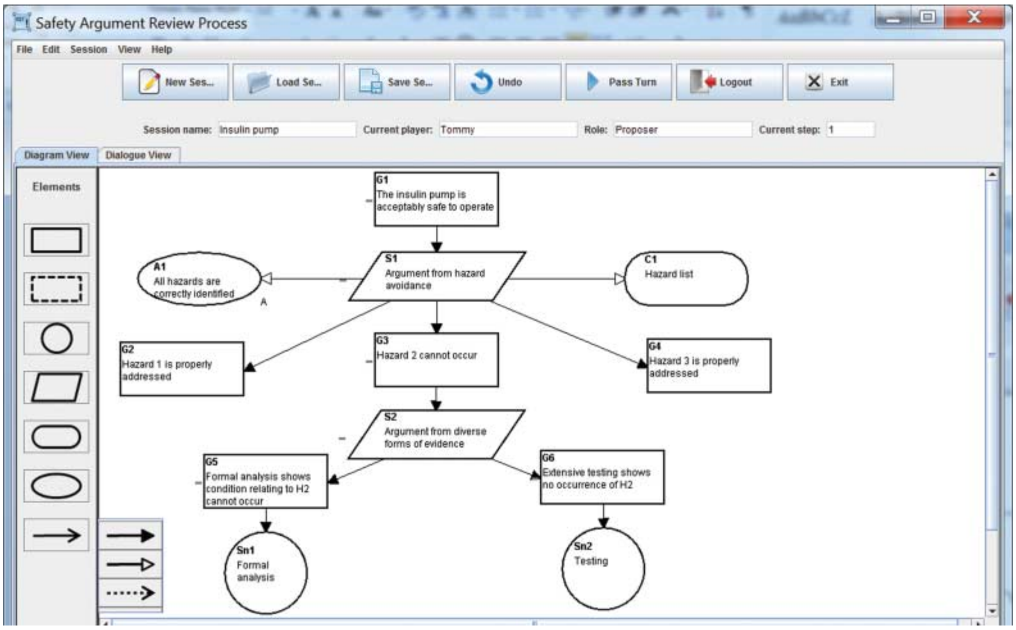
The asymmetric nature of the game where participants use different sets of moves and the commitment arrangement where only one commitment store is used for the safety case model is similar to that of Ravenscroft and Pilkington's (2000) model. A review model is introduced to manage all instances of interaction sequences. Alongside the set of GSNs, a set of review notations have been newly designed to visually represent the reviews.

SARM has a number of desirable properties for safety argument review. First, the model incorporates an influential safety argument notation, GSN, which makes it appealing to the end users. Second, the design of visual representation of review move types makes the reviews easy to manage as the moves made by the reviewer are clearly displayed on the safety case model and they will disappear when they have been responded to. Third, the inclusion of a review model which contains all the interaction sequences makes the review process traceable where participants can trace back each thread of discussion. Traceable interactions are important for assurance purposes. Fourth, the model contains a wide range of unique move types that enable the participants to express their views adequately. And finally, the simplicity of the rules poses a light cognitive load to the end users especially in a computational environment (see Section 6).

5. A system for safety argument review

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 assessors) to operationalise such a 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, aims at iteratively building a computational realisation of the model and establishing whether the model can readily be used to provide good service to the safety argument review process. A fully functional system, namely, Dialogue-based System Argument Review (DiaSAR), operationalising the proposed model, has been iteratively built at the University of York (Djaelangkara, 2012; Mazurek, Gerasimou, Madan, & Setivarahalli, 2011; Wan, 2010).

The current version of the system has a graphical user-interface that supports multiple user access and a backend database storing the user profiles and the review sessions. There are three



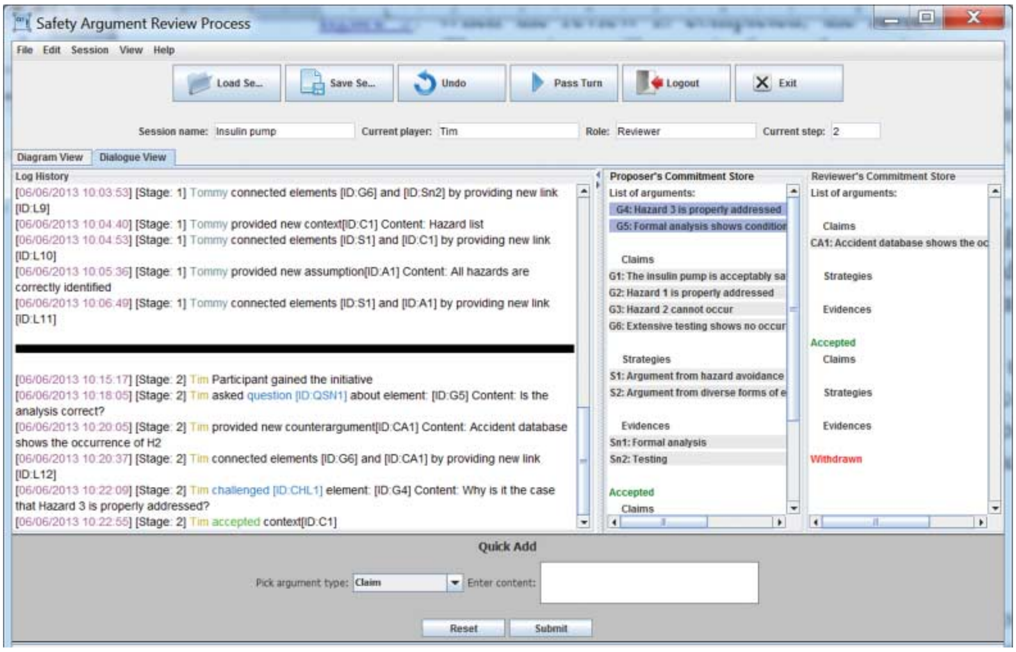
Session name: Insulin pump
Current player: Tommy
Role: Proposer
Current step: 1

Safety Argument Review Process
File Edit Session View Help
New Ses...
Load Se...
Save Se...
Undo
Pass Turn
Logout
Exit
Diagram View
Dialogue View
Elements
G1 The insulin pump is acceptably safe to operate
S1 Argument from hazard avoidance
A1 All hazards are correctly identified
A
C1 Hazard list
G2 Hazard 1 is properly addressed
G3 Hazard 2 cannot occur
G4 Hazard 3 is properly addressed
S2 Argument from diverse forms of evidence
G5 Formal analysis shows condition relating to H2 cannot occur
G6 Extensive testing shows no occurrence of H2
Sn1 Formal analysis
Sn2 Testing

Figure 3. An example user-interface for the proposer.

types of users of the system: system administrator, argument proposer, and reviewer. The system administrator manages all the users of the system and the review sessions. The argument proposers propose and subsequently defend their arguments. The reviewers criticise the arguments made by the proposer. An example system interface for the proposer can be seen in Figure 3. A proposer can create a new review session which specifies a session name, a proposer, and a reviewer of the session. The interface displays the current status of the session, e.g. the current player, his/her role, and the current step of the review. A proposer can use provided tools (i.e. claim, strategy, evidence, context, assumption, and links) to construct safety arguments following GSN syntax as described in Section 2. There are two views of the arguments: the diagram view as shown in Figure 3 and the dialogue view as shown in Figure 4 which records the dialogue history, commitment stores, and provides a text-based input. The two views are synchronised and designed using the model-view control architecture. It is up to the user to decide which view to interact with. A session can be saved and loaded for further editing. Once it has been done, the turn can be passed to the reviewer and a notification email will be sent to the reviewer. The session will then transit from the proposing state to the reviewing state.

A reviewer can log onto the system and load the sessions under review. By right clicking on any elements of the safety argument, a submenu will be available with items for the reviewer to accept, challenge, and question an argument element. A reviewer can also propose a counter-argument. Graphical notations have been developed for the set of the review tools alongside GSNs. Some of them can be seen in Figure 5. For example, a rectangle with dashed borderline represents a counter-argument and a dashed line with an open arrow represents an 'attacked by' relation (e.g. the argument 'Extensive testing shows no occurrence of H2' is attacked by argument 'Accident database shows the occurrence of H2'). A question mark represents a question (e.g. Is the analyst experienced?) and an exclamation mark represents a challenge (Why is it the case that hazard 3 is properly addressed?) made by the reviewer. Accepted elements are coloured green and withdrawn



Session name: Insulin pump Current player: Tim Role: Reviewer Current step: 2

Log History

```
[06/06/2013 10:03:53] [Stage: 1] Tommy connected elements [ID:G6] and [ID:Sn2] by providing new link
[ID:L9]
[06/06/2013 10:04:40] [Stage: 1] Tommy provided new context[ID:C1] Content: Hazard list
[06/06/2013 10:04:53] [Stage: 1] Tommy connected elements [ID:S1] and [ID:C1] by providing new link
[ID:L10]
[06/06/2013 10:05:36] [Stage: 1] Tommy provided new assumption[ID:A1] Content: All hazards are
correctly identified
[06/06/2013 10:06:49] [Stage: 1] Tommy connected elements [ID:S1] and [ID:A1] by providing new link
[ID:L11]

[06/06/2013 10:15:17] [Stage: 2] Tim Participant gained the initiative
[06/06/2013 10:18:05] [Stage: 2] Tim asked question [ID:QSN1] about element: [ID:G5] Content: Is the
analysis correct?
[06/06/2013 10:20:05] [Stage: 2] Tim provided new counterargument[ID:CA1] Content: Accident database
shows the occurrence of H2
[06/06/2013 10:20:37] [Stage: 2] Tim connected elements [ID:G6] and [ID:CA1] by providing new link
[ID:L12]
[06/06/2013 10:22:09] [Stage: 2] Tim challenged [ID:CHL1] element: [ID:G4] Content: Why is it the case
that Hazard 3 is properly addressed?
[06/06/2013 10:22:55] [Stage: 2] Tim accepted context[ID:C1]
```

Proposer's Commitment Store

List of arguments:
G4: Hazard 3 is properly addressed
G5: Formal analysis shows condition

Claims
G1: The insulin pump is acceptably sa
G2: Hazard 1 is properly addressed
G3: Hazard 2 cannot occur
G6: Extensive testing shows no occur

Strategies
S1: Argument from hazard avoidance
S2: Argument from diverse forms of e

Evidences
Sn1: Formal analysis
Sn2: Testing

Accepted
Claims

Reviewer's Commitment Store

List of arguments:

Claims
CA1: Accident database shows the oc

Strategies

Evidences

Accepted
Claims

Strategies

Evidences

Withdrawn

Pick argument type: Claim Enter content:

Figure 4. An example dialogue view of the system interface.

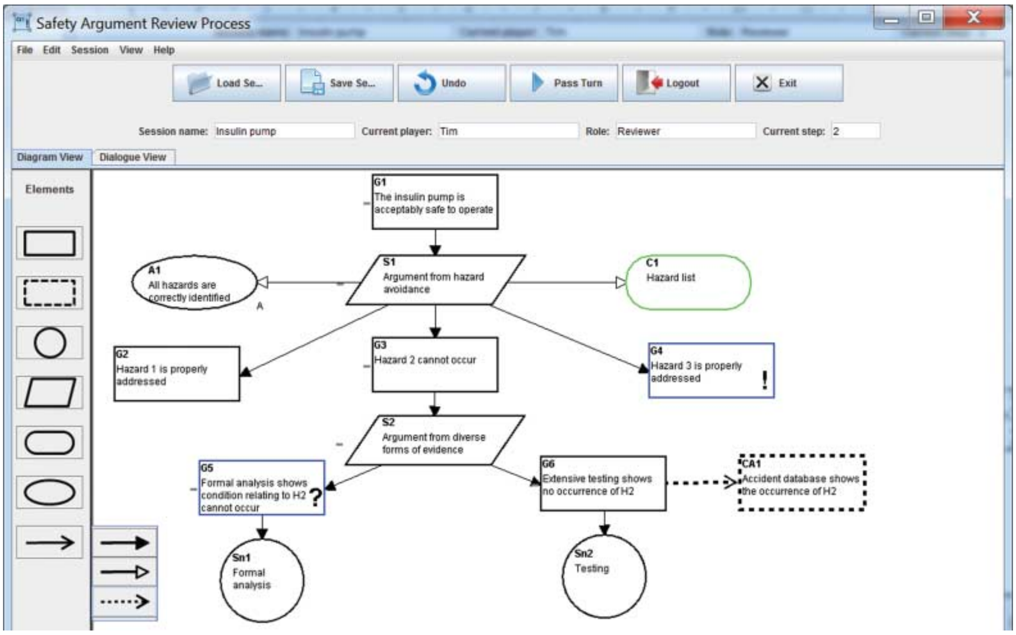
elements red. A short vertical line with open arrow at both ends is used to mark the situation where a resolution demand is made to a conflict among two or more elements.

Criticisms made to an argument element are recorded by the system as shown in Figure 4. When the review is completed, the reviewer can pass the turn back to the proposer. The session will transit from the review state to the revision state and a notification email will be sent to the proposer. Upon receiving the notification, the proposer can respond to the criticisms and revise the arguments following SARM rules, e.g. to respond to a challenge with a withdrawal, claim, evidence, or strategy. These rules are implemented as submenus attached to a review element where the proposer can select a suitable response. The responses, e.g. newly made claims, strategies, evidence, context, or assumptions, are automatically displayed as part of the safety argument, and the symbol of the review being responded to is removed from the user-interface. Once all the reviews have been responded to and other necessary revisions are done, it can be passed back to the reviewer for a second review. The process goes on until the safety argument has been fully accepted.

The system also provides functions to cater for the usability of the system. For example, the argument pane is resizable and scrollable according to the user's preference, the minus signs to some of the argument elements can be used to minimise all the elements beneath it, and the addition signs can be used to expand the view.

6. Evaluations

The system has been evaluated at different stages of its development. An initial design was drawn up, based on literature concerning safety argument review and (computational) dialectics, and interviews with two safety engineers at the University of York. This design was implemented as the first software prototype of the safety argument review software (version 1.0). This prototype then underwent a usability evaluation, with four safety engineers and two human–computer interaction


Session name: Insulin pump Current player: Tim Role: Reviewer Current step: 2

Safety Argument Review Process
File Edit Session View Help
Load Se...
Save Se...
Undo
Pass Turn
Logout
Exit
Diagram View
Dialogue View
Elements
G1 The insulin pump is acceptably safe to operate
S1 Argument from hazard avoidance
A1 All hazards are correctly identified
A
C1 Hazard list
G2 Hazard 1 is properly addressed
G3 Hazard 2 cannot occur
G4 Hazard 3 is properly addressed
!
S2 Argument from diverse forms of evidence
G5 Formal analysis shows condition relating to H2 cannot occur
?
G6 Extensive testing shows no occurrence of H2
CA1 Accident database shows the occurrence of H2
Sn1 Formal analysis
Sn2 Testing

Figure 5. An example user-interface for the reviewer.

(HCI) experts (Wan, 2010), and the outcome from this informed the development of the second prototype (version 2.0) by a group of MSc software engineering students (Mazurek et al., 2011). The second prototype was evaluated by using both user- and expert-based evaluation techniques and the outcome from this informed amendments to the next prototype (version 3.0, the current version of the software as illustrated in Section 5). This was then taken to a set of users and HCI experts for further evaluation, to assess the system's usability and acceptability with the target audience (Djaelangkara, 2012).

Two safety engineers were invited to conduct cooperative evaluations in a controlled environment separately. Each participant was asked to come to a room where the software was ready to be used on a prepared machine. The participants were then briefed about the purpose and procedure of the evaluation and the current version of the software. Each participant was given a set of tasks to perform as both a proposer and a reviewer. The tasks start from the creation of a safety case and continue until the completion of the review of the safety case. Participants are allowed to ask questions and seek clarifications. The experimenter observed the user reactions to using the system. Interviews were then conducted concerning their experience of using the system and improvements that could be made to the functionality of the system. Each evaluation took about one hour. Both users completed the predesigned tasks without difficulty. The comments from their interviews are very positive. Participant 1 thought that the system now has a clear meaning in a way that the reviewing process can be done individually by the proposer and the reviewer. He particularly liked the clarity of the status of the dialogue and the design feature that every action performed has a feedback as a result of which the number of click mistakes would be reduced. Participant 2 liked the idea of maintaining the dialogue history between the proposer and reviewer, and the fact that the database back-end is obvious and that it is now clear how it would be used in practice, in a way that the participant thought it was unclear before.

Two HCI experts were invited to evaluate the learnability of the system, that is, how easily the application can be learned and used by novice users. Each evaluation was conducted in three stages: walkthrough, usability questionnaire, and interview. At the walkthrough stage, participants were given five small tasks to perform on the application. Each task consisted of several steps. For every step executed, they would try to answer the following three questions:

- Do you know what to do?
- Do you see how to do it?
- Do you understand from the feedback whether the action was correct or not?

The participants were not allowed to ask any questions and the screen actions were recorded. At the usability questionnaire stage, each participant was given 10 minutes to explore the system on his or her own or, if they prefer, to be given a set of tasks to do. They then were asked to complete a usability questionnaire based on Nielsen's heuristics (Dix, Finlay, Abowd, & Beale, 2004). This part of the evaluation was created to identify any usability issues appearing in the software. During the interview stage, participants were asked to provide comments on how the application can be improved to be more easy to use. The evaluation results from the HCI experts are very positive as well. The usability questionnaires scored in the range 4–5 (where 0 means very poor design and 5 perfect) for each of the usability criteria and with an overall average of 4.3.

The participants from both the user and expert evaluations also made a number of suggestions that can be used to improve the functionality and usability of the system and the model. With regard to the model, users would like to see multiple proposers and reviewers in the future. They also suggest that there should be indications to let the user know that the other elements have been affected by a withdrawal of an argument element. In the light of user feedback, the current arrangement appears to be insufficient especially in cases when the withdrawn element is a sole support of another element (say Q) or when an argument element (say R) only supports the withdrawn element. The affected elements should be highlighted for users' discretion. The user, however, is liable to commit to Q if further support to Q is provided, and commit to R if R is used to support other elements.

Concerning the implementation, a number of functional and usability issues with the system are revealed. Users would like to have a delete function alongside a withdrawal. In the light of feedback, it becomes clear that the two may have different roles to play, i.e. 'withdrawal' is part of the model that can be used to remove earlier commitments and 'delete' is part of the implementation that can be used to manipulate argument elements within a turn. They also stress the need for presenting the final safety argument diagram without the unnecessary elements that resulted from the reviewing process and making sure that the completed sessions are still editable. Further details can be found in (Djaelangkara, 2012).

7. Conclusions and further work

The argument-based approach to safety case development has been widely adopted in Europe, and increasingly worldwide (e.g., Australia and Japan) and in a wide variety of domains (including defence, automotive, medical, and rail) (Haddon-Cave, 2009; Yuan & Kelly, 2012). Given the subjective nature of arguments, reviews are always necessary to independently scrutinise and challenge the arguments. Despite the graphical notations (e.g. GSN) that have been developed for the representation of safety arguments, tools for reviewing safety arguments are still lacking. Dialectical models seem to be the natural fit for this purpose. A new dialectical model specially designed for safety argument review has been proposed. A system operationalising the review model has been iteratively constructed and evaluated. The evaluators are in favour of the dialectical approach in safety argument review, particularly in a computational environment. The evaluations provide positive evidence for the usability of the system in general and the review model in particular.

It is believed that the work reported in this paper makes a valuable contribution to the field of safety arguments and dialectics. Concerning the former, we have proposed a new approach for safety argument review and developed a unique system that is ready to be used to facilitate safety engineers and assessors to review safety arguments. User experience of using such a system is essentially favourable. The potential pay-off in expanding computer system safety engineering is enormous. Concerning the latter, we have proposed a new model for safety argument review and this directly contributes to the dialectics. The model has a number of desirable properties that are essential for safety argument review, for example, conformance to GSN, visualised interaction, traceable reviews, expressive adequacy, and user friendly. Typically, the model moves from a mere graphical approach to a dialectical approach to facilitate argument evaluation. To the best of our knowledge, this is the first model that deals with safety argument review in particular and argument review in general. Furthermore, there is great scope for an interesting and fruitful interplay between research within dialectics per se, and research on their utilisation in computer system safety engineering. It is hoped that this paper will move this interplay forward.

There are several ways to carry this work forward. Our immediate work is to address the issues revealed from the evaluation and conduct a large-scale usability evaluation. With a suitable review model framework, the quality of review arguments is not guaranteed as this largely relies on the participants' strategic wisdom. We are planning to provide users with a software agent, which can detect common argument fallacies (e.g. conflict and circular arguments) in system safety arguments in line with Greenwells, Holloway, and Knight's (2005) identification. We are also planning to investigate the computational use of safety argument schemes (e.g. Yuan & Kelly, 2011) to support safety argument review as each scheme provides a set of critical questions that can be used to evaluate arguments. Furthermore, SARM can also be formally specified in line with (Amgoud, Maudet, & Parsons, 2000; Amgoud, Parsons, & Wooldridge, 2003; Johnson, McBurney, & Parsons, 2003; Kontarinis, Bonzon, Maudet, & Moraitis, 2012; Yuan & Wells, 2013), thus enabling automated analysis of its properties to take place.

Acknowledgements

The authors thank all the participants who took part in the evaluation of the Argument Review Tool as well as research and teaching staff of the Department of Computer Science at the University of York, UK, who have generously donated their time, effort, and knowledge towards the progress of the project.

References

Amgoud, L., Maudet, N., & Parsons, S. (2000). Modelling dialogues using argumentation. *Proceedings of the fourth international conference on multi-agent systems* (ICMAS 2000). Boston, MA.

Amgoud, L., Parsons, S., & Wooldridge, M. (2003). Properties and complexity of some formal inter-agent dialogues. *Journal of Logic and Computation, 13*(3), 347–376.

Bench-Capon, T.J.M. (1998). Specification and implementation of Toulmin dialogue game. *Proceedings of the 11th international conference on legal knowledge based systems* (JURIX) (pp. 5–20). Nijmegen: Gerard Noodt Institute (GNI).

Bishop, P.G., & Bloomfield, R.E. (1995). The SHIP safety case – A combination of system and software methods. *Proceedings of the 14th IFAC conference on safety and reliability of software-based systems* (pp. 12–15), Bruges.

Bishop, P.G., & Bloomfield, R.E. (1998). *A methodology for safety case development. Safety-critical systems symposium (SSS 98)*. Birmingham.

Bishop, P.G., Bloomfield, R.E., & Guerra, A.S.L. (2004). The future of goal-based assurance cases. *Proceedings of workshop on assurance cases. Supplemental volume of the 2004 international conference on dependable systems and networks* (pp. 390–395), Florence.

Buckingham Shum, S. (2008). Cohere: Towards web 2.0 argumentation. *Proceedings of the 2nd international conference on computational models of argument (COMMA'08)*, Toulouse.

Dijkstra, E.W. (1972). Chapter I: Notes on structured programming. In O.J. Dahl, E.W. Dijkstra, & C.A.R. Hoare (Eds.), *Structured programming*. London: Academic Press.

Dix, A., Finlay, J., Abowd, G., & Beale, R. (2004). *Human computer interaction* (3rd ed.). Harlow: Prentice Hall.

Djaelangkara, S. (2012). *Online system for safety argument review* (MS thesis) Department of Computer Science, University of York.

Emmet, L., & Cleland, G. (2002). Graphical notations, narratives and persuasion: A pliant systems approach to hypertext tool design. *Proceedings of the thirteenth ACM conference on hypertext and hypermedia, conference on hypertext and hypermedia* (pp. 55–64), College Park, MD.

Gordon, T.F., & Walton, D. (2006). The Carneades argumentation framework: Using presumptions and exceptions to model critical questions. In P.E. Dunne & T.J.M. Bench-Capon (Eds.), *Proceedings of computational models of argument (COMMA 2006)* (pp. 195–207). Amsterdam: IOS Press.

Greenwell, W.S., Holloway, C.M., & Knight, J.C. (2005). A taxonomy of fallacies in system safety arguments. *Proceedings of the international conference on dependable systems and networks*, Yokohama, Japan.

Haddon-Cave, Q.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*. London: Stationery Office.

Hamblin, C. (1971). Mathematical models of dialogue. *Theora, 37*, 130–155.

Hawkins, R., Habli, I., Kelly, T., & McDermid, J. (2013). Assurance cases and prescriptive software safety certification: A comparative study. *Safety Science, 59*, 55–71.

Hawkins, R., Kelly, T., Knight, J., & Graydon, P. (2011). A new approach to creating clear safety arguments. *Advances in Systems Safety – Proceedings of the nineteenth safety-critical systems symposium* (pp. 3–23). Southampton: Springer.

International Electrotechnical Commission. (2010). *Functional safety of electrical/electronic/programmable electronic safety-related systems (IEC 61508 ed2.0)*. Retrieved May 20, 2011, from http://www.iec.ch/

Johnson, M., McBurney, P., & Parsons S. (2003). When are two protocols the same? In M.P. Huget (Ed.), *Communication in multiagent systems: Agent communication languages and conversation policies* (pp. 253–268). Lecture Notes in Artificial Intelligence 2650. Berlin: Springer Verlag.

Kelly, T.P. (1999). *Arguing safety – A systematic approach to safety case management* (PhD thesis) Department of Computer Science, University of York, York.

Kelly, T.P. (2005). Using software architecture techniques to support the modular certification of safety-critical systems. *Proceedings of eleventh Australian workshop on safety-related programmable systems*, Melbourne.

Kelly, T.P. (2007). Reviewing assurance arguments – A step-by-step approach. *Proceedings of workshop on assurance cases for security – The metrics challenge, dependable systems and networks (DSN)*, Edinburgh.

Kelly, T.P. (2008). Are safety cases working? *UK Safety Critical Systems Club newsletter, 17*(2), 31–33.

Kelly, T.P., & Weaver, R.A. (2004). The goal structuring notation – A safety argument notation. *Proceedings of the dependable systems and networks 2004 workshop on assurance cases*, Florence.

Kontarinis, D., Bonzon, E., Maudet, N., & Moraitis, P. (2012). Picking the right expert to make a debate uncontroversial. *Proceedings of the fourth international conference on computational models of argument* (pp. 486–497), Vienna.

Krabbe, E. (2000). Symposium on argument and computation group: argument and computational societies, position paper. *Symposium on argument and computation*, Bonskeid House, Perthshire, Scotland.

Leveson, N. (2011). The use of safety cases in certification and regulation. *Journal of System Safety, 47*(6), 1–5.

Mackenzie, J.D. (1979). Question–begging in non-cumulative systems. *Journal of Philosophical Logic, 8*, 117–133.

Maudet, N., & Moore, D. (2001). Dialogue games as dialogue models for interacting with, and via, computers. *Informal Logic, 21*(3), 219–243.

Mazurek, M., Gerasimou, S., Madan, B., & Setivarahalli, G. (2011). *Software for safety argument review* (MSc Software Engineering Team Project Report). University of York, York.

McDermid, J.A. (1991). Safety arguments, software and system reliability. *Proceedings of the international symposium on software reliability engineering*, IEEE Computer Society, Los Alamitos, CA.

McDermid, J.A. (2001). Software safety: where's the evidence? *Proceeding of the 6th Australian workshop on industrial experience with safety systems and software*, Brisbane: Australian Computer Society (pp. 1–6).

Object Management Group. (2010). *Argument metamodel*. Retrieved from http://www.omg.org/spec/ARM

Prakken, H. (2000). On dialogue systems with speech acts, arguments, and counterarguments. *Proceedings of JELIA'2000, the 7th European workshop on logics in artificial intelligence* (pp. 224–238). Springer Lecture Notes in AI 1919. Berlin: Springer Verlag.

Prakken, H. (2005). Coherence and flexibility in dialogue games for argumentation. *Journal of Logic and Computation, 15*, 1009–1040.

Ravenscroft, A., & Pilkington, R.M. (2000). Investigation by design: Developing dialogue models to support reasoning and conceptual change. *International Journal of Artificial Intelligence in Education, 11*, 273–298.

Reed, C.A., & Rowe, G.W.A. (2004). Araucaria: Software for argument analysis, diagramming and representation. *International Journal of AI Tools, 13*(4), 961–980.

UK Ministry of Defence. (1997). *Defence standard 00-55 – The procurement of safety critical software in defence equipment*. Retrieved May 20, 2011, from http://www.dstan.mod.uk/.

UK Ministry of Defence. (2004). *Defence standard 00-56 (Issue 3), Safety management requirements for defence systems*. Retrieved May 20, 2011, from http://www.dstan.mod.uk/.

Walton, D. (1998). *The new dialectic: Conversational contexts of argument*. Toronto: University of Toronto Press.

Walton, D., & Krabbe, E. (1995). *Commitment in dialogue: Basic concept of interpersonal reasoning*. Albany: State University of New York Press.

Wan, F. (2010). *Computer-assisted argument review – A dialectics approach* (MSc thesis) Department of Computer Science, University of York, York.

Wang, Y., & Bryant, A. (2002). Process-based software engineering: Building the infrastructures. *Annals of Software Engineering, 14*(1–4), 9–37.

Wassyng, A., Maibaum, T., Lawford, M., & Bherer, H. (2011). Software certification: Is there a case against safety cases? *Foundations of computer software. Modeling, development, and verification of adaptive systems* (pp. 206–227). Lecture Notes in Computer Science, Vol. 6662. Berlin: Springer-Verlag.

Weaver, R.A., Fenn, J., & Kelly, T.P. (2003). A pragmatic approach to reasoning about the assurance of safety arguments. *Proceedings of 8th Australian workshop on safety critical systems and software (SCS'03)*, Canberra.

Yuan, T., & Kelly, T. (2011). Argument schemes in computer system safety engineering. *Informal Logic, 31*(2), 89–109.

Yuan, T., & Kelly, T. (2012). Argument-based approach to computer system safety engineering. *International Journal of Critical Computer-based Systems, 3* (3), 151–167.

Yuan, T., Moore, D., & Grierson, A. (2003). Computational agents as a test-bed to study philosophical model 'DE', a development of Mackenzie's 'DC'. *Informal Logic, 23*(3), 263–284.

Yuan, T., Moore, D., & Grierson, A. (2007). A human computer debating system and its dialogue strategies. *Special Issue on Computational Models of Natural Argument of the International Journal of Intelligent Systems, 22*(1), 133–156.

Yuan, T., Moore, D., & Grierson, A. (2008). A human-computer dialogue system for educational debate, a computational dialectics approach. *International Journal of Artificial Intelligence in Education, 18*(1), 3–26.

Yuan, T., Moore, D., Reed, C., Ravenscroft, A., & Maudet, N. (2011). Informal logic dialogue games in human-computer dialogue. *Knowledge Engineering Review, 26*(2), 159–174.

Yuan, T., & Wells, S. (2013). ProtOCL: Specifying dialogue games using UML and OCL. *Proceedings of the ICAIL'13 workshop on computational models of natural argument*, Rome.