Proceedings Paper:
Jaradat, Omar Tawffeeq Saleem and Bate, Iain orcid.org/0000-0003-2415-8219 (2017) Using Safety Contracts to Guide the Maintenance of Systems and Safety Cases. In: Proceedings - 2017 13th European Dependable Computing Conference, EDCC 2017. 13th European Dependable Computing Conference, EDCC 2017, 04-08 Sep 2017 IEEE , CHE , pp. 95-102.

https://doi.org/10.1109/EDCC.2017.20

Using Safety Contracts to Guide the Maintenance of Systems and Safety Cases

Omar Jaradat*
*School of Innovation, Design, and Engineering
Mälardalen University
Västerås, Sweden
Email: omar.jaradat@mdh.se
Telephone: +46 (21) 101369, Fax: +46 (21) 101460

Iain Bate*†
†Department of Computer Science
University of York
York, United Kingdom
Email: iain.bate@york.ac.uk
Telephone: +44 (1904) 325572, Fax: +44 (1904) 325599

***Abstract*—Changes to safety critical systems are inevitable and can impact the safety confidence about a system as their effects can refute articulated claims about safety or challenge the supporting evidence on which this confidence relies. In order to maintain the safety confidence under changes, system developers need to re-analyse and re-verify the system to generate new valid items of evidence. Identifying the effects of a particular change is a crucial step in any change management process as it enables system developers to estimate the required maintenance effort and reduce the cost by avoiding wider analyses and verification than strictly necessary. This paper presents a sensitivity analysis-based technique which aims at measuring the ability of a system to contain a change (i.e., robustness) without the need to make a major re-design. The proposed technique exploits the safety margins in the budgeted failure probabilities of events in a probabilistic fault-tree analysis to compensate for unaccounted deficits or changes due to maintenance. The technique utilises safety contracts to provide prescriptive data for what is needed to be revisited and verified to maintain system safety when changes happen. We demonstrate the technique on an aircraft wheel braking system.**

***Keywords*—sensitivity analysis, safety case, change impact, failure probabilities, maintenance.**

I. Introduction

System safety is a major property that should be adequately assured during the development process, the deployment and the operation life of safety critical systems. System safety is not assured by chance but rather it must be engineered and evaluated in a systematic manner that might be mandated by safety standards, best practices and experts' recommendations. Hence, safety critical systems are often subject to a compulsory or advisory certification process which often necessitates building the systems in compliance with domain-specific safety standards.

Following the standards' prescriptions leads system developers to generate a lot of artefacts during and after the development of their systems. These artefacts are used as safety evidence to prove that the standards obligations and recommendations were carried out. However, if the generated artefacts are not demonstrated and explained properly, there will be less certainty about their importance which may lead the overall confidence being undermined. Therefore, developers of some safety critical systems construct a *safety case* (also known as *"assurance case"*) to demonstrate the safety aspect of a system by identifying all potential risks and describing, in the light of the available evidence, how these risks have been eliminated or duly mitigated.

Typically, safety critical systems are evolutionary and they are always exposed to both predicted and unpredicted changes during the different stages in their lifecycle. Changes to a system can negatively affect the gained confidence because these changes have the potential to compromise the safety evidence which has been already collected. More clearly, evidence after a change might no longer support the developers' claims because it reflects old development artefacts or old assumptions about operation or the operating environment. In addition, the cost of obtaining certification is significant, with estimates such as 30% of lifecycle costs [5] and 25-75% of development costs [20] are spent on certification [3]. Hence, improper handling of system changes in the safety cases can reflect untrue safety status of the systems and it can also waste significant amount of the certification cost.

Despite clear recommendations to adequately maintain and review the systems and their safety cases by safety standards, existing standards offer little or no advice on how such operations can be carried out [21]. Hence, there is an increasing need for globally acceptable methods and techniques to enable easier change accommodation in safety critical systems without incurring disproportionate cost compared to the size of the change. However, since broader re-verification and re-validation require more effort and time, it is important for any proposal that aims at facilitating system changes to localise the impact of the changes. More specifically, to alleviate the cost of updating both a system and its safety case due to a change, it is crucial to minimise the effects of that change and prevent these effects from propagating into other parts of the system as far as it is practically possible.

In our previous work [12], we introduced a Sensitivity ANalysis for Enabling Safety Argument Maintenance (SANESAM) technique that supports system engineers to accommodate some types of potential changes. We also developed SANESAM+ [9] as a modified version of SANESAM that covers wider variety of changes. The key principle of SANESAM and SANESAM+ is to determine the flexibility (or robustness) of a system to changes using sensitivity analysis. The output is a ranked list of Fault Tree Analysis (FTA) events that system engineers can refine. The result after the refinement is a list of

events that will be, most likely, related to the future changes. We use safety contracts to record the information of the maximum allowed changes to those events without violating the minimum acceptable safety limits. Those contracts can be used as part of later change impact analysis to advise the engineers what to consider and check when changes actually happen. The main contribution of this paper is to propose a new technique through which SANESAM is used to contain (i.e., localise) the potential changes in the smallest possible part of a system. More clearly, we compare the calculated MAFP (Maximum Allowed Failure Probability) of the events with new estimated FP of those events due to a change. If a new estimate FP of an event is $\leq$ MAFP, then the change will not, necessarily, require a considerable system modification, otherwise, it means that there will be a deficit in that FP and more effort should be considered. There could be several ways to respond to the latter case, but some responses might require large planning and massive re-engineering effort. Alternatively, we suggest, in this paper, to use the FP margins of other events to compensate the resultant deficit. The paper uses the aircraft Wheel Braking System (WBS) [1] to illustrate different examples of changes containment.

The rest of the paper is organised as follows: In Section II, we present necessary background information. In Section III, we describe two techniques to facilitate the maintenance of safety cases. We use this description as a basis to introduce a new technique to facilitate the maintenance of safety critical systems and safety cases in Section IV. In Section V, we use the WBS system as an illustrative example. Finally, we conclude and propose potential future works in Section VI.

II. Background and Motivation

A. Safety Case

A safety case is defined as: "*A structured argument, supported by evidence, intended to justify that a system is acceptably safe for a specific application in a specific operating environment*" [22]. Hence, a safety case comprises both safety evidence (e.g. safety analyses, software inspections, or functional tests) and a safety argument explaining that evidence [13]. In order for safety cases to be developed, discussed, challenged, presented and reviewed amongst stakeholders, as well as maintained throughout the product lifecycle, it is necessary that (1) the argument to be clearly structured and (2) items of evidence to be clearly asserted to support the argument [2]. There are several ways to represent safety arguments (e.g., textual, tabular, graphical, etc.). In this paper, we use the Goal Structuring Notation (GSN) [2], which provides a graphical means of communicating (1) safety argument elements, claims (goals), argument logic (strategies), assumptions, context, evidence (solutions), and (2) the relationships between these elements. The principal symbols of the notation are shown in Figure 1 (with example instances of each concept). A goal structure shows how goals are successively broken down into ('solved by') sub-goals until eventually supported by direct reference to evidence. Using the GSN can clarify the argument strategies adopted (i.e., how the premises imply the conclusion), the rationale for the approach (assumptions, justifications) and the context in which goals are stated.

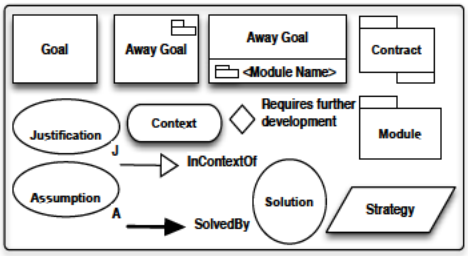

Fig. 1. Notation Keys of the GSN

B. Fault Tree Analysis (FTA)

FTA is a failure analysis method which focuses on one particular undesired event and provides a method for determining causes of this event [1]. In other words, FTA uses abductive reasoning to identify different causes to critical states (from a safety or reliability standpoint). These states might be associated with component hardware failures, human errors, software errors, or any other pertinent events. FTA helps safety engineers to identify plausible causes (i.e., faults) of undesired events [19]. Moreover, FTA is used as a method to achieve Probabilistic Safety Analysis (PSA). More specifically, probability of failure is assigned to each of the failure events based on historical data, and the failure probability of the top event is determined [18].

C. Sensitivity Analysis

Sensitivity analysis can be defined as: "*The study of how uncertainty in the output of a model (numerical or otherwise) can be apportioned to different sources of uncertainty in the model input*" [17]. The analysis helps to establish reasonably acceptable confidence in the model by studying the uncertainties that are often associated with variables in models. Many variables in system analysis or design models represent quantities that are very difficult, or even impossible to measure to a great deal of accuracy [15]. In practice, system developers are usually uncertain about variables in the different system models and they estimate those variables. Sensitivity analysis allows system developers to determine what level of accuracy is necessary for a parameter (variable) to make the model sufficiently useful and valid [4]. In this paper we use the sensitivity analysis to identify the safety argument parts (i.e., sensitive parts) that might require unneeded painstaking work to update with respect to the benefit of a given change. The results of the analysis should be presented in the safety argument so that it is always available up front to get developers' attention.

D. Safety Contracts

In 1969, Hoare introduced the pre- and postcondition technique to describe the connection (dependency) between the execution results (R) of a program (Q) and the values taken by the variables (P) before that program is initiated [7]. Hoare introduced a new notation to describe this connection, as follows:

$$P\ \{Q\}\ R$$

This notation can be interpreted as: "*If the assertion P is true before initiation of a program Q, then the assertion R will be true on its completion*" [7].

In the context of contract-based design, a contract is conceived as an extension to the specification of software component interfaces that specifies preconditions and post-conditions to describe what properties a component can offer once the surrounding environment satisfies one or more related assumption(s).

A contract is said to be a *safety contract* if it guarantees a property that is traceable to a hazard. Contracts have been exploited as a means for helping to manage system changes in a system domain or in its corresponding safety case [8], [14], [6]. In this paper, we use safety contracts to record the dependencies among failure probabilities of FTA's events.

III. SANESAM AND SANESAM+

In this section, we give an overview of SANESAM [12] and SANESAM+ [9]. SANESAM and SANEMSAM+ exploit sensitivity analysis on FTAs to measure the sensitivity of outcome A (e.g., a safety requirement being true) to a change in a parameter B (e.g., the failure probability in a component). The sensitivity is defined as $\Delta B/B$, where ΔB is the smallest change in B that changes A (e.g., the smallest increase in failure probability that makes safety requirement A false). The failure probability values that are attached to FTA's events are considered input parameters to the sensitivity analysis. A sensitive part of a FTA is defined as one or multiple FTA events whose minimum changes (i.e., the smallest increase in its failure probability due to a system change) have the maximal effect on the FTA, where effect means exceeding failure probabilities (reliability targets) to inadmissible levels. A sensitive event is an event whose failure probability value can significantly influence the validity of the FTA once it increases [12], [9].

The key principle of both techniques is to determine, for each component, the allowed range for a certain parameter within which a component may change before it compromises a certain system property (e.g., safety, reliability, etc.). More clearly, the techniques assume the existence of a probabilistic FTA where each event in the tree is specified by a current estimate of failure probability $FP_{Current|event(x)}$. In addition, they assume the existence of the required failure probability for the top event $FP_{Required(Topevent)}$, where the FTA is considered unreliable if:

$FP_{Currentl(Topevent)} > FP_{Required(Topevent)}$ [12].

SANESAM devotes $\Delta FP_{(Topevent)}$ for each event at a time, whereas SANESAM+ distributes it over all of the events. The two techniques use sensitivity analysis to determine the range of failure probability parameter for each event. The steps of SANESAM phase are shown in Figure 2 and described as follows:

Step 1. Apply sensitivity analysis to a probabilistic FTA: In this step the sensitivity analysis is applied to a probabilistic FTA to identify the sensitive events whose minimum changes have the maximal effect on the $FP_{Topevent}$. However, applying this step is not identical for both SANESAM and SANESAM+. Essentially, SANESAM calculates the maximum possible increment to the failure probability parameter of only one event at a time before the top event $FP_{Required(Topevent)}$ is no longer met. On the other hand, SANESAM+ was introduced to provide more freedom by considering multiple events at a time. That is, if multiple events in FTA are expected to change, then SANESAM+ is the one to go. Choosing SANESAM means that the developers accept the assumption that only one event is allowed to change at a time. The difference between the process of SANESAM and SANESAM+ is observed in the way we apply Step 1. One more difference is that Step 2 should be completely neglected while applying SANESAM+ process, the rest of the steps are identical.

Applying Step 1 for SANESAM is done as follows [12]:

i) Find the Minimal Cut Set (MC) in the FTA [16].
ii) Calculate the maximum possible increment to the failure probability parameter of event x before the top event $FP_{Required(Topevent)}$ is no longer met, where $x \in MC$ and $(FP_{Increased|event(x)} - FP_{Current|event(x)}) \nRightarrow FP_{Increased(Topevent)} > FP_{Required(Topevent)}$.
iii) Rank the sensitive events from the most sensitive to the less sensitive. The most sensitive event is the event for which the following formula is the minimum:

$$\frac{FP_{Increased|event(x)} - FP_{Current|event(x)}}{FP_{Current|event(x)}.}$$

Applying Step 1 for SANESAM+ can be summarised as follows [9]:

i) Find $\Delta FP_{(Topevent)}$, where

$$\Delta FP_{(Topevent)} = FP_{Required} - FP_{Current}$$

ii) Distribute $\Delta FP_{(Topevent)}$ over all events in FTA. The distribution can be performed using different equations based on the logic gates in FTA. SANESAM+ steps from 1 to 4 in [9] describe those equations and give examples of how to perform the distribution.

Step 2. Refine the identified sensitive parts with system developers: In this step, the generated list of sensitive events from Step 1 should be discussed by system developers (e.g., safety engineers) as they should choose the sensitive events that are most likely to change. The list can be extended to add any additional events by the developers. Moreover, it is envisaged that some events might be removed from the list or the rank of some of them might change. This step shall not be applied for SANESAM+.

Step 3. Derive safety contracts from FTAs: At least one safety contract should be derived for each event in the list from Step 2. The main objectives of the contracts are to 1) highlight the sensitive events to make them visible up front for developers attention, and 2) to record the dependencies

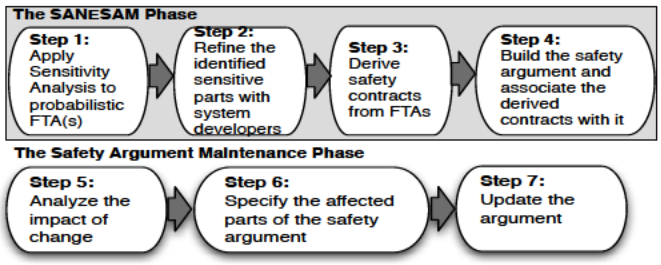

Fig. 2. The process diagram of SANESAM [12]

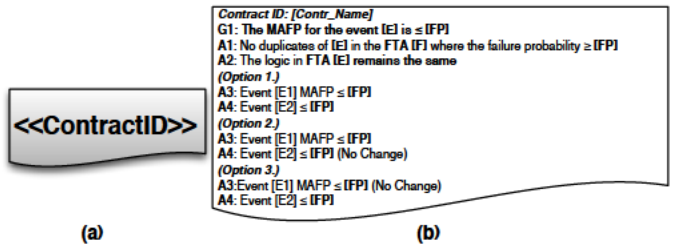

Fig. 3. (a) FTA Safety contract notation, (b) Derived safety contract

between the sensitive events and the other events in the FTA. Hence, if the system is later changed in a way that increases the failure probability of a contracted event where the increased failure probability is still within the defined threshold in the contract, then it can be said that the contract(s) in question still hold (intact) and the change is containable with no further maintenance. The contract(s), however, should be updated to the latest failure probability value. In contrast, if the change causes a bigger increment to the failure probability value than the contract can hold, then the contract is said to be broken and the guaranteed event will no longer meet its reliability target. It is worth noting that the role of safety contracts in SANESAM is to highlight sensitive events, and not to enter new event failure probabilities. We introduce a new notation to FTAs to annotate the contracted events, where every created contract should have a unique identifier, as shown in Figure 3-a. We also create a template to document the derived safety contracts. Figure 3-b shows an instantiation of the contents of one of the derived safety contracts for WBS.

Step 4. Build the safety argument and associate the derived contracts with it: In this step, a safety argument should be built and the derived safety contracts should be associated with the argument elements. Further instructions of how to associate the derived safety contracts with a safety argument are described in [12].

IV. Safety Contracts Driven Maintenance

The way we suggest to cope with some types of changes is to contain their effects in the smallest possible set of events to prevent (or minimise) the ripple of these effects from propagation. In this section, we describe a new technique that enables the containment of certain class of changes in safety critical systems and safety cases. It is worth noting that this technique utilises the same rules by which SANESAM and SANESAM+ calculate the sensitivities and associate them with a safety argument via safety contracts. However, the technique adds additional steps to enable effective usage of the safety margins in a probabilistic FTA. The new technique provides solutions to accommodate a change even if the change broke one or more safety contracts. The only needed input for the process of the technique is a probabilistic FTA. The process comprises 6 steps that can fall into two main phases, before and after introducing a change. The three steps before perfroming a change are similar to the first phase of SANESAM and SANSAM+ as shown in Figure 2. However, we have made some non-substantial changes to some of these steps, where we describe the change to each step when we describe the step itself. Steps 4-6 are novel and they were designed and specified for the new technique.

Steps before performing a change:

Step 1. Apply sensitivity analysis to a probabilistic FTA: This step is performed exactly as instructed in the process of either SANESAM or SANESAM+.

Step 2. Derive safety contracts: In this step, we need to derive safety contracts from FTAs as described in Section III. However, there are two main differences in the derivation of safety contracts in this work. First, the guaranteed MAFPs in the safety contracts are basically the results of either multiplication or summation of multiple children events. Hence, there is no point to derive contracts for basic events in FTA because they simply do not have children events. The second main difference is that the contracts should provide multiple options for developers to measure the tolerance of a change's impact. More clearly, each derived safety contract should assume that only one child event is affected, multiple children events are affected or all of them are affected.

Step 3. Associate the derived contracts with safety arguments: Unlike the same step in SANESAM (in Section III) and to enable more freedom, the proposed technique in this work considers that the construction of safety arguments is not necessarily a part of the process. Hence, we assume the existence of a safety argument no matter how it is represented (e.g., textual, tabular, graphical, etc.). The most important for us is the association itself because this association highlights the suspect elements in the argument to bring them to developers' attention. Typically, there is n-to-m mapping between the events of a FTA and different parts of a safety argument. Hence, a derived contract that guarantees a property, value, range, etc. should be associated with every part that is related to that guarantee in the argument.

Steps after performing a change:

Step 4. Check the ability of FTA to contain greater FP(s) than those already exist: The key principle of this step is to compare the new estimated FP of an affected event with the guaranteed MAFP in the safety contract of that event, or probably in the safety contracts of higher events.

As a quick check, we can determine the MC and calculate the expected FP of the top event taking into consideration the new increased FPs of the affected events and the current FPs (not the MAFP) of the unaffected ones. If the new calculated FP of the top event is $\geq$ MAFP, then the change is not containable and this means that there will be a deficit in the overall FP budget where the response to the change might require re-engineering effort. Dealing with such a situation is beyond the scope of this paper. In contrast, if the new

TABLE I. Rating the Impact of Change

Impact Level	Highlight Colour	Description	Impact on Safety case	
			Argument	Evidence
Low		Change to an event is contained within its safety contract	No change necessary	Might reuse the same evidence
Medium		Change to an event is not contained within its safety contract, however is contained by another higher level safety contract with sufficient margin	No change necessary	Might need new evidence
High		The change is not contained within any of the derived safety contracts and the overall failure target of the system cannot be met	Major impact on the safety argument structure	Need new evidence

calculated FP of the top event is $\leq$ MAFP, this means that the change's impact is containable somewhere in the FTA, but we need to know which safety contract contains it. To do that, the new estimated FP of an impacted event should be checked against the guaranteed MAFP in the safety contract of that event, where it is containable iff it is $\leq$ MAFP. If the change's effect (i.e., difference between the new estimated FP and the MAFP) is not containable in the safety contract of the impacted event, then the safety contract of the ancestor event should be investigated as whether or not it can contain it. If the change's effect still cannot be contained by the ancestor, safety contracts in one more level up should be investigated and so on and so forth until a safety contract contains it. Once the contract which contains the change's effect is identified, all associated claims with this contract together with their supporting arguments and evidence should be highlighted as suspect.

Step 5. Re-balance the FPs of the FTA's events as a preparation for future changes. If any event has received a change that necessitates increasing its failure probability where the increment is still within the MAFP threshold in its safety contract, then it can be said that the safety contract in question still holds (intact) and the change is containable with no further significant maintenance. However, we need to re-balance the FPs of the FTA's events after accommodating a change to prepare for further accommodation(s) of future potential changes. Hence, we need to find $\Delta FP_{(Topevent)}$ which is the difference between the required FP and the new FP of the top event after containing a change. The current FPs of all unaffected events together with the new FPs of the affected events are used to calculate $FP_{New(Topevent)}$ based on the determined MC from Step 4. The resultant $FP_{New(Topevent)}$ is subtracted from $FP_{Required(Topevent)}$.

The calculated $\Delta FP_{(Topevent)}$ might be equal to 0 or it can be an insignificant fraction which is not worth further effort. In this case, Step 1-ii should be omitted (i.e., no need for distribution) and we only need to update the safety contracts as described in the next step.

Step 6. Update the affected safety contracts with the new FPs: The contracts should be updated by the latest failure probability value(s) after containing a change. This step can be seen as Step 2, the difference is that we do not derive new contracts, but we rather update the ones we derived earlier.

In order to enhance the visibility of a change's impact on the system design and safety case, we highlight the parts of the system design and the elements of the safety case that are related to the affected events in FTA. Three levels of impact based on the impact propagation within FTA were defined, namely, *Low, Medium* and *High*. Table I categories the changes and suggest highlighting/representing them with different colours based on the rating of changes' impact.

V. Illustrative Example

We apply our proposed technique described in Section IV to the Wheel Braking System (WBS) in which we assume three different change request scenarios and evaluate their impacts on the safety case. For the sake of both simplicity and space, we, in this section, summarise the key points of the application and provide its results. The detailed application of the technique is available in a separate technical report [11].

A. Wheel Braking System (WBS): System Description

The WBS is described in Appendix L of Aerospace Recommended Practice ARP-4761 [1] for safety assessment processes. The main function of the system is to provide wheel braking as commanded by the pilot when the aircraft is on the ground. The system is composed of three main parts: 1) Computer-based part which is called the Brake System Control Unit *(BSCU)*, 2) *Hydraulic* part, and 3) *Mechanical* part. The *BSCU* is internally redundant and consists of two channels, *BSCU System 1 and 2* (*BSCU* is the box in the grey background in Figure 4). Each channel consists of two components: *Monitor* and *Command*. *BSCU System 1 and 2* receive the same pedal position inputs, and both calculate the command value. The two command values are individually monitored by the *Monitor 1 and 2*. Subsequently, values are compared and if they do not agree, a failure is reported. The results of both *Monitors* and the compared values are provided to the *Validity Monitor*. A failure reported by either system in the *BSCU* will cause that system to disable its outputs and set the *Validity Monitor* to invalid with no effect on the mode of operation of the whole system. However, if both monitors report failure, the *BSCU* is deemed inoperable and is shut down [10]. Figure 4 shows high-level view of the *BSCU* implementation. More details about the *BSCU* implementation can be found in ARP-4761 [1]. Figure 5 shows the "Loss of Braking Commands" probabilistic FTA (the original FTA is without the grey shapes) whilst Figure 6 shows GSN fragment of the WBS safety argument.

B. Safety Contracts Driven Maintenance: An Example

Since we have now all the required inputs for our technique (i.e., probabilistic FTA in addition to top event MAFP and current FP), we can start applying the *Steps before performing a change* (Steps 1-3 in Section IV), as follows:

Step.1: Apply sensitivity analysis: In this example we apply SANESAM+:

i) Find $\Delta FP_{(Topevent)}$. $\Delta FP_{(Topevent)}$ = **3.15E-05**.
ii) Distribute $\Delta FP_{(Topevent)}$ over all events. Figure 5 shows the distribution result for each event in the grey boxes.

Step 2. Derive safety contracts: After calculating the MAFPs for all of the events, a safety contract was derived for each non basic event. The template of the safety contract in Figure 3-a is used to represent the derived safety

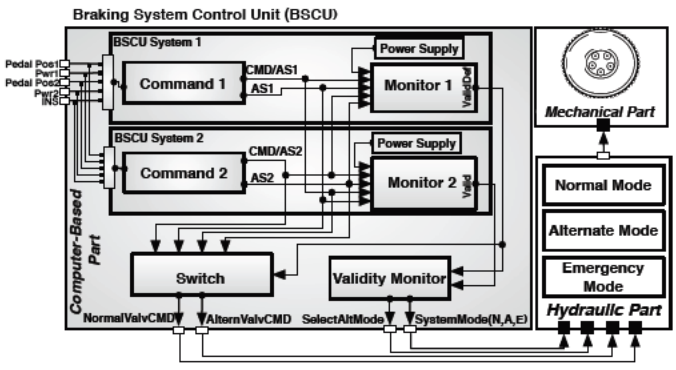

Fig. 4. A high-level view of the WBS [12]

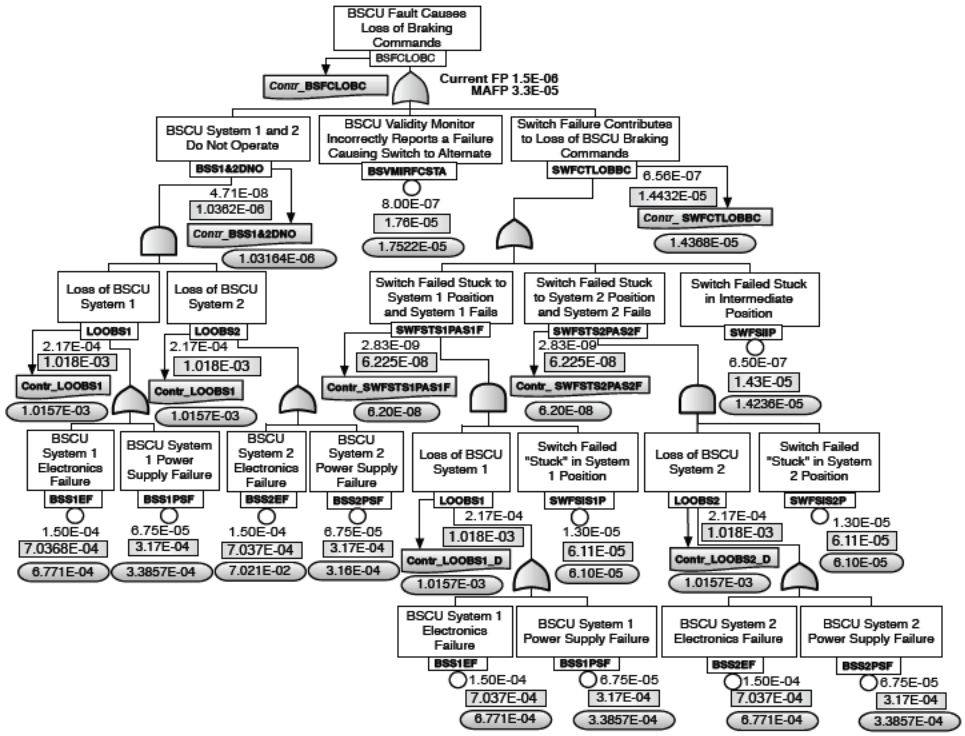

Fig. 5. Loss of Braking Commands FTA [1]

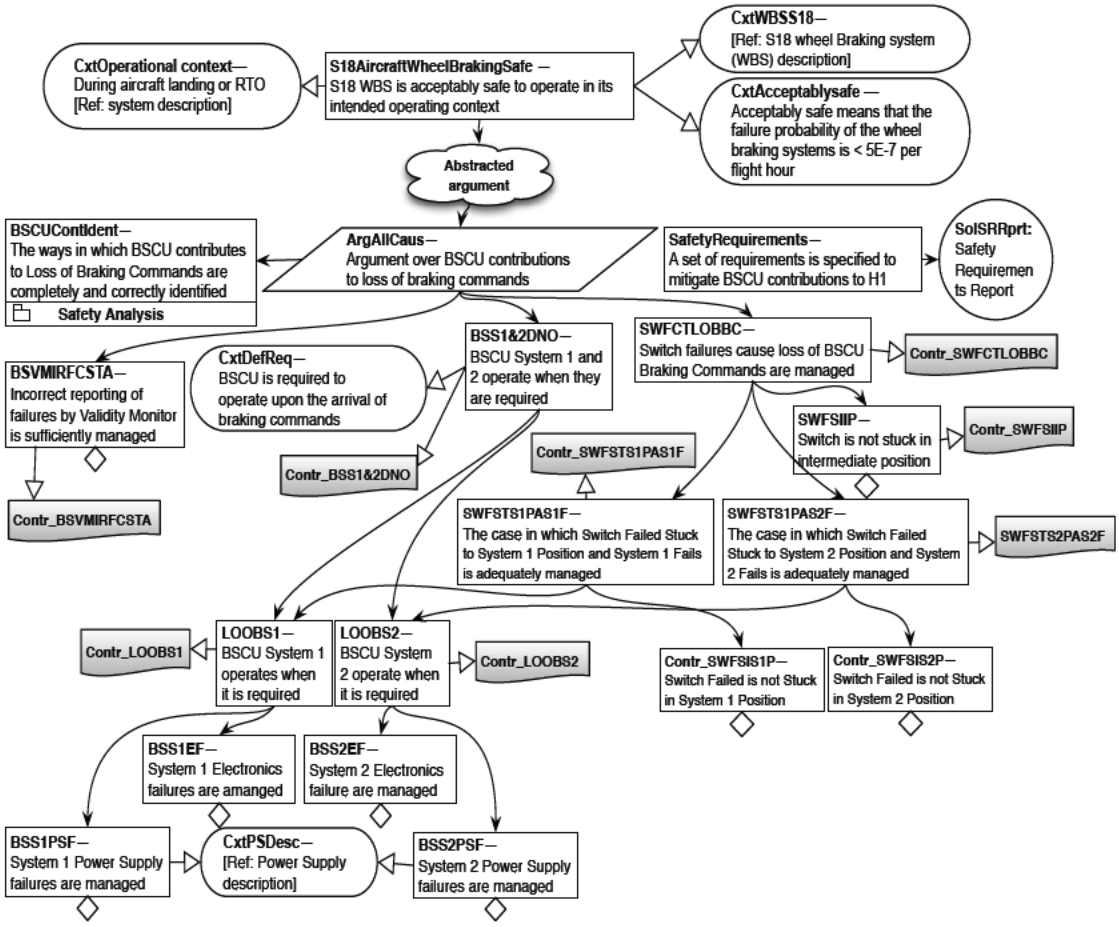

Fig. 6. Safety argument fragment for WBS

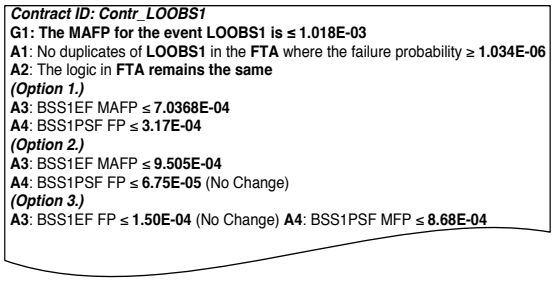

Fig. 7. A derived safety contract

contracts. Also the contract notation in Figure 3-b is used to annotate the contracted events. Each contract considers multiple assumptions options based on the number of the children events. Figure 5 shows the derived contracts using the contract notations in grey. Figure 7 provides an internal view of the *Contr_LOOBS1* contract which is derived for the event *LOOBS1* as an example.

Step 3. Associate the derived contracts with the safety argument: In this example, we use a GSN argument fragment to show the association. Figure 6 shows how the derived safety contracts from FTA are associated with a safety argument fragment for WBS using the proposed contract notation in Figure 3-a. We do not want to affect the way GSN is being produced but we want to bring additional information for developers' attention. It is worth mentioning that a safety contract should be associated with all claims that are related to the event which the contract is derived for. For example, the safety contract *Contr_SWFSTS2PAS2F* should be associated with any articulated claims about the state when Switch Failed Stuck to System 2 Position and System 2 Fails.

Now, let us assume some change scenarios that can resemble real life change requests.

Change request scenario (1): The WBS developers have received a change request from the senior management asking to replace the current installed power supplies in BSCU 1 and 2 by a different model. Based on the provided product specifications by the new power supplies manufacturer, the FP of that model is *3.00E-04*, which means that it is less reliable than the FP of the current model in use (i.e., *6.75E-05*). Subsequently, step 4 should be followed to assess the impact of the given change scenario.

Step 4. Check the ability of FTA to contain greater FP(s) than those already exist: As a quick check, we want to update the FPs of the affected events based on the new given FPs and calculate the new FP of the top event. The new FP of the top event after the replacement is **1.646E-06** and since *1.646E-06* $<$ *3.3E-05*, the increments to the FPs of *BSS1PSF* and *BSS2PSF* are tolerated (i.e., containable) in the FTA but the question is: ***Where can they be contained?***

To answer this question we need to specify the affected contracts by the change and check whether or not they still hold in the light of the new FP. The change request will affect four contracts, namely, *Contr_LOOBS1*, *Contr_LOOBS2*, *Contr_LOOBS1_D* and *Contr_LOOBS2_D*. Each derived contract contains different options in the assumptions list (as shown in Figure 7). We choose *(Option 1.)* in the four contracts and check if the MAFPs of *BSS1PSF* or *BSS2PSF* can contain the new FP. Since *3.16E-04* (MAFP) $>$ *3.00E-04* (new FP), the increments to *BSS1PSF* and *BSS2PSF* are contained in the four contracts and they still hold. This implies that replacing the power supply is rated as a GREEN change which means (according to Table I) that there is no need to make any structural changes to the system design nor the safety argument. However, a manual check for the argument is still needed to replace the information of the old power supply with new valid information. For example, the description which the context *CxtPSDesc* refers to (in Figure 6) is out of date and should be replaced by the new power supply description.

Step 5. Re-balance the FPs of the FTA's events as a preparation for future changes: The reduction in the margins of the *BSS1PSF* and *BSS2PSF* FPs should be shared by all of the events in the FTA. That is, all current FPs should contribute to make up the contraction of *BSS1PSF* and *BSS2PSF* FP margins due to the power supply replacement, as follows:

1) Find $\Delta FP_{(Topevent)}$ which is the difference between the required FP (i.e., *3.30E-05*) and the new $FP_{Current(Topevent)}$ after containing the change which we have determined earlier (i.e., *1.646E-06*). $\Delta FP_{(Topevent)}$= *3.136E-05*.
2) Repeat Step 1-ii (i.e., the SANESAM+ approach which we have already mentioned under Step 1 in this Subsection) to distribute *3.136E-05* over all FPs' margins in the FTA. The grey squashed rectangles in Figure 5 represent the new MAFPs after the change.

Step 6. Update the affected safety contracts: Since new MAFPs have been calculated for all of the events, all derived contracts should be updated to reflect the new MAFP values.

Change request scenario (2): This scenario is similar to scenario (1). The only difference though is the FP value of the new power supply model, which is in this case equals to *5.00E-03* and thus it has less reliability than the current FP and even lesser than the one from the first scenario. As a quick check, the FP of the top event after introducing the change is ***2.8106E-05***, which means that it is $< FP_{Required(Topevent)}$ and thus the change is tolerable. By applying the same steps we did in the previous scenario we will find out that *Contr_BSS1&2DNO* is the contract which contains the change.

Change request scenario (3): This scenario is similar to the previously discussed scenarios (2) and (3). The difference here is that the FP value of the new power supply model is *6.00E-03*, which means that it has less reliability than the current FP and it is the least reliable in this the three scenarios. The new calculated FP for the top event of this scenario is ***3.9432E-05*** and it is $>$ *3.3E-05* (the MAFP for the top event). That is, the resultant change effects due to replacing the power supply by this specific model is not containable and the entire FTA is going to be impacted. Hence, the WBS cannot meet its current safety requirements without considering major structural changes or updates.

Figure 8 shows a high level view of the change effects in the FTA that is caused by replacing the power supply in the three discussed change scenarios. The figure also shows how the safety contracts are used to highlight the affected parts in the WBS design and the safety argument. More

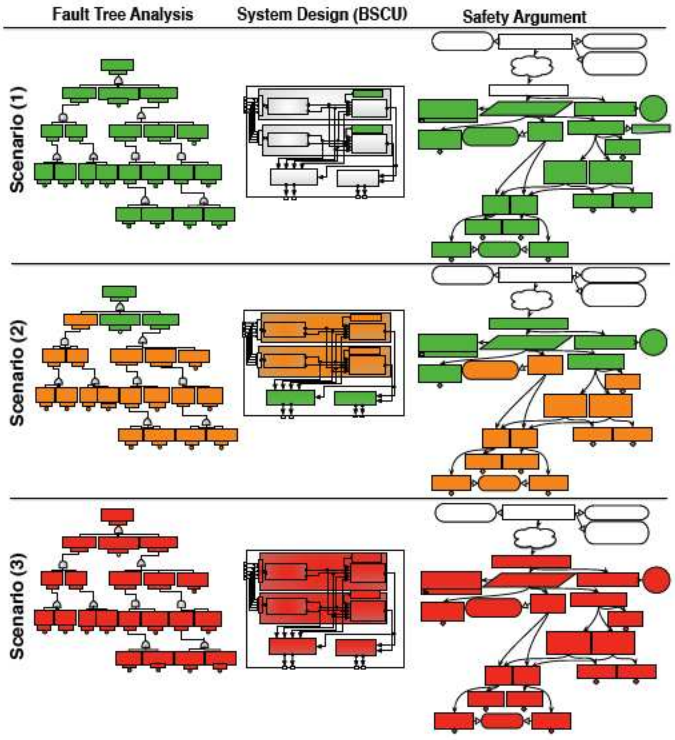

Fig. 8. The effect of change on the FTA, system design and the safety argument: An overview of the three scenarios

detailed description on how the technique is applied to the three scenarios is available in [11].

VI. Conclusion and Future Work

In our previous works [9], [12] we introduced SANESAM and SANESAM+ as techniques to facilitate the maintenance of safety cases using safety contracts. In this paper, we use the key principle of SANESAM and SANESAM+ to introduce a new technique that can save huge efforts in re-verification or re-certification due to some design changes. The technique can serve as a first impact analysis layer that helps system's developers to estimate the size of effort needed to accommodate a design change. The technique can also guide the developers to avoid massive re-engineering efforts when it is not really needed. Although the technique can be effective in maintaining safety systems and safety cases, the scope of the changes addressed by it may seem limited in the general maintenance scenario. However, these types of changes are the most critical from a safety perspective and they are worth making the emphasis. Future work will focus on considering different properties other than failure probabilities (e.g., timing) in order to consider additional types of changes. In addition, development of an automation tool is considered as a potential direction. We also intend to perform a case study to validate both the feasibility and efficacy of the technique.

Acknowledgment

This work has been partially supported by the Swedish Foundation for Strategic Research (SSF) (through SYNOPSIS and FiC Projects) and the EU-ECSEL (through SafeCOP project). We thank Sasikumar Punnekkat for his fruitful help and comments.

References

[1] SAE ARP4761 Guidelines and Methods for Conducting the Safety Assessment Process on Civil Airborne Systems and Equipment, 1996.

[2] Goal Structuring Notation working group, November 2011.

[3] I. Bate, H. Hansson, and S. Punnekkat. Better, faster, cheaper, and safer too – is this really possible? In *Proceedings of the 17th IEEE International Conference on Emerging Technologies for Factory automation*, 2012.

[4] L. Breierova and M. Choudhari. An introduction to sensitivity analysis. Technical report, Massachusetts Institute of Technology (MIT), 1996.

[5] R. Cleaveland. Formal certification of aerospace embedded software. In *National Workshop on Aviation Software Systems: Design for Certifiably Dependable Systems*, 2006.

[6] P. Graydon and I. Bate. The nature and content of safety contracts: Challenges and suggestions for a way forward. In *Proceedings of the 20th IEEE Pacific Rim International Symposium on Dependable Computing (PRDC)*, November 2014.

[7] C. A. R. Hoare. An axiomatic basis for computer programming. *Commun. ACM*, 12(10):576–580, Oct. 1969.

[8] J. L. Fenn, R. Hawkins, P. J. Williams, T. Kelly, M. G. Banner, Y. Oakshott. The who, where, how, why and when of modular and incremental certification. In *Proceedings of the 2nd IET International Conference on System Safety*, pages 135–140. IET, 2007.

[9] O. Jaradat and I. Bate. Deriving hierarchical safety contracts. In *The 21st IEEE Pacific Rim International Symposium on Dependable Computing (PRDC 2015)*, November 2015.

[10] O. Lisagor, M. Pretzer, C. Seguin, D. J. Pumfrey, F. Iwu, and T. Peikenkamp. Towards safety analysis of highly integrated technologically heterogeneous systems – a domain-based approach for modelling system failure logic. In *The 24th International System Safety Conference (ISSC)*, Albuquerque, USA, 2006.

[11] O. Jaradat and I. Bate. Using safety contracts to guide the maintenance of systems and safety cases: An example. Technical report, Mäalardalen University, http://www.es.mdh.se/publications/4726-, April 2017.

[12] O. Jaradat, I. Bate, and S. Punnekkat. Using sensitivity analysis to facilitate the maintenance of safety cases. In *Proceedings of the 20th International Conference on Reliable Software Technologies (Ada-Europe)*, pages 162–176, June 2015.

[13] O. Jaradat, P. Graydon and I. Bate. An approach to maintaining safety case evidence after a system change. In *Proceedings of the 10th European Dependable Computing Conference (EDCC)*, UK, 2014.

[14] P. Conmy, J. Carlson, R. Land, S. Björnander, O. Bridal, I. Bate. Extension of techniques for modular safety arguments. Deliverable d2.3.1, technical report, Safety certification of software-intensive systems with reusable components (SafeCer), 2012.

[15] J. Pate, K. R. Edlin, and K. I. Kawano. Sensitivity analysis of hardware-in-the-loop (HWIL) simulation systems. In *Proceedings of Modelling and Simulation*. ACTA, May 2005.

[16] M. Rausand and A. Høyland. *System Reliability Theory: Models, Statistical Methods and Applications*. Wiley-Interscience, NJ, 2004.

[17] A. Saltelli. *Global sensitivity analysis: the primer*. John Wiley, 2008.

[18] P. Shankar, J. Mathieson, R. Ramachandran, J. D. Summers, and G. M. Mocko. Can design evaluation tools predict/prevent change propagation? In *Proceedings of Tools and Methods of Competitive Engineering*, 2012.

[19] M. Stamatelatos, W. Vesely, J. Dugan, J. Fragola, J. Minarick, and J. Railsback. Fault Tree Handbook with Aerospace Applications. Handbook, National Aeronautics and Space Administration, 2002.

[20] N. R. Storey. *Safety Critical Computer Systems*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1996.

[21] T. Kelly and J. McDermid. A systematic approach to safety case maintenance. In *Proceedings of the Computer Safety, Reliability and Security*, volume 1698 of *Lecture Notes in Computer Science*, pages 13–26. Springer Berlin Heidelberg, 1999.

[22] U.K. Ministry of Defence. *00-56 Defence Standard — Safety Management Requirements for Defence Systems*, December 1996.