Incremental Safety Assessment: Enabling the Comparison of Safety Analysis Results

O. Lisagor; Department of Computer Science, The University of York; York, UK

M. Bozzano; Fondazione Bruno Kessler; Trento, Italy

M. Bretschneider; Airbus Operations GmbH; Hamburg, Germany

T. P. Kelly; Department of Computer Science, The University of York; York, UK

Keywords: Automated Safety Analysis, Incremental Safety Assessment, Methodology, Modelling,
Safety Analysis, System Safety Engineering

Abstract

In our 2008 ISSC paper we have reviewed some pragmatic challenges of incremental system safety assessment and outlined requirements for a framework to identify, track and manage design and analysis changes introduced between iterations. This ‘follow-up’ paper reports our recent work to address these issues.

The paper defines an intuitive “lightweight refinement” relation between results of the analyses (minimal cut sets) obtained at different assessment iterations and/or development milestones. The relation is used, however, not to prove conformance between successive assessments, but rather to identify emerging significant differences that require consideration and justification by the safety engineers.

Naïve refinement is further extended with the notion of “equivalence mapping” to allow comparison of minimal cut sets defined over different vocabularies of failures. This extension permits comparison of analyses of models of the same system defined by different engineers, at different levels of detail, using different methods and tools. The extension also enables engineers to control the granularity of comparison. Concepts presented throughout the paper are illustrated on an example of an electrical power distribution system of a hypothetical aircraft.

Introduction

The potential benefits of an iterative and incremental safety assessment that starts at the earliest stages of the design process are widely recognized today. For example, Aerospace Recommended Practices 4754 and 4761 (SAE, 1996a, b) call for a Preliminary System Safety Assessment (PSSA) to be “an iterative analysis embedded within the overall development” and an “ongoing process starting in the early phases of the development” (SAE, 1996a).

In our earlier paper (Lisagor and Kelly, 2008) we have observed that, despite this long established principle, adequate approaches to incremental safety assessment have not yet emerged. Traditional and novel, e.g. model-based, safety assessment methods typically view evolution of the analysis results between assessment iterations from the restricted perspective of (gradual) decomposition. This view is too narrow for the purpose of practical incremental assessment as it only permits adding more structure – but not new substantive information or behavior – to the models that underlie safety analysis. We showed that the notions of refinement (de Roever and Engelhardt; 1998, Woodcock and Davies, 1996), that permit adding of the new information to reduce non-determinism and increase fidelity, may provide a more suitable basis for the comparison of results of subsequent assessment iterations. However, our paper argued that standard refinement relations, as used by the formal methods community, may need to be adapted to the safety assessment context, are too strong for a practical application and must be ‘embedded’ in a more general framework for identifying changes of the analysis results that both are significant and cannot be considered an improvement.

Changes to the safety analyses may be a result of changes *to* the design of the system or a better understanding *of* the design (and/or safety requirements) by the safety engineers. Furthermore, safety engineers routinely simplify analysis of the system at earlier iterations of the assessment to enable more timely feedback to the development process. As a result the violations of refinement (or a similar relation) must be expected in the context of the incremental safety assessment and cannot be treated as an ‘exception’. An example of the change that clearly does not observe any reasonable notion of refinement can be found in an illustrative example presented in the Appendix L of the ARP 4761 document (SAE, 1996b; Lisagor and Kelly, 2008).

In this paper we present a 'lightweight refinement' relation defined to facilitate incremental safety assessment in general and identification of significant changes between analysis iterations in particular. In the first two sections we formally define a simplified relation and discuss its application to a hypothetical evolution scenario of an electrical power distribution system. We then present definition of a more robust relation that does not rely on analysis results being defined over a common vocabulary of failures. The paper concludes with the discussion of other possible applications of the 'lightweight refinement' and an overview of some related work. The work presented in this paper is being carried out as part of a collaborative European MISSA project[1].

Naïve Relation

We define 'lightweight refinement' as a relation between two sets of Minimal Cut Sets (MCSs). It is assumed that these two sets are obtained from two models (or fault trees) related to the same system by analysis with respect to the same system-level failure condition (or, more broadly, a hazard). The two models can relate to the same system design proposal or to two iterations of design. For brevity of exposition we refer to these sets of MCSs as "abstract" (obtained from an earlier model) and "concrete" (obtained from a model that emerges later in the development process). To start, we further assume that both abstract and concrete MCSs are expressed over the same vocabulary of basic events (failures); this assumption is revoked later in the paper.

Informally, we are seeking a relation that would stipulate that concrete results are no worse than the abstract ones. This means that for every concrete MCS we should be able to find at least one abstract MCS such that the former contains the latter. In other words, to preserve 'lightweight refinement,' MCSs can be removed or expanded by later analysis iterations; however, neither the reduction of MCSs nor introduction of entirely new cut sets is permissible.

Formal Definition: To formally define the 'lightweight refinement' relation, let $\boldsymbol{R_A}$ and $\boldsymbol{R_C}$ denote abstract and concrete results respectively. Further, let $\boldsymbol{V_F}$ denote a vocabulary of (i.e. a set of all possible) failures. According to the explicit assumption introduced earlier:

$$R_A, R_C \subseteq \wp(V_F), \tag{1}$$

Where $\boldsymbol{\wp(V_F)}$ denotes a powerset of the vocabulary (i.e. all possible combinations of failures). Finally, let $\boldsymbol{M_C}$ be a concrete MCS:

$$M_C \in R_C \tag{2}$$

We can then define a relation between a single concrete MCS, $\boldsymbol{M_C}$, and (an entire set of) abstract results $\boldsymbol{R_A}$ which we call '*Enabled*':

$$Enabled(M_C, R_A) \equiv \exists M_A \in R_A \bullet M_A \subseteq M_C \tag{3}$$

In other words, a concrete MCS is said to be 'enabled' if it can be matched to an identical or 'worse' MCS in the set of abstract results. The 'lightweight refinement', ***REF***, merely requires that every concrete MCS is enabled:

$$REF(R_C, R_A) \equiv \forall M_C \in R_C \bullet Enabled(M_C, R_A) \tag{4}$$

Comparison Tool: As was mentioned in the previous section we do not expect that the 'lightweight refinement' will totally hold in the context of incremental modelling and safety assessment. Instead, the relation permits us to identify MCSs that exhibit new behavior that has not been considered and 'signed-off' by the safety engineers at earlier iterations of the analysis. Such MCSs simply violate the above relation.

To automate the search process we have implemented a comparison tool. In its most simple mode of use, the tool takes two analysis result files – identified as abstract and concrete – and generates a file that contains only those concrete MCSs that violate the 'lightweight refinement'. The comparison tool is compatible with output formats of all analysis tools currently used within the MISSA Project. Formally, if we denote the set of MCSs contained in the violations file as $\boldsymbol{T}$ and, for brevity, the 'exclusive or' operator as $\oplus$ symbol, the comparison tool guarantees that:

[1] More Integrated Systems Safety Assessment: http://www.missa-fp7.eu/

$$(T \subseteq R_C) \wedge (\forall M_C \in R_C \bullet M_C \in T \oplus Enabled(M_C, R_A)) \qquad (5)$$

Illustration: To illustrate the concepts of 'lightweight refinement' and violations consider the simple example of Table 1. The left column of the table lists abstract MCSs and the right – concrete results. The vocabulary of failures in this case is *{a, b, c, d, e, f, g}*. The concrete MCSs *{a, d}*, *{a, b, c}* and *{b, d, e}* observe refinement as they can be matched to 'enabling' abstract cut sets *{a}*, *{b, c}* and *{b, d, e}* respectively. However, two concrete cut sets – *{g}* and *{b, d, f}* – cannot be matched to any 'smaller' abstract results and will therefore appear in the violations file.

Table 1 – Example of Abstract and Concrete Cut Sets Subjected to Comparison

Abstract Cut Sets (F_A)	Concrete Cut Sets (F_C)
{a} {b, c} {c, d} {b, d, e} {b, e, f} {d, e, f}	{g} {a, d} {a, b, c} {b, d, e} {b, d, f}

Electrical Power Distribution System Case Study

To illustrate the application of the 'lightweight refinement' and the associated comparison tool in the context of the incremental safety assessment, we use a simplified Electrical Power Distribution System (EPDS) of the hypothetical aircraft. The architecture of the system – shown in Figure 1 – contains six busbars that are key (output) interfaces of the system. The busbars can be seen either as two triplets (corresponding to the AC and DC outputs of the system) or as three pairs (denoting three notional 'sides': Side 1, Side 2 and Essential Side). The system is ultimately powered by three AC generators: two – denoted as *Gen 1* and *Gen 2* – driven directly by aircraft engines and an emergency generator – *Gen E* – powered by the aircraft hydraulic system. The conversion from AC to DC is performed by three transformers (*TR 1*, *TR 2* and *TR ESS*) – one on each EPDS 'side'. The three sides of the system are connected by three cross feed lines – shown as horizontal connections in Figure 1; for brevity we refer to them as the upper AC cross-feed, lower AC cross-feed and DC cross-feed. Cross-feed lines provide the system with a significant degree of redundancy in terms of possible power transfer paths from generators (and transformers) to any busbar. The exact configuration of the system is determined by the states of 14 contactors (each – either closed or open).

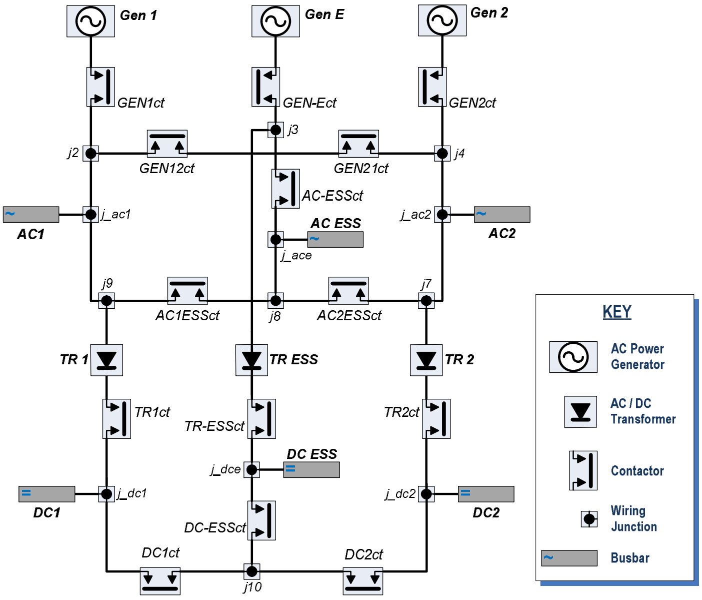


Figure 1 – Schematic of the Physical Architecture of the EPDS

Hierarchy of EPDS Models: To reconstruct a possible incremental process we define a hierarchy of nine models of the system. Each model is constructed under Failure Logic Modelling methodology and specified in AltaRica language (Arnold *et al*, 2000). Since the 'lightweight refinement' relation is defined over sets of Minimal Cut Sets, details of the modelling methodology and language fall outside the scope of this paper. We refer readers to (Lisagor *et al*, 2006) for the methodology overview and (Bieber *et al*, 2002) for illustration of modelling safety-related behavior in a dataflow-centred dialect of AltaRica that we use.

The EPDS models are analyzed by the Cecilia OCAS tool, developed by Dassault Aviation. The analysis tool generates combinations of failures (minimal cut sets) that are necessary and sufficient for 'causing' a particular condition of interest (such as, in terms of EPDS, a loss of power provision via a particular busbar or a combination of busbars). We assume that each generator can suffer from two kinds of spontaneous failures – leading to a loss of any power provision or provision of insufficient power. The same principle applies to the transformers. Contactors may also fail in two ways leaving them stuck in either closed or open position (and, in either case, unresponsive to further control commands). Finally, busbars and wiring junctions can fail to propagate any power.

Having described the 'ground rules' that underlie all EPDS models we turn to the model evolution steps. The first model of the hierarchy contains no reconfiguration information. This means that a busbar is considered to be generating electrical power, provided it has not itself failed and there exists a healthy 'transfer path' between the busbar and any functional generator. The path is considered to be healthy if it relies only on 'unfailed' junctions and contactors that are not stuck in an 'open' position. Analysis of such models allows investigating whether the architecture of the physical electrical network is sufficiently robust and whether it may in principle provide adequate basis for further design.

In the second model we start investigating the implications of different modes of operation. In particular we recognize that, following some failure scenarios, the EPDS should be operated in the Emergency mode. In this mode the Emergency Generator is used to power two essential busbars. To protect this 'power provision of last resort' all non-essential busbars and transformers and two 'normal' generators will be disconnected from the network. Further protection of the Emergency Generator is provided by prohibiting its use outside the Emergency Mode (so in all non-emergency modes the power is provided by *Gen 1* and/or *Gen 2* only). Therefore, a busbar is considered powered in this model if it has not itself failed and:

a) a healthy path exists from the busbar to an operational primary generator, or
b) the busbar is marked as essential and can be connected by a healthy path through the essential 'side' of EPDS to the emergency generator.

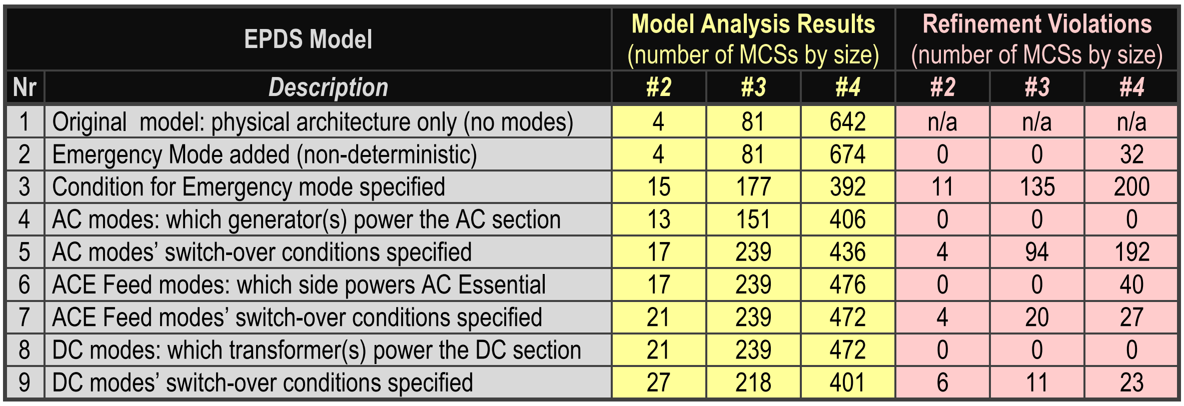
Non-essential busbars can also be inadvertently powered in the Emergency Mode due to failures of contactors (in a closed position) that establish accidental paths to the emergency generator (such as failure of *AC1ESSct* potentially leading to *AC1* being powered in the emergency mode). This model, however, specifies no concrete switch-over condition for the Emergency Mode. In terms of model analysis it means that, to be considered a minimal cut set, a combination of failures must lead to the system-level failure condition in both normal and emergency modes. For example, a combination of failures of junctions *j_ac1*, *j_ac2* and *j_ace* will appear in analysis results for the failure condition of "*loss of power provision on all AC busbars*"; in contrast, failures of *j_ac1*, *j_ac2* and *j8* will not be considered an MCS for this condition since *AC ESS* busbar can still be powered in the emergency mode. The third model of the hierarchy removes this non-determinism by specifying that the emergency mode is entered if, and only if, both non-essential AC busbars are unable to provide power.

We repeat this pattern in the remainder of the EPDS model hierarchy. So the fourth model introduces AC modes that determine which primary generator(s) – i.e. *Gen 1*, *Gen 2* or both – are used to provide power to the network. In the fifth model the switch-over conditions between these modes are specified. The sixth model introduces a constraint that the lower AC cross-feed line shall *not* be used to connect Side 1 to Side 2. It also introduces two alternative modes that determine which side should power *AC ESS* busbar. The next model specifies the reconfiguration conditions for these modes. Finally, eighth and ninth models introduce modes related to the power distribution paths in the DC section of the system and determine their switch-over conditions (respectively).

Model Analyses and Refinement Violations: Overall the hierarchy reflects a possible evolution path of the EPDS design where non-determinism is gradually reduced by design decisions. These decisions, however, not only partition the behavior of the system into modes of operation they also incrementally constrain the behavior. This means that the results of the safety analysis of a design proposal are not necessarily an improvement of the results at the previous iteration. We use the ‘lightweight refinement’ relation to identify the “safety cost” (if any) of design decisions taken between iterations. In particular, each EPDS model is first analyzed with respect to a number of system-level failure conditions of interest. In this paper we consider a single failure condition of “*total loss of power provision on essential busbars*”. The left side of Table 2 summarises the results of analyses by presenting a number of identified MCSs of sizes 2, 3 and 4 for each of the nine EPDS models (the analysis identified no single points of failure for this condition).

Second, analysis results for each model are compared with the results obtained from the preceding model in the hierarchy (in each such comparison the former is considered as concrete results and the later – as abstract). The violations of the ‘lightweight refinement’ found in each such comparison are summarised in the three rightmost columns of the Table 2.

Table 2 – Results of EPDS Models’ Analyses and their Comparison



EPDS Model		Model Analysis Results (number of MCSs by size)			Refinement Violations (number of MCSs by size)		
Nr	Description	#2	#3	#4	#2	#3	#4
1	Original model: physical architecture only (no modes)	4	81	642	n/a	n/a	n/a
2	Emergency Mode added (non-deterministic)	4	81	674	0	0	32
3	Condition for Emergency mode specified	15	177	392	11	135	200
4	AC modes: which generator(s) power the AC section	13	151	406	0	0	0
5	AC modes’ switch-over conditions specified	17	239	436	4	94	192
6	ACE Feed modes: which side powers AC Essential	17	239	476	0	0	40
7	ACE Feed modes’ switch-over conditions specified	21	239	472	4	20	27
8	DC modes: which transformer(s) power the DC section	21	239	472	0	0	0
9	DC modes’ switch-over conditions specified	27	218	401	6	11	23

Even at the level of this summary, some of the results of comparison reported above appear non-trivial. For example, the last EPDS model yields a smaller number of MCSs of sizes three and four than – the preceding (8th) model. Nonetheless, some of these cut sets violate the refinement relation and indicate significant new safety implications of design decisions taken between iterations.

An in-substance review of the actual cut sets that violate the refinement highlights some interesting ‘side effects’ of the design decisions. For instance, we have stated above that the only ‘novelty’ introduced by the second model in the EPDS hierarchy was identification of the Emergency Mode. With the declared purpose of this mode of operation being protection of the essential power provision capability, we were at first surprised that this model yielded 32 *new* minimal cut sets for this failure condition. Review of these cut sets, has revealed the reason. The definition of the emergency mode has constrained power distribution paths from the Emergency Generator (*Gen E*) only to the one through the ‘essential side’ of the system. Therefore, the path to the *DC ESS* busbar via transformer *TR 1* and junctions *j4*, *j8*, *j9* and *j10* (as well as a symmetrical path via *j4*, *j8*, *j7*, *TR 2* and *j10*) that was available is the first (mode-free) model is no longer permissible in the second model (as it would clearly require activation of a non-essential transformer and would lead to the ‘emergency’ power being distributed by at least one non-essential DC busbar). All 32 violations of the refinement can be attributed to this effect of the design decision.

Overall, we found that the comparison process based on the ‘lightweight refinement’ relation is often capable of significantly reducing the volume of the analysis results that require substantial review and, thus, can facilitate prioritisation of safety engineers’ efforts. At the same time, safety implications of design decisions, highlighted by the refinement checking, could be otherwise easily ‘hidden’ by an apparent reduction in a number of MCSs.

Dissimilar Failure Vocabularies and a More Robust Relation

Whilst already yielding significant results, the refinement relation introduced in the previous sections is nevertheless naïve in that it relies on the assumption that a vocabulary of failures remains stable between assessment iterations. This assumption will only hold in a small number of cases. First, it is unlikely to hold if abstract and concrete models of the system are defined by different safety engineers or using different modelling tools. In such context it is doubtful that naming conventions for system components and their failures will be strictly maintained between iterations. Therefore, the abstract results will need to be translated into the new concrete vocabulary (or vice versa) before automated search for violations of refinement can commence.

Second, and more important, between assessment iterations a model of the system can undergo significant transformations that affect the structure (architecture) of the model rather than being limited to strictly 'local' changes to the behavior of individual components. Examples of such transformations (previously discussed in more detail in (Lisagor and Kelly, 2008)) include hierarchical decomposition and failure [mode] refinement.

Failure Dictionary: To permit effective comparison of results in this more general context a more robust refinement relation must be defined over two sets of MCSs, $\boldsymbol{R_A}$ and $\boldsymbol{R_C}$, that are themselves defined over two different vocabularies $\boldsymbol{V_A}$ and $\boldsymbol{V_C}$ respectively; thus revising equation 1 into:

$$R_A \subseteq \wp(V_A); \quad R_C \subseteq \wp(V_C) \tag{6}$$

To perform any meaningful comparison of these two sets of results it is necessary to provide a "dictionary" – a set of mappings between elements of the two vocabularies. Informally, a dictionary contains all permissible interpretation of the abstract failure in terms of concrete failures. We make no assumptions about the shape of the dictionary: abstract failures may be mapped to more than one (alternative) concrete failure, two (or more) abstract failures can be mapped to the same concrete failure or an abstract failure may remain unmapped. Formally, a dictionary – $\boldsymbol{D}$ – is defined as a set of ordered pairs, such that the first element of each pair belongs to a concrete vocabulary and the second – to the abstract one. In other words, a dictionary is simply a subset of a Cartesian product of two vocabularies:

$$D \subseteq V_C \times V_A \tag{7}$$

Revised Lightweight Refinement Definition: In this new context, the refinement is concerned with the ability to find, for every concrete MCS, an abstract cut set that has a valid interpretation which is better or identical to the original concrete cut set. We therefore need to redefine the *Enabled* relation that we have specified in equation 4. In particular, the simple inclusion operator is no longer adequate and, instead, we need to stipulate a similar relation between the concrete MCS and an interpretation of an abstract cut set. However, since two abstract failures can be mapped by the dictionary to the same concrete failure, such interpretation is, in the general case, a multiset. To avoid introducing cumbersome multiset operators we first define (recursively) a condition ***Match*** over a concrete and abstract MCSs ($\boldsymbol{M_C}$ and $\boldsymbol{M_A}$ respectively) as well as the dictionary ($\boldsymbol{D}$) that adapts the inclusion relation of equation 3 to the new context.

$$\begin{gathered} Match(M_C, M_A, D) \equiv \\ (M_A = \varnothing) \vee (\exists f_A \in M_A, f_C \in M_C \bullet \langle f_C, f_A \rangle \in D \wedge Match(M_C \setminus \{f_C\}, M_A \setminus \{f_A\}, D)) \end{gathered} \tag{8}$$

The ***Match*** relation always holds for an empty abstract MCS $\boldsymbol{M_A}$, regardless of the contents of the concrete cut set and the dictionary (the base case of induction). Otherwise, the relation holds if it is possible to remove a pair of concrete and abstract failures from the respective MCSs such that the pair is mapped in the dictionary and ***Match*** holds over remaining MCSs (inductive step).

With the ***Match*** condition being defined, the ***Enabled*** and 'lightweight refinement' (***REF***) relations can be respectively redefined as:

$$Enabled(M_C, R_A, D) \equiv \exists M_A \in R_A \bullet Match(M_C, M_A, D) \tag{9}$$

$$REF(R_C, R_A, D) \equiv \forall M_C \in R_C \bullet Enabled(M_C, R_A, D) \tag{10}$$

It is important to stress that we have revised the signature of the lightweight refinement relation. In the revised version, a refinement can be said to hold only with respect to a particular dictionary. Consequently, adequacy of this dictionary can affect completeness and correctness of the results of the incremental assessment and, thus, must be assured by the safety engineers.

The comparison tool, mentioned in the earlier section of this paper, is in fact based on this revised definition of the lightweight refinement.

Illustrations: We first illustrate the revised relation on a simple example of Table 3. The left column of the table contains abstract MCSs (defined over a vocabulary of *{a, b, c, d, e}*), right column – concrete MCSs (with the corresponding vocabulary being *{s, t, w, x, y, z}*) and the middle column contains a dictionary between the two vocabularies. In this example the comparison tool would match concrete cut sets *{w, y}* and *{s, w, x}* to the same abstract *{b, c}* set, whilst *{t, w, z}* can be matched to either *{a}* or *{c, d}*. The remaining *{s}* and *{s, t, x}* both lead to the violation of the 'lightweight refinement' relation.

Table 3 – Example of Abstract and Concrete Cut Sets over Different Vocabularies (Linked by a Dictionary)

Abstract Cut Sets (F_A)	Dictionary (D)	Concrete Cut Sets (F_C)
{a} {b, c} {c, d} {b, d, e}	<t, d> <w, c> <x, b> <y, b> <z, a>	{s} {w, y} {t, w, z} {s, w, x} {s, t, x}

For a further illustration we briefly return to the EPDS. All models of the evolution scenario that we have previously discussed have been constructed by the same author and using the same modelling approach. To investigate the application of the 'lightweight refinement' relation in a more realistic context we have been provided with an alternative model of the same system defined by our collaborators at ONERA (French aerospace research lab). Whilst still specified in the same language (AltaRica), this model implements reconfiguration rules that are more close to the ones implemented on the aircraft than the hypothesised rules that were used in our earlier hierarchy. The analysis of this new model with respect to the same failure condition as before (total loss of essential power supply) yields 6, 85 and 83 minimal cut sets of sizes 2, 3 and 4 respectively. Our task was therefore to compare those results (considered "concrete") with the results of the analysis of the last model presented in Table 2 before (considered "abstract").

However, the ONERA model and our model cannot be compared directly as they are not defined over the same vocabulary of failures. First, the naming conventions for some components and most failures are subtly different between the two models. These discrepancies are trivially resolved by the dictionary.

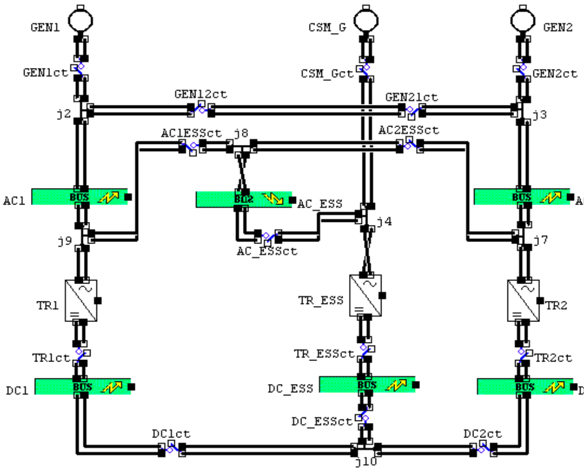


Figure 2 – Architecture of the EPDS Model Constructed by ONERA (Cecilia OCAS Representation)

Second, and more important, the architecture of the ONERA model (Figure 2) is slightly different from the one we have assumed. In particular, whilst we have modelled each busbar and its respective connector to the electrical network as two separate components (both capable of failure), in the ONERA model these are represented as a single busbar component (so that the network both feeds- and is fed from- the busbar directly). To resolve this discrepancy we map failures of each busbar and each associated junction (e.g. *j_ac1*, *j_dc1*, *j_ace*, etc) of our model to the failure of the corresponding busbar in the ONERA model (yielding a many-to-one mappings in the dictionary). Finally, both models have a number of wiring junctions (for example *j2* and *j4*). Whilst in our model these junctions can fail spontaneously, the ONERA model assumes them to be not capable of failing. Consequently, failures of these junctions do not appear in the dictionary.

The comparison of analysis results with respect to the mapping described above yields only 12 violations of refinement – all of size 3. All of these violations can be traced to subtle changes in reconfiguration rules between the two models whose impact would be difficult to deduce without the ability to compare results automatically.

Other Applications of 'Lightweight Refinement'

As was stated in the previous section neither our definition of the 'lightweight refinement' nor the implementation of the analysis tool presupposes any particular shape of the dictionary between two vocabularies of failures. Whilst this flexibility comes at a 'cost' of requiring safety engineers to verify the adequacy of the dictionary, it provides some opportunities for less trivial application of the 'machinery' we have described.

First, the dictionary can be used to control the granularity of comparison. Indeed, the results compared in the previous section have been obtained from models with broadly identical architecture and comparable failures of components. Suppose, however, that for some type of component – say, a transformer – failures identified in two models could not be directly related. The straightforward application of the refinement – even in presence of a 'natural' dictionary – is likely to yield a high number of violations. Whilst all of those violations should be reviewed, safety engineers could benefit from some form of interactive support for rationalising the newly identified behavior and identifying particular 'patterns' of violations. An example of a dictionary that may facilitate the review of the differences between analysis results in this context, can map every failure of a transformer in one model to each failure of the corresponding transformer in the other model. The resultant comparison will essentially disregard differences between individual failures considering any such failure as simply "*a failure of transformer 'N'* ". We have in fact used this comparison strategy to rationalise violations of refinement yielded by EPDS models that introduce short-circuit conditions and their effects (which extended the hierarchy presented in this paper).

Similarly, if the system (or model) architecture undergoes significant changes between assessment iterations, safety engineers may wish to perform even less detailed comparison. For instance, it is possible to map every failure of every transformer in the abstract model to every failure of every transformer in the concrete model. The resultant comparison will be then performed at a level of "*a failure of a transformer*". Extending this strategy 'ad absurdum', a dictionary that maps every failure of one model to every failure of another model will yield a comparison that is equivalent to merely checking whether the model- or system- change between assessment iterations has maintained the minimal size (cardinality) of MCSs intact.

Second, in our experiments with the comparison of analysis results, we have observed that the review of refinement violations is surprisingly effective in facilitating identification of errors in the models. We have found a number of errors and inconsistencies in models that we have previously considered 'error free' on the basis of review of component characterisations, selective (ad hoc) simulation and review of analysis results. In general, the validation or – more broadly – assurance of the adequacy of the models used for safety analysis is increasingly recognized as one of the key challenges for both traditional and model-based safety assessment approaches (Lisagor *et al*, 2010, Manion, 2007). A comparison of models based on the lightweight refinement relation can provide one practical way of increasing confidence in the 'quality' of the models. We are planning to report on this application of the relation in a later publication.

Related Concepts

Our notion of 'lightweight refinement' bears some (superficial) similarity with the concept of (untimed) 'trace refinement' (Schneider, 1993) of the Communicating Sequential Processes (CSP) and similar formalisms. Both notions of refinement treat models as 'black boxes' and are defined in terms of models response to stimuli. Similarly, within the scope of this comparison, our 'dictionary' has some similarity with the CSP's notion of alphabet or event renaming (Schneider, 2000). However, our relationship is significantly weaker and more selective than any notion of refinement in the formal software engineering methods community.

Within the broad model-based safety assessment context there are also some similarities with the work reported by Bernard *et al* on refinement of AltaRica models (Bernard *et al*, 2008). The 'reverse' direction of the simulation relation that underlies their notion of refinement is compatible with the orientation of the inclusion relation in our definition of refinement (i.e. stipulating that abstract MCSs must be contained within the concrete ones rather than vice-versa). However, our relation is, again, significantly weaker and – as was stated before – does not require access to the state-space of the model. We believe that both qualities are beneficial in the context of practical incremental safety assessment. Firstly, treatment of models as 'black boxes' permits a meaningful comparison of more dissimilar models. Secondly, based on our experience in application of the 'lightweight refinement', we believe that a stronger relation is likely to yield qualitatively larger numbers of violations. Many of those will have no significance to the safety assessment of the [individual] system and may, therefore, undermine the purpose of comparison. Saying this, Bernard *et al*'s relation is developed specifically for the context of compositional multi-system safety assessment of an aircraft. More specifically, their relation is used to allow safety engineers to reduce computational complexity of the models (making the product of the composition tractable by the current analysis tools) whilst preserving the correctness of the overall analysis. In contrast, our relation is likely to be both too weak and too selective to be applicable in this context.

Conclusions

Having previously outlined some challenges for the incremental safety assessment and argued that such incremental approach must rely on a framework for identification and tracing changes in assessment results, in this paper we have presented an approach to comparing two sets of analysis results. We have formally defined a 'lightweight refinement' relation that underlies our approach to comparing sets of minimal cut sets. Informally, the refinement relation stipulates that every MCS obtained by later analysis (e.g. analysis of a concrete, more detailed, model) must be equivalent to- or an improvement of- some MCS found in the earlier analysis. This simple inclusion operator over sets of MCSs and its orientation form the conceptual core of the relation. However, our relation has been adapted for comparing minimal cut sets defined over dissimilar 'failure vocabularies'. This means that comparison can be performed across results obtained from the models that are specified using different languages and modelling approaches. Similarly, models do not have to be defined at the same level of granularity and their structure does not need to be identical. Saying this, our notion of dictionary is simplified. We are planning to consider more complex relationships between abstract and concrete failures, which cannot be represented as simple ordered pairs, in the future research.

We have stressed that in practical context we do not expect the 'lightweight refinement' relation to hold throughout the evolution of system design and its models. Instead we use the relation specifically to identify all violations. Such violations encapsulate significant changes that the system and its model have undergone between iterations from the restricted perspective of a particular system-level failure condition (or a hazard). Consequently, it allows assessing the 'safety impact' of design and modelling decisions whilst reducing the review burden of safety engineers and allowing prioritizing the investment of effort. We have illustrated an incremental modelling and safety analysis process on the example of an Electrical Power Distribution System of a hypothetical aircraft. Overall, our early experiments in application of the relation have yielded encouraging results. In most cases the number of MCSs requiring attention is significantly reduced by automated comparison (sometimes achieving an order of magnitude reduction). We also observed that comparison of analysis results may provide added benefit in terms of model validation and identification of inadequacies and errors.

Acknowledgements

The work reported in this paper has been undertaken within *More Integrated Systems Safety Assessment (MISSA)* project funded by the European Commission under the Framework 7 programme (grant reference: ACP7-GA-2008-212088). We gratefully acknowledge research, development and evaluation contribution by our project colleagues. In particular, *Ben Warner (Atego)* has developed a mapping tool – and intuitive GUI application that simplifies dictionary definition. *Dr Pierre Bieber (ONERA)* has provided us with an alternative hierarchy of the EPDS models (one of which is discussed in the paper) and their descriptions.

References

1. Arnold, A., G. Point, A. Griffault, and A. Rauzy, The AltaRica Formalism for Describing Concurrent Systems. Fundamenta Informaticae, 2000. **34**: p. 109-124.
2. Bernard, R., et al., AltaRica Refinement for Heterogeneous Granularity Models Analysis in *16ème Congrès de Maîtrise des Risques et de Sûreté de Fonctionnement.* Avignon, France, 2008.
3. Bieber, P., C. Castel and C. Seguin, Combination of Fault Tree Analysis and Model Checking for Safety Assessment of Complex System, in proceedings of *4th European Dependable Computing Conference*, 2002. LNCS 2485: pp. 19-31. Springer Verlag.
4. de Roever, W.-P. and K. Engelhardt, Data Refinement: Model-Oriented Proof Methods and their Comparison. 1998: Cambridge University Press.
5. Lisagor, O. and T. Kelly, Incremental Safety Assessment: Theory and Practice, in *26th International System Safety Conference (ISSC).* Vancouver, 2008. System Safety Society.

6. Lisagor, O., J.A. McDermid and D.J. Pumfrey, Towards a Practicable Process for Automated Safety Analysis, in *24th International System Safety Conference (ISSC).* Albuquerque, NM, 2006. System Safety Society.

7. Lisagor, O., L. Sun and T. Kelly, The Illusion of Method: Challenges of Model-Based Safety Assessment, submitted to *28th International System Safety Conference (ISSC).* Minneapolis, MN, 2010. System Safety Society.

8. Manion, M., The epistemology of fault tree analysis: an ethical critique. International Journal of Risk Assessment and Management, 2007. **7**(3): p. 382-430.

9. Schneider, S., Timewise Refinement for Communicating Processes in proceedings of *9th International Conference on Mathematical Foundations of Programming Semantics*, 1993. LNCS 802: pp. 177-214. Springer Berlin / Heidelberg.

10. Schneider, S., Concurrent and Real Time Systems: The CSP Approach. Worldwide Series in Computer Science, D. Barron and P. Wegner series eds. 2000, Chichester, UK: John Wiley & Sons.

11. Society of Automotive Engineers (SAE), *Certification Considerations for Highly-Integrated or Complex Aircraft Systems* *(ARP 4754 / ED-79).* 1996a, SAE International / EUROCAE: Warrendale, PA. [Aerospace Recommended Practice]

12. Society of Automotive Engineers (SAE), *Guidelines and Methods for Conducting the Safety Assessment Process on Civil Airborne Systems and Equipment* *(ARP4761).* 1996b, SAE International: Warrendale, PA. [Aerospace Recommended Practice]

13. Woodcock, J. and J. Davies, *Using Z: Specification, Refinement and Proof*. Prentice-Hall International Series in Computer Science, C.A.R. Hoare series ed. 1996: Prentice Hall.

Biographies

O. Lisagor, Research Associate, Computer Science Department, The University of York, Heslington, York, YO10 5DD, UK; telephone – +44 1904 434728, fax – +44 1904 432708, e-mail – oleg@cs.york.ac.uk.

Oleg Lisagor is a research associate at the HISE group of the University of York and is currently leading one of the work packages of MISSA – a Collaborative European Project. His research interests lie in the general area of model-based safety assessment. In particular, Oleg is looking into safety-related analyses of complex, heterogeneous and software-intensive systems at early stages of the system’s life cycle.

M. Bozzano, PhD, Senior Research Scientist, Embedded Systems Unit, Centre for Information and Communication Technology, Fondazione Bruno Kessler, Via Sommarive 18, 38123 Trento, Italy; telephone – +39 0461 314367, fax – +39 0461 314591, e-mail – bozzano@fbk.eu.

Dr Marco Bozzano is senior research scientist, working in the ES Research Unit of the Centre for Information and Communication Technology of FBK, and he is currently leading one of the work packages of MISSA – a Collaborative European Project. His research interests include model checking and formal safety assessment.

M. Bretschneider, Dr.Rer.Nat, Expert in System Safety/Reliability, Methods & Research Department (EADB), Airbus Operations GmbH, Am Kreetslag, 21111, Hamburg, Germany; telephone – +49 40 74373965, fax – +49 4074 37 8707, e-mail – matthias.bretschneider@airbus.com.

Dr Matthias Bretschneider is a system safety and reliability expert in Airbus Operations. Based on company's Hamburg site, his primary expertise is concerned with stochastic models especially in the context of CS/FAR 25 airworthiness requirements. Having been an active participant in three major EU projects – ESACS, ISAAC and MISSA – Dr Bretschneider's current research interests lie in the areas of model-based safety assessment and application of formal methods to the tasks and challenges of system safety analysis.

T. P. Kelly, PhD, Senior Lecturer, Computer Science Department, The University of York, Heslington, York, YO10 5DD, UK; telephone – +44 1904 432764, fax – +44 1904 432708, e-mail – tpk@cs.york.ac.uk.

Dr Tim Kelly is a Senior Lecturer in software and safety engineering within the Department of Computer Science at the University of York. He is also Academic Theme Leader of the UK MoD Software Systems Engineering Initiative Dependability Theme. His expertise lies predominantly in the areas of safety case development and management.