A Systematic Approach to Safety Case Maintenance

T P Kelly, J A McDermid

Department of Computer Science
University of York
Heslington
York YO10 5DD

Abstract

A crucial aspect of safety case management is the ongoing maintenance of the safety argument through life. Throughout the operational life of any system, the corresponding safety case can be challenged by changing regulatory requirements, additional safety evidence and a changing design. In order to maintain an accurate account of the safety of the system, all such challenges must be assessed for their impact on the original safety argument. This is increasingly being recognised by many safety standards. However, many safety engineers are experiencing difficulties with safety case maintenance at present, the prime reason being that they do not have a systematic and methodical approach by which to examine the impact of change on safety argument. The size and complexity of safety arguments and evidence being presented within safety cases is increasing. Nowhere is this more apparent than for Electrical, Electronic and Programmable Electronic systems attempting to comply with the requirements and recommendations of software and hardware safety standards such as IEC 61508 [1] and U.K. Defence Standards 00-54 [2], 00-55 [3], and 00-56 [4]. However, this increase in safety case complexity exacerbates problems of comprehension and maintainability later on in the system lifecycle.

This paper defines and describes a tool-supported process, based upon the principles of goal structuring, that attempts to address these difficulties through facilitating the systematic impact assessment of safety case challenges.

1 Introduction

In the first instance the safety case argument will typically be constructed and presented (e.g. to a regulatory authority) prior to the system operating for the first time. The argument is often therefore based on estimated and predicted system and operator behaviour rather than observed evidence. For this reason alone, even in the absence of changes to the system or the regulatory environment, it is almost inevitable that the safety case will require updating throughout the operational lifetime of the system. Operational experience must be reconciled with the predictions made in the initial safety argument.

The system operators, as the ‘owners’ of the safety case, are typically responsible not only for its initial production but also for its maintenance throughout the lifetime of the system. There is growing recognition in the standards that appropriate mechanisms must be in place for the ongoing maintenance of the safety case. For example, the U.K. Railways (Safety Case) Regulations 1994 [5] states in Regulation 6(1) that:

> *“A Person who has prepared a safety case persuant to these Regulations shall revise its contents whenever it is appropriate…”*

Similarly, for developers of software based defence systems in the U.K., the Ministry of Defence Safety Standard 00-55 [3] states in section 4.7.1. that:

> *"After the preparation of the operational Safety Case, any amendments to the deployment of the system should be examined against the assumptions and objectives contained in the Safety Case."*

Although standards, such as those mentioned, demand appropriate and adequate revision of safety cases, they offer little advice on how such operations can be carried out. The safety case is a complex web of inter-dependent parts: safety requirements, argument, evidence, design and process information. As such, a single change to a safety case may necessitate many other consequential changes - creating a 'ripple effect'. The difficulty faced with current safety cases lies in discerning those consequential changes through the morass of poorly structured documentation. The level of assurance as to how well a safety case has been updated in the light of a change depends largely on the degree to which the document has been understood. There is little guarantee that all changes have been dealt with equally and systematically. Subjectivity plays a greater role in safety case maintenance than is desirable.

2 Current Problems in Safety Case Maintenance

Working from the published literature on this topic and from discussions with Rolls-Royce safety engineers the key problems currently being faced in safety case maintenance have been identified as the following:

- **Difficulty in recognising (importance of) challenges to the safety case**
  Some changes, such as a minor operational role change or a change in operator behaviour, may seem innocuous at first when given superficial consideration, but actually have a significant impact with respect to the context and argument of the safety case.
- **Difficulty in identifying the indirect impact of change**
  Identifying the initial impact of a change is only the starting point of the change management process. Safety arguments are a web of dependencies: it is necessary to identify the 'knock-on' effects of changes. To identify these *indirect* effects the engineer must be able to see clearly the structure of the argument and where the dependencies lie. However, these dependencies are often inadequately presented, or are obscured in current text-based safety arguments.
- **Lack of assurance / justification of the change process**
  Faced with a potential challenge to the safety case, those responsible for the maintenance of the safety case must decide on an appropriate response. This response will lie somewhere between the two extremes of doing nothing to the safety case and doing 'everything' (i.e. complete safety case revision). These decisions about the level and nature of response made to a particular challenge must be expressed explicitly and justified in order to have confidence in the ongoing validity of the safety case.
- **Insufficient information recorded to support the change process**
  The previous problems have addressed the *quality* of the information recorded in the safety case. However, there is also a problem concerning the *quantity* of information recorded. If information (such as the assumptions and context surrounding safety claims) simply isn't recorded in the safety case then recognition of the impact of any changes requires a significant amount of

detective work! In many existing safety cases, context is often assumed knowledge and assumptions are often implicit.

Together these problems result in an informal and often subjective change management process where even a basic level of repeatable and systematic impact analysis cannot be guaranteed. Given that the safety case should be maintained as a living argument that always correctly portrays the safety of a system, this informality is a serious concern.

3 Application of GSN to Change Management

A fundamental concern underlying the problems of safety case maintenance identified in the previous section is the poor perception of the individual elements of conventionally structured safety cases and of the interdependencies that exist between them. The Goal Structuring Notation provides a clear conceptual model of the safety case – representing its elements and interdependencies explicitly. Using the framework GSN provides as a basis for establishing a configuration model for safety cases, we illustrate that is possible to formulate a systematic approach to reasoning about and handling change.

The safety case can be considered as consisting of the following four elements:

- **Requirements** – the safety objectives that must be addressed to assure safety
- **Evidence** – information from study, analysis and test of the system in question
- **Argument** – showing how the evidence indicates compliance with the requirements
- **Context** – identifying the basis (e.g. assumptions) of the argument presented

These elements are obviously inter-dependent. **Error! Reference source not found.** illustrates the macro-dependencies that exist between these four elements.

This is a simplification of the dependencies that exist between these elements. Dependencies could also exist, for example, between pieces of evidence – e.g. between component failure modes and rates in a Failure Modes and Effects Analysis and basic events in Fault Tree Analysis. **Error! Reference source not found.** however, communicates those dependencies that exist through the *intentional* relationships of the *safety argument*. Recognising these dependencies helps to highlight where consistency must be maintained when handling change.

This paper proposes an approach that helps engineers to ask specific and structured questions regarding the impact of change on the safety case through utilising the documented dependencies presented in a goal structured safety argument.

The Goal Structuring Notation [6,7] has been specifically defined to model the entities and relationships shown in **Error! Reference source not found.**. *Requirements* are represented in the notation as top level *Goals*. *Evidence* is represented in the notation as *Solutions*. *Contextual information* is represented in the notation as *Context*, *Assumption*, *Justification* and *Models*. *Argument* is communicated through the structuring of *Goals* supported by *sub-goals*. **Error! Reference source not found.** illustrates how a simple goal structured argument for a computer based control system can be divided into the four essential elements – requirements, context, evidence and argument.

In Figure 2, the requirement is to support the claim that the ‘Control System is Safe’. The argument is presented on the basis of two ‘legs’. Firstly, on the left-hand side of

the goal structure, the claim is that all hazards associated with the control system have been sufficiently addressed. To provide the context for such a claim it is necessary to make clear the source of identified control system hazards. For this purpose, 'Ref Y' is used to refer to the results of a Functional Hazard Analysis of the functions of the control system. Secondly we have to define what is meant by the term 'sufficient'. This is done by reference to tolerability targets (i.e. acceptable risk levels) in 'Ref Z'.

The second leg of the argument, shown on the right hand side of the goal structure, argues the claim that 'Software has been developed to an integrity level appropriate to the hazards involved'. This claim also must clearly sit in the context of the identified hazards. Additionally we need to make clear what is meant by 'development to an Integrity Level (IL)'. This is done through reference to the IL guidelines provided by the appropriate safety standard 'Ref X' (e.g. Defence Standard 00-55 [3]).

The hazard claim is then supported by individual claims (one qualitative, two quantitative) relating to the three key control system hazard identified. The specific quantitative target quoted is justified as being the appropriate limit for hazards of 'catastrophic' severity. The qualitative argument that the control system condition (H1) cannot occur is then supported by reference to Formal Verification evidence. The quantitative claims are supported through reference to Fault Tree Analysis of hazards H2 and H3.

The integrity level claim is split into different integrity level claims for the primary and secondary elements of the control system (e.g. in accordance with the apportionment rules of Defence Standard 00-56 [4]). Both these claims are then supported by reference to appropriate evidence of the control system development process (e.g. tool and test audits).

Through the explicit links of a goal structure, such as those shown in **Error! Reference source not found.**, traceability is provided between the elements of the safety case argument. The following relationships are communicated:

- How requirements are supported by argument claims
- How argument claims are supported by other (sub) argument claims
- The context in which argument claims are stated
- How argument claims are supported by evidence

Such relationships are also present in conventional text-only safety cases. However, it is rare that they are communicated as clearly and explicitly as in a goal structure.

3.1 Establishing a Safety Case Configuration Model

In conventional configuration management, the 'configuration' refers to

> *"The totality and the inter-relationships of the hardware, software, firmware, services and supplies that make up the system at a given reference point in time" [8]*

This definition can be adapted to the safety case domain. In this context, we define the configuration as:

> *"The totality and the inter-relationships of the requirements, argument, evidence and context that make up the safety argument at a given reference point in time"*

A conventional configuration model consists of two parts:

- **Configuration Items** (CIs): Entities within a configuration that satisfy an end use function that can be uniquely identified at a given reference point. [8]

- **Configuration Relationships** (CRs): The relationships between Configuration Items that have been established at a given stage in the development lifecycle [9]

Using the framework of the Goal Structuring Notation it is possible to relate these concepts to the safety case domain.

- **Configuration**: A goal structured safety argument
- **Configuration Items** (CIs): Individual entities within the goal structure representation of a safety argument – i.e. goals, strategies, solutions, contexts, models, assumptions, justification etc.
- **Configuration Relationships** (CRs): The relationships established between the elements of a goal structure – i.e. instances of the *SolvedBy* and *InContextOf* relations. For example, these include the relationship declared between a parent goal and a child goal, and between a goal and an associated assumption.

Using the Goal Structuring Notation as a configuration model (and therefore an individual goal structure as a configuration), the chapter now goes on to propose a process for managing change applied to the safety case in the following section.

4 A Safety Case Change Process

The safety case change activity can be thought of as consisting of two phases:

- The **Damage** Phase – Where a change is assessed for its impact on the safety argument of the safety case
- The **Recovery** Phase – Once the damage has been identified, the process of identifying a recovery action and following though the consequences of that action in recovering the safety argument.

There is an iterative (and potentially concurrent) relationship between these two phases. The action identified to *recover* the damaged part of the safety case may also result in *damage* to other parts of the safety case. For any one change, several iterations of the damage and recovery activities may be necessary to arrive again at a consistent and correct safety case. This highlights the importance of having an efficient and systematic process for carrying out these activities.

Using a goal structured safety case it is possible to provide a systematic structure to the activities carried out in these two phases. This structure is shown in Figure 3.

The following describe how using GSN as a configuration model can support the six steps identified in Figure 3.

4.1 Step 1: Recognise Challenges to the Validity of the Safety Case

An important aspect of the through life maintenance of the safety case is awareness of challenges that could potentially render the safety case argument invalid – i.e. being aware of the vulnerability of the safety case argument to external change.

Using the model of the safety case proposed in Figure 1 – the role of the safety argument within the safety case is to establish the relationship between the *available evidence, safety objectives* and *contextual information* (such as design information). These three elements can be viewed as the 'givens' of the safety argument. Challenges to the validity of a safety argument will arise through challenging one of these givens, i.e. something in the 'real-world' context (outside of the safety case) will challenge the basis of the safety case presented.

The safety case will have been produced initially to present a valid safety argument with respect to the regulations, evidence and contextual information appropriate at the time. The difficulty in safety case maintenance is that any or all of these three elements may *change over time*. For example:

- An additional regulatory requirement may be added following an operational incident. An example of this from the civil aerospace domain would be the addition of a regulation regarding inadvertent thrust reverser deployment [10] following the Lauda Air thrust reverser deployment in flight accident.
- The design of a system may be changed for perfective, corrective or adaptive maintenance reasons or through technology obsolescence. Hogberg, in [11], describes responding to unanticipated *problems* with the design of a class of nuclear reactors.
- Assumptions regarding the operational lifetime of a system also form an important part of the safety case context. Such assumptions may be challenged by a desire to extend plant life beyond the originally intended period. Clarke, in [12], describes such a case for the life-extension of the U.K. civil Magnox nuclear reactors.
- Operational experience may challenge the evidence used as the basis of the original safety argument. For example, the safety case may estimate that a certain failure mode of a component will occur at a certain rate. This rate may be brought into question by operational data.
- Operator behaviour may change as familiarity and experience with a system grows. Similarly, operators may attempt to operate a system outside of its originally defined operational bounds. Such situations can challenge the validity of the *context* (especially assumptions) in which the original safety argument is presented.

In abstract it is easy to define the role and purpose of this first (recognition) step in the maintenance process. However, in reality *recognition* of challenges can be extremely difficult, and challenges can be subtle. For example, in the latter case of changed operator behaviour consider a control system with a warning alarm that sounds to indicate a hazardous situation, such as the buzzer that is sounded in U.K. trains fitted with the Advanced Warning System when a track-side 'Signal is Passed at Danger' (SPAD). Initially, the safety argument rests upon the (perhaps implicit) assumption that operators will take note of such alarms and act accordingly. However, over time the operator may become conditioned to ignore the alarm (e.g. because it sounds more often than necessary) such that they instinctively cancel the alarm when raised and continue operation as if it had never occurred. If this became the case, those responsible for the ongoing validity of the safety case must recognise that this challenges the basis (premise) on which the original safety case was argued (i.e. the context of the safety argument is now different from that originally assumed).

The start point of a systematic change process is the activity of attempting to identify and acknowledge such changes on a routine basis (as recognised in many of the safety standards such as the HSE Civil Nuclear Safety Assessment Principles [13] and Defence Standard 00-55 [3]).

The starting point of a systematic process for ensuring the ongoing validity of the safety case is the identification and recognition of such changes on a routine basis. Operational data should be collected through in-service monitoring. This is

recognised in a number of the existing safety standards. For example, the HSE Civil Nuclear Safety Assessment Principles [13] contain the following principle:

> *Maintenance, inspection and testing (Principle 329):*
>
> *The requirements for in-service testing, inspection or other maintenance procedures and frequencies for which specific claims have been made in the safety case should be identified and included in a maintenance schedule.*

To record system anomalies and updates, failure and correction maintenance action reporting systems should be established. In Defence Standard 00-55 [3] the following requirement is stated:

> *8 Data Management*
>
> *8.1 The Contractor shall establish a Data Reporting Analysis and Corrective Action System (DRACAS) which shall be a documented closed loop system for reporting, collecting, recording, analysing, investigating and taking timely corrective action on all incidents that may have an impact on safety.*

Having used such reporting systems to recognise and record information that may impact the safety case, the next step in the process is to express those challenges in the terms of the recorded safety argument.

4.2 Step 2: Expressing Challenge in Goal Structure Terms

Step 2 is concerned with expressing an identified potential challenge in terms of a challenge to elements within the goal structure representation of the safety case argument.

There is a correspondence between the types of change introduced and the elements of a typical goal structure. A 'GSN Challenge' will be expressed always in terms of a challenge to elements of the notation representing the requirements, evidence or context.

Figures (4-6) illustrate the mappings shown in the above table by providing sketch examples of requirements, evidence and context challenges expressed in GSN terms. It is important to realise that within this step, and therefore also in the examples presented, the concern is to express the *initial* challenge to a goal structured safety argument (i.e. the start point of impact assessment), rather than the total impact (which will be explored in Step 3).

The convention introduced to denote that a GSN element or relationship is challenged is to place a cross (×) over that item.

Figure 5 illustrates a requirement change that translates into a challenge to a context reference made within a goal structure. The HSE Safety Assessment Principles are given as context to a strategy that bases its arguments upon them. If these principles change (e.g. are revised or added to) the basis of the existing argument is challenged.

Figure 4 depicts a real-world evidence change that translates directly into a challenge to a solution given within a goal structure. In this case, a fault tree is used to satisfy the probability claim for Hazard X. If the fault tree is called into question (e.g. through operational experience contradicting the basic fault event probabilities used or the implicit claims of independence) the role of this piece of evidence as a solution in the safety argument is challenged.

Figure 6 shows a real-world context change that translates directly into a challenge to a context reference made within a goal structure. In this case, the claim of operational

safety is defined only within certain operating limits. If these operating limits were exceeded for any reason, the basis of the claim is challenged.

4.3 Step 3: Using the Goal Structure to Identify Impact of Challenge

If a solution item is challenged (as shown in Figure 4) it challenges its *role* as a solution to all goals relying upon it through the *SolvedBy* relationship (shown by the lines headed with solid arrows). Equally, if a context item is challenged (as shown in Figure 6) it challenges the relationship with all goals previously expressed in the context of that item using the *InContextOf* relationship (shown by the lines headed with hollow arrows).

It is the challenge to the *structure* of the safety argument that must be explored (propagated) to determine the ultimate impact of any challenge on the *claims* of the safety argument. Based upon the semantics of the notation, the rules for the propagation of change within a goal structure are provided within the following sections.

Propagation of Challenges to Goals, Strategies and Solutions

Changing a goal, strategy or solution (G) within a goal structure challenges the following relationships within the goal structure:

- The role of G as a solution of parent goals or strategies (i.e. items higher up the goal structure). This is not a concern for the top goals of a goal structure.
- The role of G as a parent (objective) of supporting elements (i.e. to items lower down the goal structure). This is obviously not a concern for the solution elements of a goal structure.
- The relationship between G and its stated context (i.e. to items left and right of the core argument)

Propagation of Challenges to Context, Models, Justifications and Assumptions

The effect of changing a context element is made more complicated that that of changing a goal, strategy or solution owing to the *inheritance* of context elements implied by the semantics of the notation. Changing a context element challenges not only the most immediately associated goal or strategy but also *all* of the child goals and strategies underneath that item within the goal structure.

Changing a context element (C) challenges the following elements within the goal structure:

- All goals, strategies and solutions (G) that introduce C as context (through the *InContextOf* relationship).
- All goals, strategies and solutions which inherit C as context (i.e. all children of *G*).

When a goal, strategy or solution is challenged by a context change, the rules of change propagation for these elements (defined in the previous section) apply.

Potential vs. Actual Change Effect – The Role of the Safety Engineer

It should be noted that the rules described for the propagation of change over a goal structure define the *potential* change effect rather than necessarily the *actual* change effect. The approach taken is *pessimistic*. The role of the safety engineer responsible for maintaining the safety argument is then to examine each of these potential areas of

impact to decide which require further investigation and which can be ignored (i.e. where the change can be considered *benign*).

The impact of any change will typically be explored to the point at which a (believed) appropriate recovery action can be determined.

4.4 Step 4: Deciding Upon Action to Recover Damaged Argument

Recovery is the process of returning the safety argument to a correct, consistent and complete state. The impact of a change (identified in Step 3) may mean that claims made within the safety argument (e.g. concerning the meeting of regulatory or customer requirements) are no longer supported. In such cases, the safety argument must be 'repaired' in order to bring the safety argument back to the original state of supporting the claims.

It is necessary to decide upon an appropriate action to recover the safety argument. This decision is set in the context of, and should be focused by, the impact that has been identified. For example, if after Step 3 it is found that the claim that 'No single point of failure can lead to hazard' can no longer be supported, then appropriate action should be taken towards *re*-supporting this objective – e.g. by making a design change that introduces redundancy. (However, it is important to note that safety may be only one of the factors in deciding upon a recovery action, others perhaps being availability, through-life cost etc.)

In deciding how to recover the argument the following questions should be considered:

- **Can the requirements of the safety argument be altered (e.g. weakened) such that the safety argument still holds?**
- **Can the context of the safety argument be altered (perhaps restricted) such that the safety argument still holds**?
- **Can additional evidence be found / created such that the safety argument still holds?**

The particular action to recover from a challenge can only be decided on a case-by-case basis. However the impact history recorded from Step 3 will offer useful information. Consider the impact path shown in Figure 7.

In Figure 7 a general safety claim can no longer be supported **because** a supporting system reliability claim has failed. This claim has failed **because** a supporting fault tree claim has failed. This claim has failed **because** a component reliability claim has failed. This claim has failed **because** a supporting Failure Modes and Effects (FMEA) solution has been challenged (e.g. by operational experience).

The overall consequence of this change is that the general safety claim fails. However, the impact path communicates to the safety engineer that more reliable components are required in order that the FMEA evidence can once again support the component reliability claim. The fault tree can then be updated to continue to support the system reliability claim, and the latter can then continue to support the general safety claim.

Side-Effects of Recovery Action

The motivation for identifying and taking recovery action is the need to repair that part of the safety argument identified as damaged (as a result of Step 3). However the recovery action may itself necessitates further change to the safety argument. For example, a design change proposed in response to a challenge to one part of the safety

argument may well challenge evidence used in another part. The impact of the recovery action must be assessed and managed in the same manner as the initial challenge.

4.5 Step 5: Recover Identified Damaged Argument

Damaged claims were identified at the end of step 3. In the damage process the effects of change are identified through the extent to which they impact, bottom-up, the claims made in the safety argument. The recovery process however works in the opposite direction – top-down – starting from the most fundamental claim challenged (i.e. the claim that is highest in the goal structure) and recovering the argument step-by-step downwards until the claims can be related back to the available evidence.

5 Reactor Control System Safety Argument Example

This section illustrates the application of the impact assessment process to part of the safety argument for a computer based nuclear reactor control system (derived from details given in [14]). A potential challenge to the validity of the timing analysis evidence is considered.

Figure 8 shows the structure of claims made regarding the timeliness of the trip system response.

Step 1: Recognising the Challenge to the Safety Case

After initial acceptance of the safety argument, it is later recognised that there was a flaw in the static timing analysis tool used to determine the worst case response time.

Step 2: Expressing Change in Terms of GSN Elements

After examining the peripheral (context, solution and top requirement) elements of the safety argument, it is identified that this challenge directly concerns ***Sn3*** *– Timing Analysis Results as* shown in Figure 8.

Step 3: Use GSN to Identify Impact

The challenge to **Sn3** damages the claim **G.TIM.STATIC.1.** Damaging **G.TIM.STATIC.1** potentially damages the claim **G.TIM** (as shown in Figure 9).

At this point, one observes that a diverse argument has been applied in the quantitative claims put forward in support of **G.TIM**. Both analysis and test have been used. Even though **G.TIM.STATIC** is questioned, there is still the test claim **G.TIM.TEST** claim to support **G.TIM**. One observes also that a safety margin exists between the **G.TIM** and **G.TIM.TEST** claims, which increases confidence of **G.TIM.TEST** being able to support **G.TIM**.

Step 4: Decide upon Recovery Action

Given the diversity of the argument, it is possible to decide simply to accept the damage created by challenging the timing analysis results. However, the remaining argument would be weaker and more questionable. Another possibility would be to batch the change, and recover from the timing analysis challenge at a later point in time.

If responding to the change immediately, the safety engineer must identify an approach that will recover the damaged leg of the argument (i.e. the damaged **G.TIM.STATIC**, **G.TIM.STATIC.1** and **Sn3** elements). The decision could be to

throw it away and replace with a completely different supporting argument – i.e. prune back the argument to **G.TIM** and start again. Alternatively, the engineer could decide to replace 'like for like' and reinstate the argument in a form similar to that used already. Given that the challenge was due only to a flaw in the tool, reinstating the argument in the same form, after reworking the analysis on the corrected version of the tool, is probably the most effective option.

The safety engineer must now consider whether this action has any undesirable side effects on the rest of the argument, in addition to recovering the damage already identified.

Step 2: Expressing Recovery Action in Terms of GSN Elements

An examination of the peripheral elements of the safety argument shows that the recovery action of reworking the analysis does not necessarily damage any other element of the argument. However, the search does highlight the assumption **A10 (**shown in Figure 8) that the instructions timings used in the analysis are correct. This assumption must be preserved as the analysis is reworked.

Step 5: Recovering the Damaged Argument

After reworking the timing analysis, the safety engineer is in a position to recover the damaged argument. Working top-down from **G.TIM**, he or she needs to question whether the damaged **G.TIM.STATIC** goal must be restated. For example, if the new results were to show a new worst case response time of 2.9 seconds, **G.TIM.STATIC** would need to be restated accordingly. When **G.TIM.STATIC** has been recovered, the engineer must next examine **G.TIM.STATIC.1** and consider whether this also needs to be restated. It does not, and so **G.TIM.STATIC.1** can also be recovered. **Sn3** must now be examined to see whether it needs to be redefined. In fact **Sn3** must be altered to refer to the *new* timing analysis results.

6 Justification of the Change Process

Using this process *all* potentially impacted items are first highlighted. Amongst all of the *potentially* impacted items there may be some items that a safety engineer will easily be able to confirm are not affected and some that require further impact investigation. Such decisions of 'no-impact' can have a significant influence on whether the full consequences of a change are recognised. In order to maintain confidence in the change process and rather than leaving such decisions undocumented and unsubstantiated it can be useful to annotate the argument with justifications of where 'no-impact' decisions have been made. Together with the annotations of the changes that *were* made to the structure, these annotations of 'no-change' aid future comprehension of the argument and help explain how it has (and has not) be changed through time.

7 Safety Argument Design *for* Change

Having considered a number of change scenarios over various goal structures, we have been able to identify and assess a number of strategies that can help safety arguments to improve their ability to withstand the effects of change. In particular, we have recognised the usefulness of the following two approaches:

- Safety Margins
- Diverse Evidence / Argument

A safety margin is created wherever a sub-goal or solution not only satisfies a parent goal, but also *exceeds* the requirement, thus providing a *safety margin.* By doing this, confidence is increased in the satisfaction of the parent and there is a 'margin for error' if the claims put forward in support of the parent goal are weakened at any future occasion (e.g. when the claim is challenged by operational data).

Figure 10 shows an example use of a safety margin within a goal structure.

In Figure 10 the goal G2 exceeds the requirement set out by G1. The margin acts as a 'crumple zone'. Change can propagate through a goal structure up to G2. The margin between G1 and G2 absorbs the change and prevents further propagation, thus protecting the argument above G1.

Figure 11 shows an example use of a diverse argument within a goal structure.
A diverse argument exists wherever a number of *individually sufficient* claims or evidence are put forward to support a particular parent goal. By doing this, confidence is increased in the satisfaction of the parent. For increased 'robustness' the individual arguments should ideally be based upon *independent* forms of evidence. For example, this could mean:

- Diverse forms of safety analysis and testing information
- Appealing to independent safety mechanisms in the design
- Estimated vs. Historical / Operational data

The greater the diversity achieved between the forms of argument put forward the greater the confidence there will be in the satisfaction of the parent goal. The degree of independence between the argument will reduce the vulnerability of the argument to common mode failures (e.g. if a certain form of evidence is challenged or the effectiveness of a safety mechanism is questioned).

8 Limitations of the Approach

The following are the principal limitations of the approach described in this paper:

- Reliance upon correspondence between safety argument and safety case
- Influence of dependencies external to the safety argument

A brief explanation of each of these limitations is provided here.

The change impact assessment approach described in this paper is couched in terms of a safety argument recorded as a goal structure. The ability of the approach to express accurately and fully the impact of changes on the safety case depends on the degree to which the goal structured safety argument corresponds to the documented safety case. The usefulness of the approach in helping to maintain the safety case document depends on how well the relationship between the goal structure and document is understood. Employing document references with the goal structure (e.g. labelling a goal with the document section where that requirement is expressed) can explicitly draw out such links and improve this situation.

There are dependencies other than those shown in **Error! Reference source not found.** that can exist between the safety case elements of *requirements, evidence* and *context,* e.g. between pieces of safety case evidence. The impact of changes through these dependencies must be resolved before attempting to use a goal structure to assess the impact on the safety argument, e.g. it is necessary to realise that changing the FMEA also affects the FTA before assessing the impact of that change within the safety argument. Goal structures record a subset of the dependencies that exist

between the safety case elements. In order to get a complete model of dependencies between the elements, additional models are required to record the remaining dependencies. For example, evidence to evidence dependencies could be recorded through a data model such as that presented by Wilson, Kelly and McDermid in [7].

9 Tool Support in SAM (Safety Argument Manager)

Support for the change process defined in this paper has been implemented within the SAM (Safety Argument Manager) 4 tool. A screen shot of the SAM tool support for change management is shown in Figure 12. Using the tool it is possible to follow through the steps of damaging and repairing a goal structure. The tool supports the propagation of a change according to the rules defined in section 4.3. As highlighted in section 4.3, the assessment of the impact of a change cannot be performed mechanically (by a tool) as it is an interactive process between the tool and engineer. The role of the tool is to pessimistically prompt the engineer to consider all potential effects of the change. The engineer guides the tool to propagate particular impact paths.

Our experience has been that, although it is possible to follow the through the steps of the impact assessment process by hand, for anything other than trivially simple example, tool support is vital.

10 Conclusions

This paper presents a systematic approach to the management of safety case change. Starting from a goal structured representation of the safety argument, we have shown how it is possible to use the recorded dependencies of the goal structure to follow through the impact of a change and (having decided upon a corrective action or actions) recover from change. Observed successful strategies that can be employed in the production of safety arguments to mitigate the effects of change have been presented. Although there are recognised limitations to the approach presented, the principal benefit is that it provides a structured and systematic approach to reasoning about the effects of change where previously very limited support was available.

11 Appendix: Key to Goal Structuring Notation (GSN) Symbols

Figure 13 provides a key to the Goal Structuring Notation (GSN) symbols used in the examples presented in this paper.

12 Acknowledgements

The authors would like to acknowledge the financial support given by Rolls-Royce plc for the work reported in this paper.

13 References

[1] IEC. 61508 – Functional Safety of Electrical / Electronic / Programmable Electronic Safety-Related Systems. International Electrotechnical Commission, Draft Standard, 1997.

[2] MoD. 00-54 Requirements of Safety Related Electronic Hardware in Defence Equipment. Ministry of Defence, Interim Defence Standard, 1999.

[3] MoD. 00-55 Requirements of Safety Related Software in Defence Equipment. Ministry of Defence, Defence Standard, 1997.

[4] MoD. 00-56 Safety Management Requirements for Defence Systems. Ministry of Defence, Defence Standard, 1996.

[5] HSE. Railway Safety Cases – Railway (Safety Case) Regulations 1994 – Guidance on Regulations. U.K. Health and Safety Executive, HSE Books, 1994.

[6] Kelly T, McDermid J. Safety Case Construction and Reuse Using Patterns. In: Proc. 16th International Conference on Computer Safety and Reliability (SAFECOMP'97). York, 1997.

[7] Wilson S, Kelly T, McDermid J. Safety Case Development: Current Practice, Future Prospects. In: Proc. Safety and Reliability of Software Based Systems - Twelfth Annual CSR Workshop. Bruges, 1997.

[8] Field M, Foulkes R. Project Management (PMT 605), Unit 7: Change Control. Milton Keynes: The Open University, 1987.

[9] Ross R. Software Configuration Management in the Support of Formal Development. PhD Thesis, Software Verification Research Centre, School of Information Technology. Brisbane: University of Queensland, 1997.

[10] JAA. Joint Airworthiness Requirements JAR-E: Engines (Change 8). U.K. Civil Aviation Authority, 1990.

[11] Hogberg L. Shutting Down 5 Reactors: Reasons Why and Lessons Learnt. Nuclear Europe Worldscan 1994;14:42-43.

[12] Clarke A W. Magnox Safety Review: Extending the Life of Britain's Work Horses. Nuclear Energy 1989;28:215-220.

[13] HSE. Safety Assessment Principles for Nuclear Plants. U.K. Health and Safety Executive, HSE Books, 1992.

[12] Bishop P, Bloomfield R, Emmet L, Jones C, Froome P. Adelard Safety Case Development Manual. London: Adelard, 1998.

Vitae

Tim Kelly is a lecturer in safety-critical systems engineering within the Department of Computer Science at the University of York. His research interests include the rigorous justification and development of safety-critical systems (including the development, maintenance and reuse of safety cases) and architectural and pattern-oriented approaches to software design and development. He received an MA in Computer Science from the University of Cambridge and a DPhil in Computer Science (for his research into safety case development) from the University of York. Tim has previously worked within the Rolls-Royce University Technology Centre (UTC) in Systems and Software Engineering and the Rolls-Royce High Integrity Systems and Software Centre.

John McDermid is Professor of Software Engineering at the University of York. He directs the High Integrity Systems Engineering group at the University which is a recognised centre of excellence in safety critical computing for the UK Aerospace industry. He directs the Rolls-Royce University Technology Centre (UTC) in Systems and Software Engineering and the DCSC. He has published 6 books and over 200 papers.

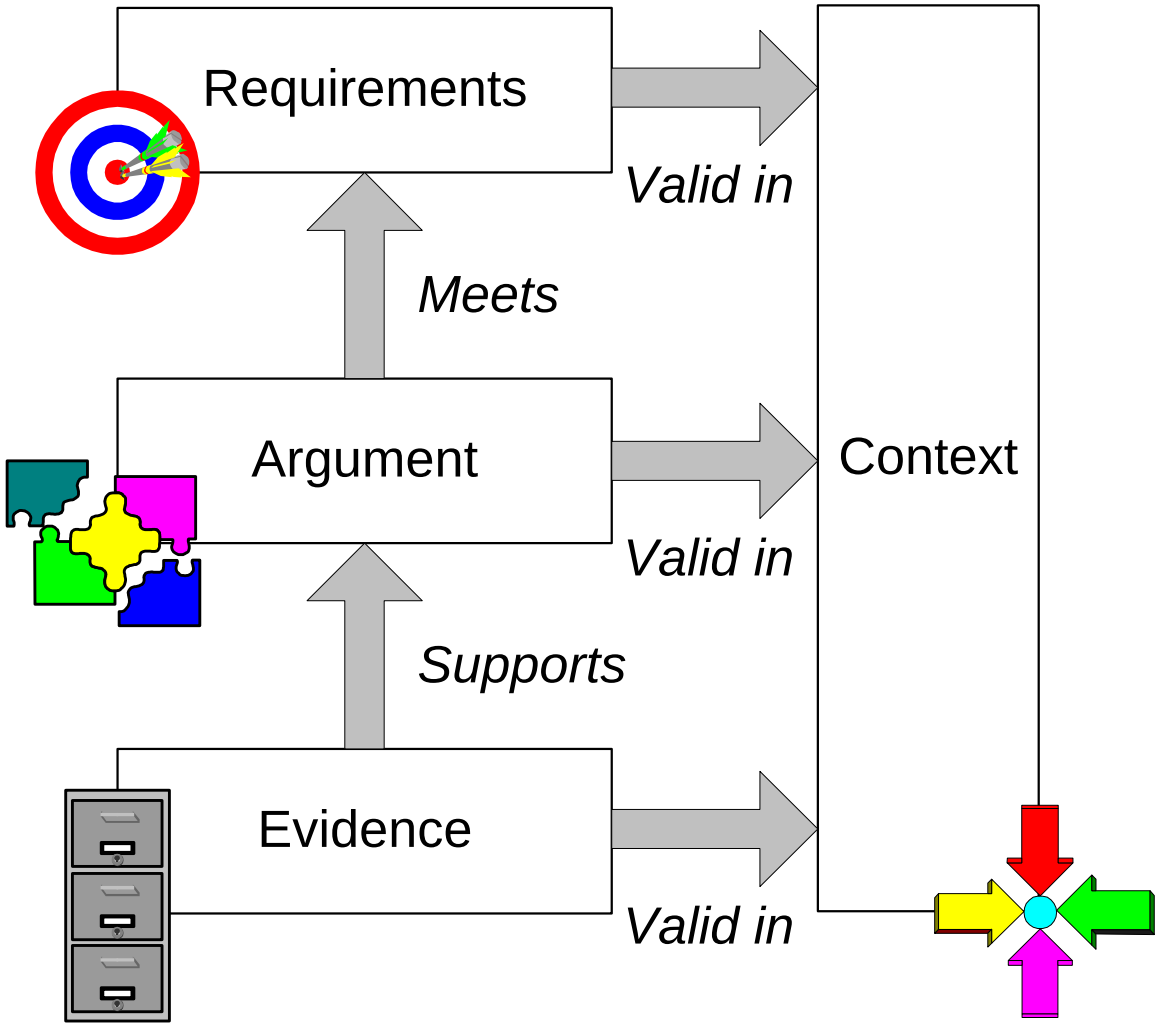

Figure 1 - Dependencies between elements of the Safety Case

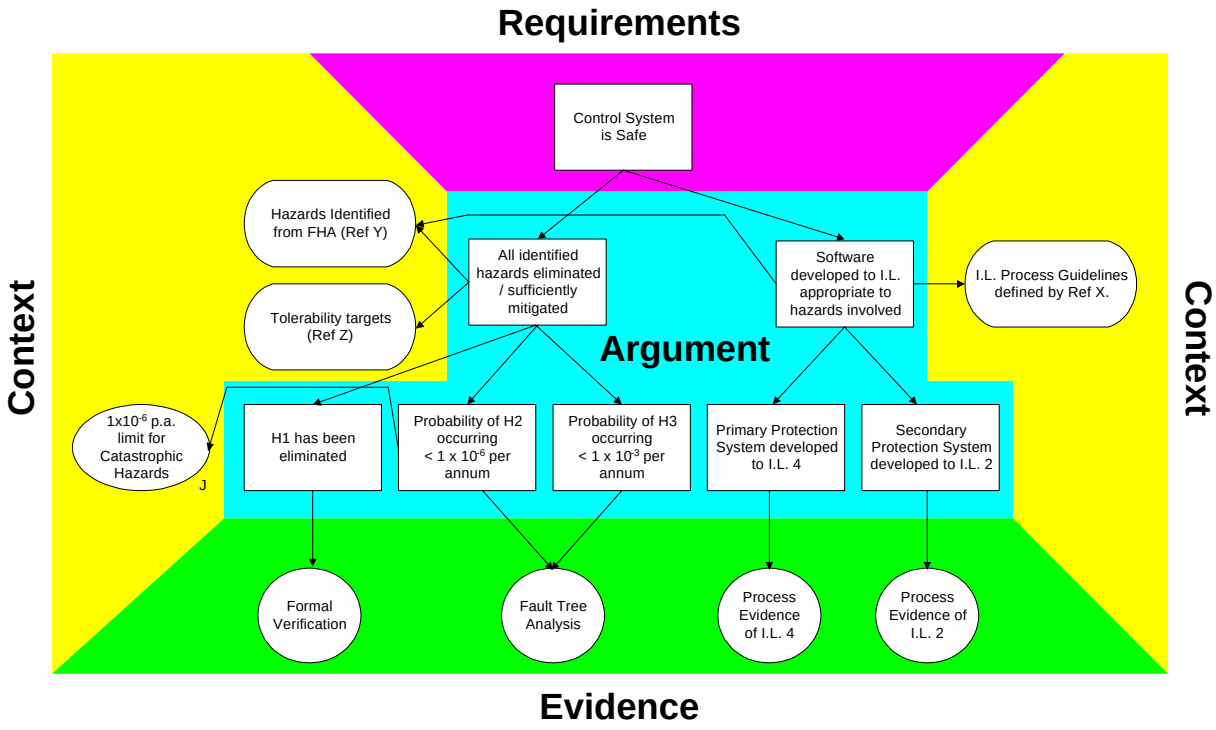

Figure 2 - Relationship between safety case elements and the GSN (Control System Argument)

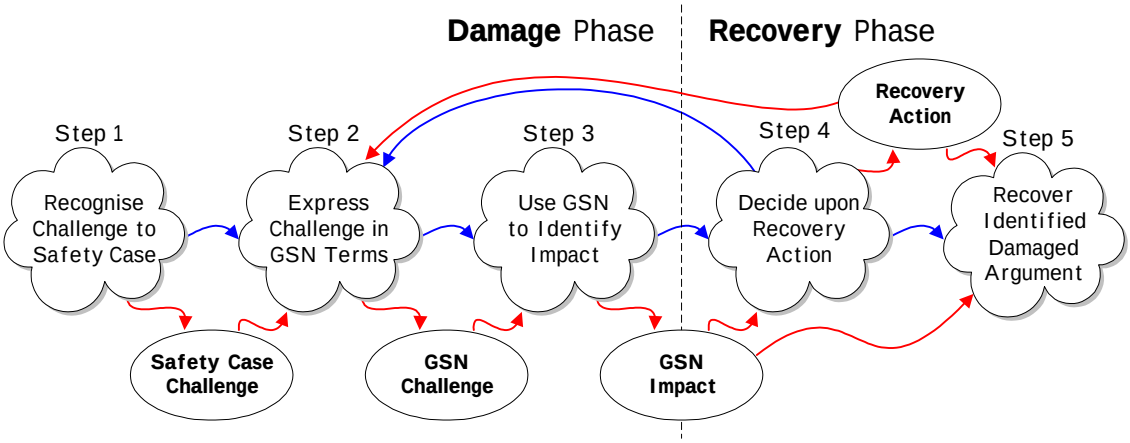

Figure 3 – A Process for Safety Case Change Management

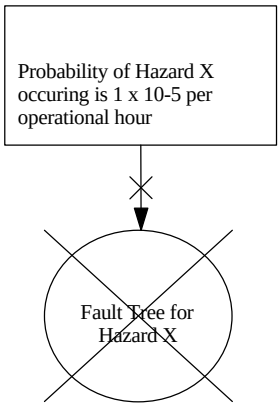

Figure 4 - Evidence Challenge Example

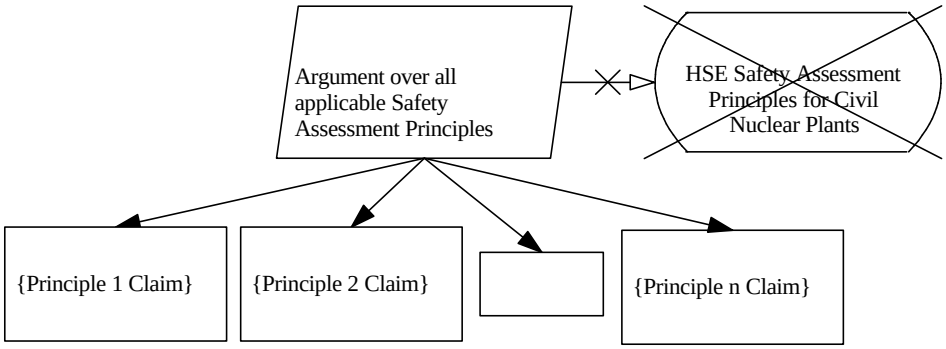

Figure 5 - Requirements Challenge Example

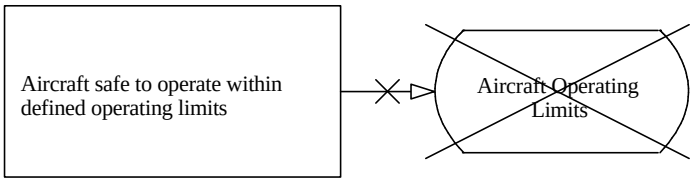

Figure 6 - Context Challenge Example

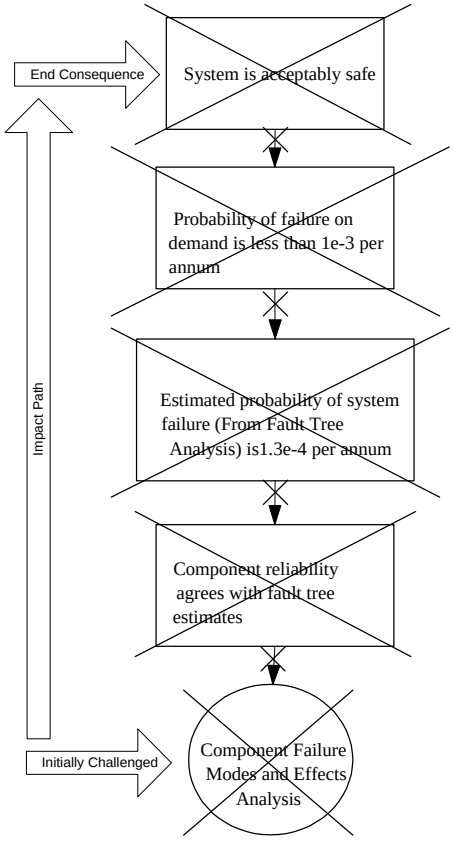

Figure 7 – An Example Impact Path

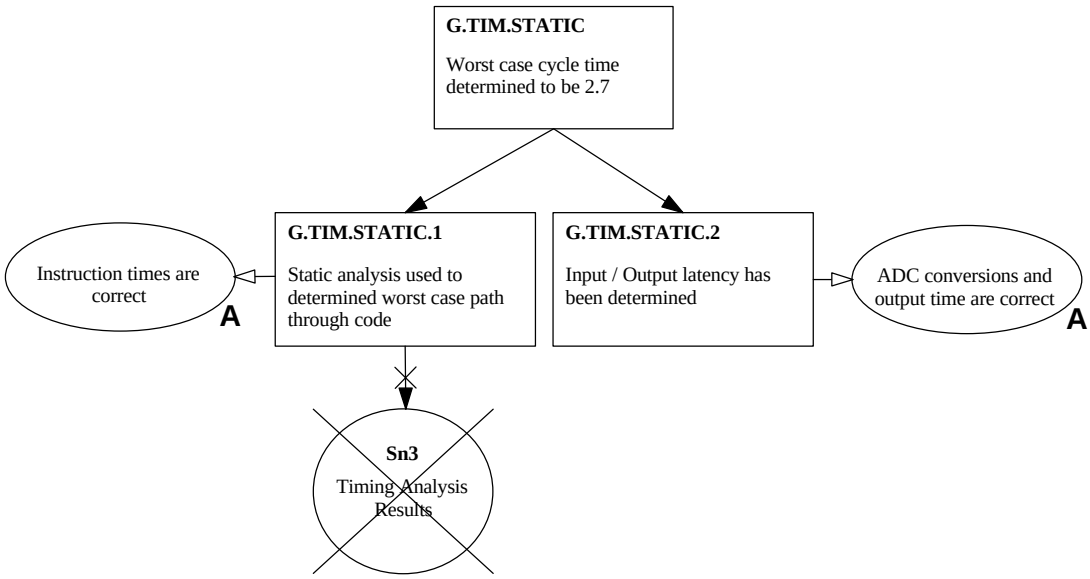

Figure 8 – Challenging the Trip System Timing Analysis Results

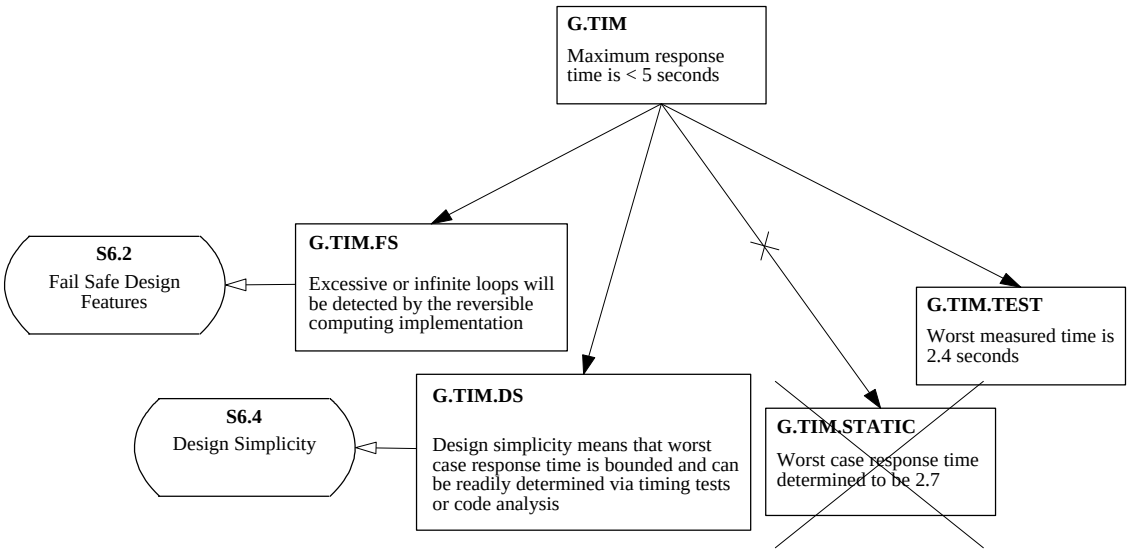

Figure 9 – Challenging the Trip System Timing Analysis Claim

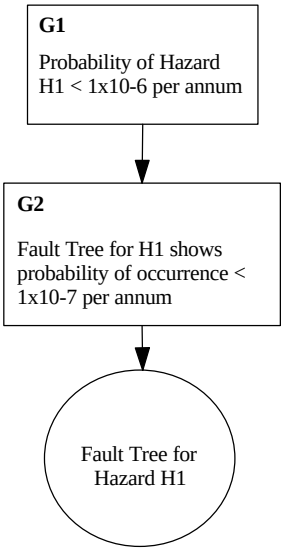

Figure 10 – Use of a Safety Margin within a Goal Structure

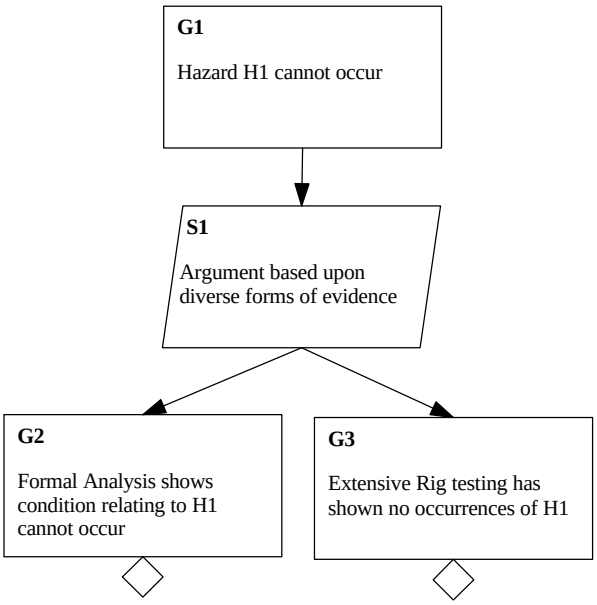

Figure 11 – Use of a Diverse Argument within a Goal Structure

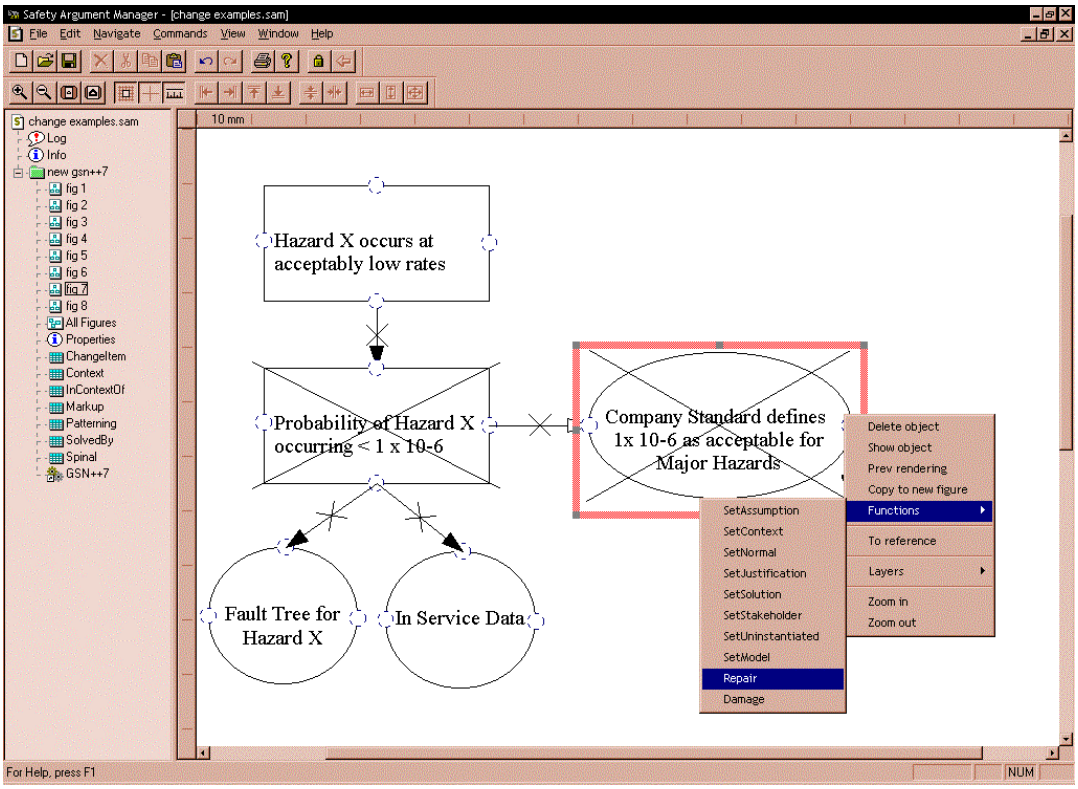

Figure 12 - Tool Support for the Change Process

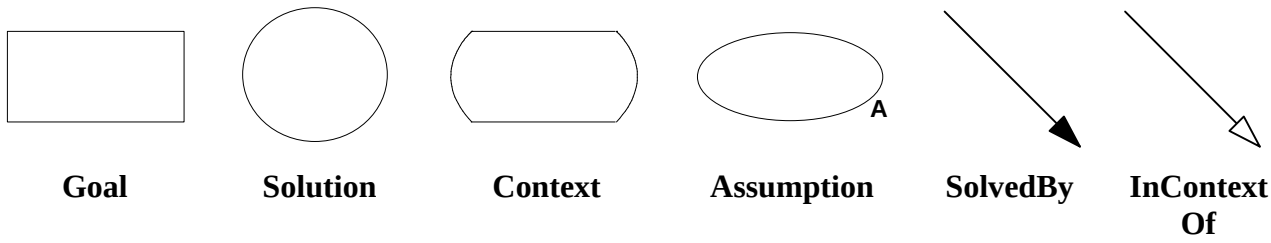

Figure 13 – Goal Structuring Notation (GSN) Symbols