Safety Case Change Impact Analysis with the Use of Evidence Framework Metamodel

Andrzej Wardziński

Gdańsk University of Technology, Narutowicza 11/12, 80-233 Gdańsk, Poland
andrzej.wardzinski@eti.pg.gda.pl,

Abstract. Effective safety case change management requires a thorough analysis of the change impact on argument structure and the evidence. Usually the evidence is organized by some evidence framework and controlled by configuration management system. We propose to facilitate the safety case change management process by extending the evidence framework with metamodel data to assist change impact analysis. In the paper we describe safety case in the system life cycle context and analyse the safety case change process. We present an evidence framework metamodel and describe the change management process with use of the metamodel. The process is demonstrated in a small medical device case study

Keywords: safety case, change management, evidence framework

1 Introduction

Safety cases evolve in the system life cycle and are the subject of change management as all the other project artefacts. It is a common observation that most of the effort related to a system is devoted to the system maintenance not to the development. We should not expect that safety case is an exception from that rule. When the maintenance effort is abandoned the safety case becomes a shelfware what can lead to disastrous consequences (like described in [1]). Efficient configuration management should ensure the safety case is always consistent, up to date and the status of all its elements is known. As safety cases can grow to a large size the change management process and change impact analysis in particular can become a complex and tedious task, possibly error-prone. Effective change impact analysis requires identification of elements related to or dependent on the change cause. Most of the system maintenance activities will cause the evidence updates and we need assurance that the safety case as a whole remains consistent and valid. Effective change management process should allow for precise identification of the scope of safety case changes and supervision over the identified changes.

A safety case can refer to hundreds of evidence items. A complete set of evidence elements is often called an evidence framework (EF). All the evidence is usually managed under a configuration management system regime. A safety case can relate to a wide spectrum of processes, risks, technologies and life cycle phases starting from the system concept and development through deployment and operation to maintenance. It may happen that the evidence comes from different subprojects, vendors or work groups and each evidence subset is managed separately. In such a case it would be valuable to strenghten the process of safety case change control.

In the paper we will focus on the evidence change impact analysis. We are looking for ways to support safety engineers in the task of safety case evidence change impact analysis.

A more general question is if the evidence framework should be context aware and include informations that facilitate change management. The obvious EF objective is to support the safety argumentation and usually its structure is focused directly on the safety argument and hazards (risks). We analyse possible extension with additional information to facilitate safety case management and change impact analysis in particular.

2 Safety Case and Evidence Change Management

Change management is one of the configuration management processes. Safety case change management should preserve its quality and consistency. Kelly et al. proposed a systematic approach for this process in which impact analysis was based on GSN model analysis [2]. Safety argument structure is clearly one of the possible impact paths however the paper does not discuss what are the other ways of the change propagation. Authors noticed that if a context element is challenged (i.e. is to be modified) then it challenges the relationship with all claims expressed in this context. As a consequence if a given context element (e.g. an system component design) is used for a few claims then all of them are challenged.

Górski et al. proposed the use of a formalized description based on UML to precisely specify the meaning and context of safety case claims [3]. UML modelling allows for specification of precise references to the system elements, their states and attributes, data flow or processes. Using this approach a safety engineer would be able to identify claims related to a given system element, feature, event or hazard. This information can be useful in the process of change management.

Another type of relations that may be relevant to change management is safety causality and hazards. Some standard and guidelines require evidence explicit traceability to hazards [4, 5]. Hazard traceability analysis is useful in demonstrating that controls for all identified hazardous situations have been specified and that the controls have been verified and validated in the final device design.

In practice the essential activity when managing safety arguments and evidence is searching for a particular set of elements related to the topic one is working on at the moment. It can be any system element, hazard, event, document or any other aspect of the system and its context. The first solution is simply searching for text phrases in the safety case. Searching for text patterns is an informal solution but can be quite useful. It can be specially useful when a user can make a search and then make a group operation on the search result like changing status.

More advanced way is based on using tags. Some tools allow to specify tags for safety case elements and then search for them. Denney at al. described a query language for searching and filtering safety cases with the use of complex conditions related to tags [6]. The query result is a safety case subtree containg elements which satisfy the query conditions. Users can define tags related to any context or aspects of the safety case like hazards, safety requirements, system structure or system element types.

OMG SACM standard describes specific attributes for the assurance case elements and evidence elements [7]. The standard specifies a class model which can be used to store and exchange assurance case data with the use of XML file format. From the change management point of view it is essential how the model describes dependencies between elements and status data.

SACM model is divided into two parts, one related to the argument structure and the second one for the evidence. The main assurance case class in the standard is *ModelElement* and all the other classes inherit from this class. SACM standard defines *TaggedValue* class to denote any type of context information for assurance case elements. The SACM model is focused on the argument structure and precisely defines relations between claims, strategies, context elements etc. It does not specify other types of relations but you can easily use *TaggedValue* for this purpose.

The second part of the SACM standard is related to the evidence model. It recognises that evidence life cycle is determined by events and specifies *EvidenceEvent* group of classes to descide them. You can find sub-classes like *IsCreatedAt*, *IsGeneratedAt*, *IsModifiedBy* or *IsTransferredTo* which allow you to define evidence element relation to stakeholders, locations and processes. You can try to use the process attribute to indicate dependencies where an evidence item is an input or output of a given lifecycle process.

IEC 62741 standard for dependability cases defines the evidence framework as a way of presenting the evidence used to demonstrate the claims and support the argument [5]. EF presented by the standard is focused on risks and it requires all the evidence items to be explicitly linked to risks. The scope of information suggested for the evidence framework in informative Annex A includes lifecycle stage, claim and sub-claim, risks and requirements for the evidence, acceptance criteria and approval status however the standard does not precude extending it with additional information.

3 Change Impact Analisys in the Context of Safety Cases

Change management process is composed of a few steps. One of the steps is change impact analysis. Incorrect change impact identification brings a threat that you miss some changes and that can introduce potentially dangerous inconsistency to the system.

The evidence framework artefacts are delivered by all the processes in the system life cycle from the concept phase through development, tests, deployment to operation and disposal. Standards like ISO 15026 [8] and IEC 62741 [5] describe assurance case evolution throughout the system life cycle. The standards stress that the evidence is to be clearly derived from a properly planned safety programme (or dependability programme). The evidence framework can be seen as a bridge between the real world and the safety case. All the changes in the system lifecycle have impact on the safety case through the evidence framework. This relation is sometimes presented in a way as it is shown in Figure 1.

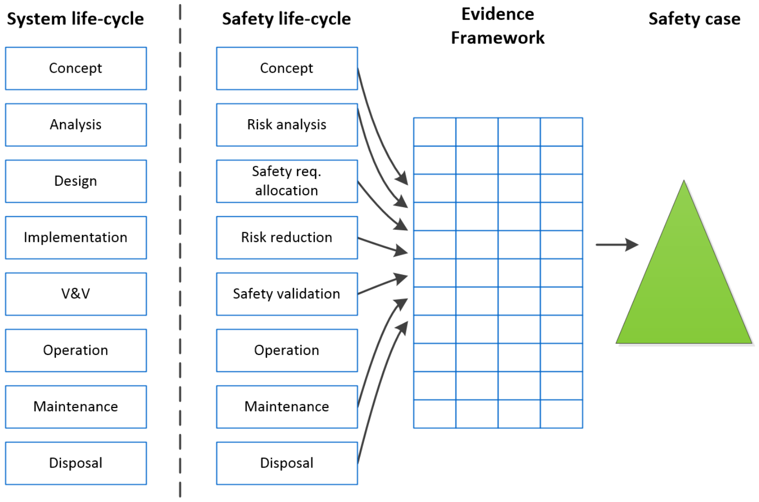


Fig. 1. Safety case in system life cycle context.

Evidence change impact analysis has to take into account the whole spectrum of relations in the system and its lifecycle. We can distinguish a set of relations groups which we call *perspectives*. As a perspective we understand a consistent set of interrelated attribute values that represent a specific type of evidence relationship to the real world. We assume that if two evidence elements are dependent when their attributes of at least one perspective are interrelated. We will use the concept of a perspective to identify evidence change propagation paths. Some examples of evidence perspectives relevant for change impact analysis are:

- system structure (e.g. when a module design document is changed you should check consistency with the subsystem and system design)
- development process (e.g. when a module design is changed you should check if dependent lifecycle artefact like module test plans and test reports are still valid)

- technology (e.g. when a sensor is changed you should verify if other sensors of the same type or using the same technology are also in the scope of change)
- operation data flow (e.g. for a data format change it is necessary to verify all the data usage points like interfaced modules ending with external interfaces, displays or user manuals)
- safety causality (the change may have impact on related hazard analysis artefacts)

The evidence framework metamodel should allow to track change impact for all possible perspectives. The list of perspectives presented above probably is not complete. Safety engineers should take into account all the relationships they observe in the system when analysing the change impact.

4 Evidence Framework Metamodel for Change Impact Analysis

Our approach is to equip the evidence framework with perspectives to allow tracking dependencies to analyse change impact. We start with identification what properties should be presented in the perspective metamodel assuming that it would be useful to develop one common model adequate for all the perspectives. If we take for example system structure perspective we can distinguish two main aspects relevant for change propagation. The first one is the system structure. We assume the metamodel should be able to represent the hierarchical structure for each perspective. The second aspect is modelling dependencies between perspective elements. Modelling dependencies would allow for tracking impact of an element change on other elements. An example of a dependency is an interface between two components of the system structure. Change of one component would require a review of all the interfacing components to verify if their interfaces are not challenged. We assume dependencies work in both ways. When you change a module design document you should verify the change impact on the module test plan. On the other hand when you change the module test plan you should make sure of its consistency with the design document is preserved.

We will develop a simple EF metamodel and then apply it for change impact analysis to validate its effectiveness. The basis of the model is *Perspective* data type which describes the structure and dependencies of one perspective. For a perspective we define two functions to test if two elements are related in the context of the perspective hierarchy or dependencies.

$$\begin{array}{lrcl} \mathit{Perspective} :: & \mathit{elements} & : & \mathit{token} \\ & \mathit{parent} & : & \mathit{elements} \xrightarrow{m} \mathit{elements} \\ & \mathit{dependencies} & : & \mathit{elements} \xrightarrow{m} \mathit{elements}\text{-}\mathbf{set} \end{array}$$

$$\mathbf{inv}\ (\mathit{mk}\text{-}\mathit{Perspective}(e, p, r)) \triangleq \mathit{noLoops}(p)$$

$$\mathit{ancestors} : \mathit{token} \times \mathit{Perspective} \to \mathit{setof}\,\mathit{token}$$

$$\begin{array}{l}
\mathit{ancestors}(e, p) \quad \triangleq \\
\qquad \textbf{if } e \notin \textbf{dom}\, p.\mathit{parent} \\
\qquad \textbf{then } \{\,\} \\
\qquad \textbf{else } \{p.\mathit{parent}(e)\} \cup \mathit{ancestor}(p.\mathit{parent}(e), \mathit{dp})
\end{array}$$

$$\mathit{hierarchyRelated} : \mathit{token} \times \mathit{token} \times \mathit{Perspective} \to \mathbb{B}$$

$$\begin{array}{l}
\mathit{hierarchyRelated}(e1, e2, p) \quad \triangleq \\
\qquad e1 \in \mathit{ancestors}(e2, p) \vee e1 \in \mathit{ancestors}(e1, p)
\end{array}$$

$$\mathit{dependencyRelated} : \mathit{token} \times \mathit{token} \times \mathit{Perspective} \to \mathbb{B}$$

$$\begin{array}{l}
\mathit{dependencyRelated}(e1, e2, p) \quad \triangleq \\
\qquad e1 \in p.\mathit{dependencies}(e2) \vee e1 \in p.\mathit{dependencies}(e1)
\end{array}$$

We can test if two evidence items are interrelated in the context of a given perspective using *hierarchyRelated*() and *hierarchyRelated*() functions. Perspective type can be used when creating evidence framework. We will start with an evidence framework definition with just two perspectives: one related to system structure the second to represent system lifecycle deliverables dependencies. For a given evidence element we can search for other iterrelated evidence elements in a given evidence framework using relations of available perspectives.

$$\begin{array}{lrcl}
\mathit{EvidenceFramework} :: & \mathit{evidence} & : & \mathit{EvidenceElement}\text{-}\textbf{set} \\
 & \mathit{structure} & : & \mathit{Perspective} \\
 & \mathit{deliverables} & : & \mathit{Perspective}
\end{array}$$

Status = { valid, change }

$$\begin{array}{lrcl}
\mathit{EvidenceElement} :: & \mathit{item} & : & \mathit{EvidenceItem} \\
 & \mathit{structureRef} & : & \mathit{structure}.\mathit{elements} \cup \{\mathit{null}\} \\
 & \mathit{deliverablesRef} & : & \mathit{deliverables}.\mathit{element} \cup \{\mathit{null}\} \\
 & \mathit{status} & : & \mathit{Status}
\end{array}$$

$$\begin{array}{l}
\mathit{dependentEvidence} : \mathit{EvidenceElement} \times \mathit{EvidenceFramework} \\
\qquad\qquad\qquad\quad \to \mathit{EvidenceElement}\text{-}\textbf{set}
\end{array}$$

$$\begin{array}{l}
\mathit{dependentEvidence}(\mathit{ev}, \mathit{ef}) \quad \triangleq \\
\qquad \{e \in \mathit{ef}.\mathit{evidence} \mid e \neq \mathit{ev} \wedge e.\mathit{status} = \mathit{valid} \wedge \\
\qquad ((\mathit{hierarchyRelated}(e, \mathit{ev}, \mathit{ef}.\mathit{structure}) \\
\qquad \wedge\, \mathit{structureRelated}(e, \mathit{ev}, \mathit{ef}.\mathit{deliverables})) \\
\qquad \vee\, (\mathit{hierarchyRelated}(e, \mathit{ev}, \mathit{ef}.\mathit{deliverable}) \\
\qquad \wedge\, \mathit{structureRelated}(e, \mathit{ev}, \mathit{ef}.\mathit{structures}))\}
\end{array}$$

The *dependentEvidence*() function can be used in the process of change impact analysis. The function will identify interrelated evidence elements however cannot judge what is the change impact and if an identified element is to be modified or not. This has to be done by a human as it requires analysis of the evidence contents. In some cases the relation is so strong that having one evidence element modified the other always is to be updated (e.g. tests are to be repeated and new test reports are needed when a module design is changed) but our model does not support this type of information. The general algorithm of change impact identification for a given modified evidence element e can be defined as follows:

```
$IdentifyChangeImpact(\ e : EvidenceElement)$
$e.status = change$
$scope = dependentEvidence(e)$
foreach $s$ in $scope$
        if safety engineer confirms impact of $e$ on $s$ then
                IdentifyChangeImpact( $s$ )
        end
end
```

5 Infusion pump case study

We will demonstrate how EF metamodel supports change impact analysis on an example of the infusion pump [9]. The device infuses intravenously medications to the patient at a prescribed basal rate. The device design document comprises AADL model of the system in the operation context. We present a simplified view of the system structure in Figure 2.

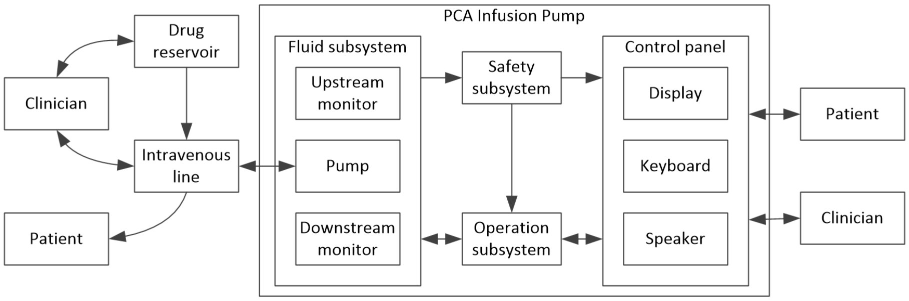


Fig. 2. Infusion pump system structure in its operational context (derived from [9]).

There are dozens of hazards identified for the infusion pump system [4, 9]. We have created the safety case for a subset of the system hazards using NOR-STA software tool. The safety case created for the system has a few layers of argumentation. The layers goals are generally as follows:

1. system is safe when hazards are mitigated
2. hazards are mitigated when preventive, detective and corrective controls are effective
3. controls are effective when safety requirements are satisfied
4. safety requirements are satisfied when the verification process demonstrate satisfaction of requirements criteria

Each level of the argumentation refers to the safety assurance process (e.g. to justify that hazard controls are correctly identified) and to the evidence supporting the claims.

One of the infusion pump hazards is 'air in line'. The intravenous line should be all filled in with the infused medicine. Air when present in the line can be infused into the patient bloodstream causing embolism. Small bubbles in bloodstream can block capillaries in vital organs, like the brain, causing possibly pain, inflammation, neurological damage and even paralysis. Big air bubbles can fill the heart and make it compressing the air instead of pumping the blood. It is estimated that more than 200 cm^3 of air in bloodstream can kill the patient.

The device is equipped with a downstream monitor capable of detecting air bubbles. Like for all other hazards preventive, detective and corrective controls have been identified to control the hazard. Detective controls refer to the downstream monitor sensor for detecting the hazard and alarming the user and the clinician. There are three safety requirements (SR.D1 to SR.D3) identified for air in line hazard detection during the hazard analysis. The corresponding argumentation step in presented in Figure 3.

There are four main categories of the infusion pump safety requirements dependig on the component type. There are requirements related to software functions, electronic hardware elements (like sensors), electrical and mechanical elements (like the pump) and humans (like the patient or clinician). For each requirement category different argumentation strategy should be applied. For example for hardware components the argumentation strategy can require evidence like design document, design review, test plan and test report. A differente set of evidence is needed for software safety requirements.

The set of evidence elements used in the infusion pump safety case is presented in the evidence framework matrix in Figure 4. Each dark square in the matrix denotes an evidence element in a context of two evidence framework perspectives. System structure perspective has been created on the basis of AADL model. In fact AADL models contain precise information about the system structure and interfaces so the metamodel perspective could be generated automatically from the model.

To demonstrate how the metamodel supports the change impact analysis let's assume one of the evidence items is for some reason selected to be modified. In our example the modified element is the downstream monitor. We will analyse two scenarios of the change. In the first scenario the downstream monitor sensor is to be changed for a device supplied by other vendor and the characteristics

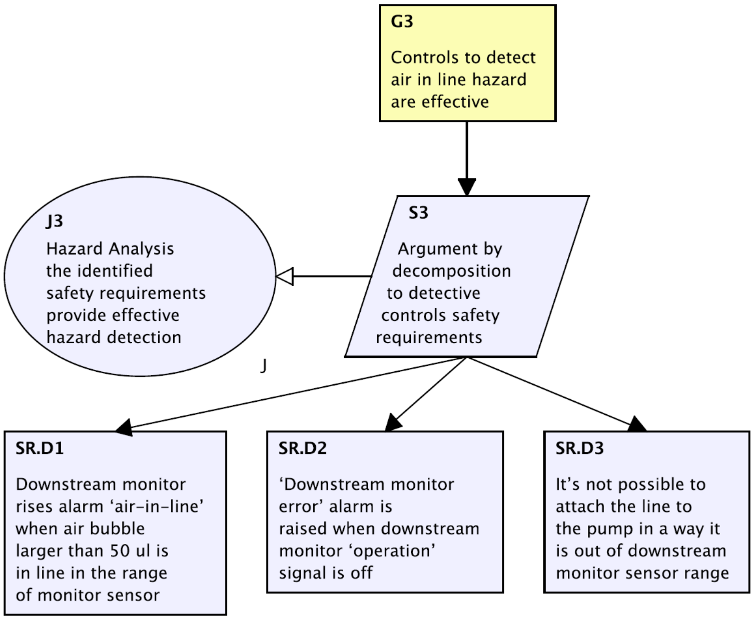

G3
Controls to detect air in line hazard are effective
S3
Argument by decomposition to detective controls safety requirements
J3
Hazard Analysis the identified safety requirements provide effective hazard detection
J
SR.D1
Downstream monitor rises alarm 'air-in-line' when air bubble larger than 50 ul is in line in the range of monitor sensor
SR.D2
'Downstream monitor error' alarm is raised when downstream monitor 'operation' signal is off
SR.D3
It's not possible to attach the line to the pump in a way it is out of downstream monitor sensor range

Fig. 3. Argumentation step related to air-in-line hazard detective controls.

		Concept		Implementation			V&V				Deployment			Operation	
		PHA	Requir.	Design	Code	Review	Test Plan	Test Report	Int.Test Plan	Int.Test	Inst.Instr.	User Man.	Procedure	Training	System Log
		D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14
Users	S1	■													
Patient	S2		■												
Clinician	S3		■									■	■	■	
Infusion set	S4	■							■	■		■	■	■	
Drug reservoir	S5		■	■			■								
Intravenous line	S6		■	■			■								
PCA Infusion Pump	S7	■							■	■					
Fluid subsystem	S8		■	■		■	■	■			■				
upstream monitor	S9		■	■		■	■	■							
pump	S10		■	■		■	■	■							
downstream monitor	S11		■	■		■	■	■							
Operation subsystem	S12		■	■	■	■	■	■			■	■			■
Safety subsystem	S13		■	■	■	■	■	■			■				■
Control panel	S14		■								■	■			
Display	S15		■	■		■	■	■							
Keyboard	S16		■	■		■	■	■							
Speaker	S17		■	■		■	■	■							

Fig. 4. Infusion pump evidence framework matrix

of the new component is exactly the same as of the original one and it supports the same interface.

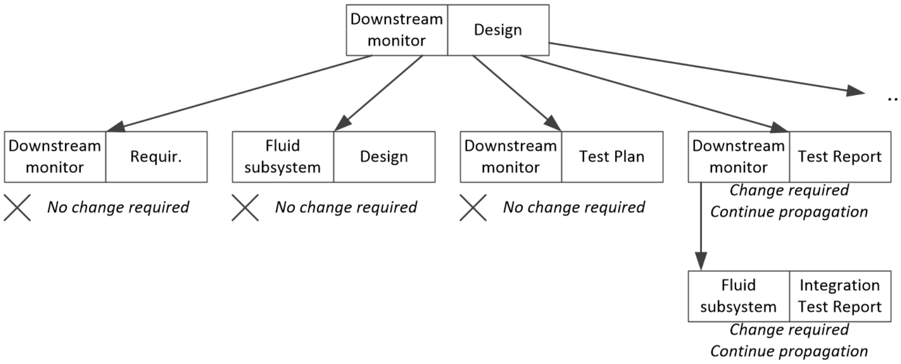


Fig. 5. Steps of the change impact analysis for the first sensor change scenario

We will follow the steps of *identifyChangeImpact*() algorithm. The initial changed element is downstream monitor design document. In the first step we will identify a set of evidence items directly interrelated to the monitor design. The change propagation path can go along the structure perspective to fluid system design or along the lifecycle perspective to downstream monitor requirements or test report. This step of change impact analysis is shown in Figure 5. Each of the selected evidence items is to be assessed by a human if a given item is required to be modified to preserve system consistency. In this case change of the downstream monitor does not cause any change of its interfaces therefore change of other elements is not required. In accordance with the lifecycle perspective test reportes are challenged by the monitor monification. Test reports are to be updated what means that tests should be repeated.

Now let's examine the second scenario. The change in the second scenario is more complex as the new sensor differs from the existing one and uses different interface. In this scenario change of the sensor interface makes the fluid subsystem design document outdated. As a consequence the fluid subsystem design is to be included in the scope of changes. This step in presented in Figure 6. The figure shows nine other evidence items influenced by the fluid subsystem design change. One of the lifecycle depencencies states that design documents are interrelated with integration test plans and reports. Using the system hierarchy perspective *identifyChangeImpact*() can identify that fluid subsystem design document change can be propagated to the system integration test plan and report.

The results of change impact analysis for both scenarios are presented in the evidence framework matrix in Figures 7 and 8 respectively. You can find three categories of the change impact presented in the EF matrixes. First, numbers

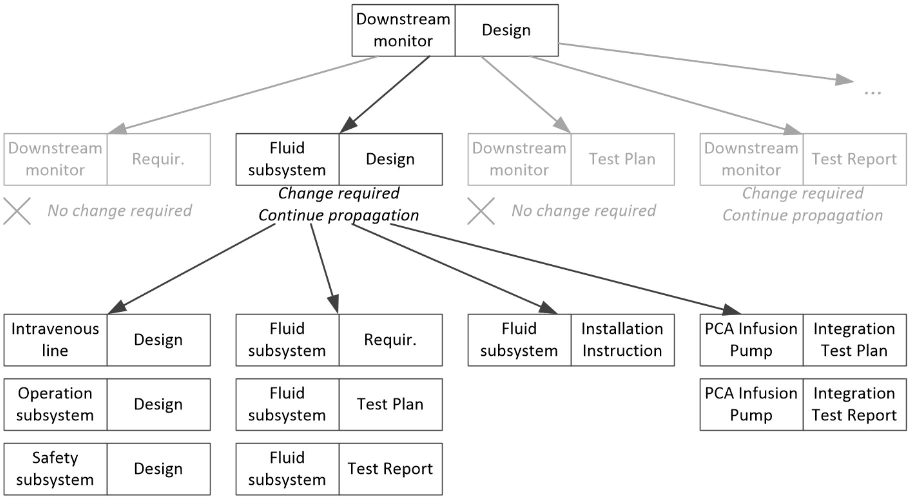


Fig. 6. Steps of the change impact analysis for the second sensor change scenario

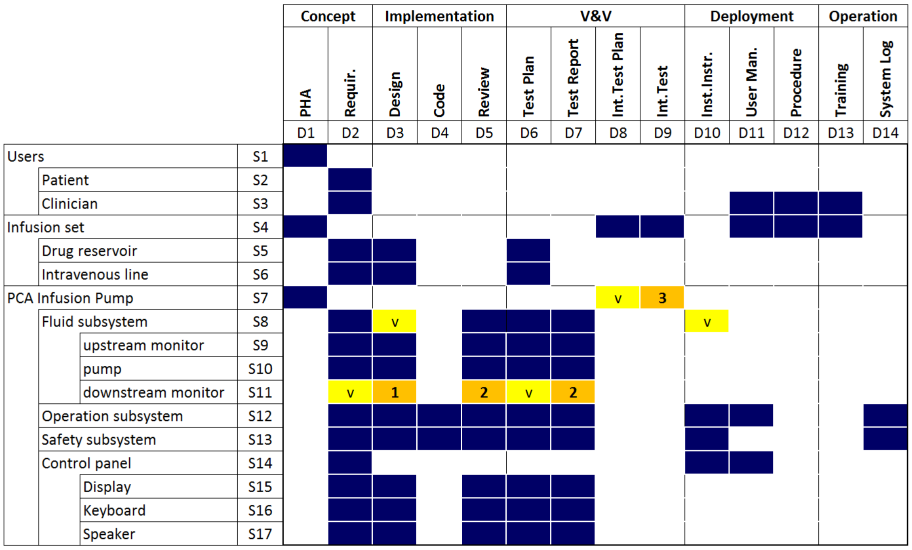
indicate items in the scope of change (the number denotes the step when the change was identified), letter 'v' is used for items that have been verified by humans to be valid and out of the scope of change (elements on the border of the change scope). All the other items marked as dark squares are out of the scope of impact analysis.

The evidence classification into three categories can be used to create a change profile (Figure 9). Change profile characterizes the scope and border of the change. Possible change profiles can be automatically generated for a given initial change and can be used for verification of the change scope correctness. In particular in this way omission of an element in the change impact analysis could be detected. Any difference between the modelled change profile and the real life set of modified and verified evidence elements should be carefully inspected to detect the cause of inconsistency.

The user should be aware of the limitations of the approach. The change profile will not be correct when some perspectives relevant for change impact are not specified or are specified incorrectly. The user can manage the scope of the metamodel information to achieve appropriate support for change impact analysis.

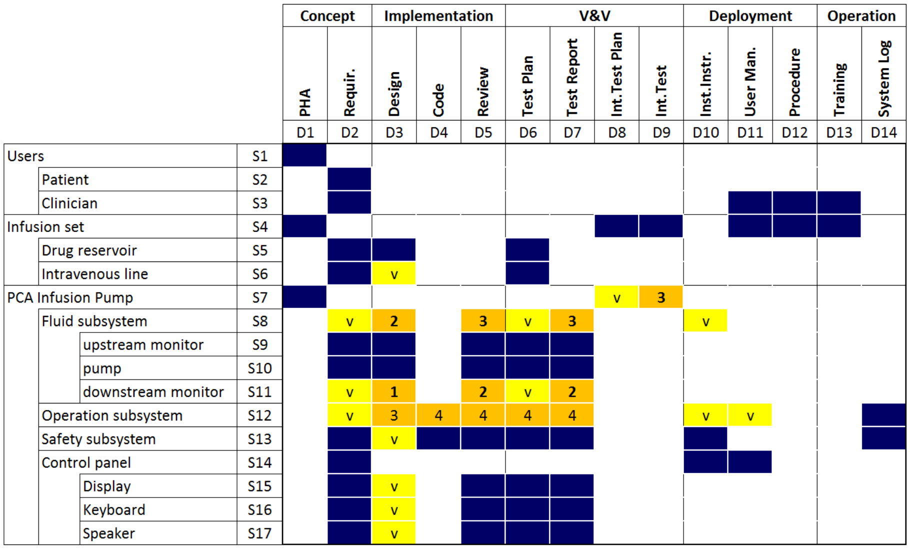
6 Summary

The evidence framework represents a consistent set of evidence items organized to support safety case claims related to system risks or hazards. Consistency is one of the crucial values if the evidence framework. In the paper we propose the use of the hierarchical tag system to describe evidence relations and assist the



		Concept		Implementation			V&V				Deployment			Operation	
		PHA	Requir.	Design	Code	Review	Test Plan	Test Report	Int.Test Plan	Int.Test	Inst.Instr.	User Man.	Procedure	Training	System Log
		D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14
Users	S1	■													
Patient	S2		■												
Clinician	S3		■									■	■	■	
Infusion set	S4	■							■	■		■	■	■	
Drug reservoir	S5		■	■			■								
Intravenous line	S6		■	■			■								
PCA Infusion Pump	S7	■							v	3					
Fluid subsystem	S8		■	v		■	■	■			v				
upstream monitor	S9		■	■		■	■	■							
pump	S10		■	■		■	■	■							
downstream monitor	S11		v	1		2	v	2							
Operation subsystem	S12		■	■	■	■	■	■			■	■			■
Safety subsystem	S13		■	■	■	■	■	■			■				■
Control panel	S14		■								■	■			
Display	S15		■	■		■	■	■							
Keyboard	S16		■	■		■	■	■							
Speaker	S17		■	■		■	■	■							

Fig. 7. Change 1 impact presented in the evidence framework matrix



		Concept		Implementation			V&V				Deployment			Operation	
		PHA	Requir.	Design	Code	Review	Test Plan	Test Report	Int.Test Plan	Int.Test	Inst.Instr.	User Man.	Procedure	Training	System Log
		D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14
Users	S1	■													
Patient	S2		■												
Clinician	S3		■									■	■	■	
Infusion set	S4	■							■	■		■	■	■	
Drug reservoir	S5		■	■			■								
Intravenous line	S6		■	v			■								
PCA Infusion Pump	S7	■							v	3					
Fluid subsystem	S8		v	2		3	v	3			v				
upstream monitor	S9		■	■		■	■	■							
pump	S10		■	■		■	■	■							
downstream monitor	S11		v	1		2	v	2							
Operation subsystem	S12		v	3	4	4	4	4			v	v			■
Safety subsystem	S13		■	v	■	■	■	■			■				■
Control panel	S14		■								■	■			
Display	S15		■	v		■	■	■							
Keyboard	S16		■	v		■	■	■							
Speaker	S17		■	v		■	■	■							

Fig. 8. Change 2 impact presented in the evidence framework matrix

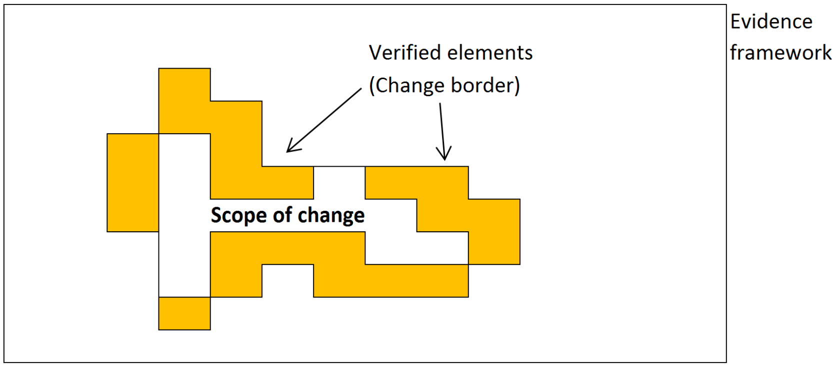


Fig. 9. Change profile in the context of the evidence framework

change impact analysis. We introduce the concept of a perspective to denote a specific dimension of evidence internal relations and relations to the real world elements. Examples of a perspective are the system structure, data flow, system life cycle, hazards and others. Some of the perspectives are usually already in use for evidence management (like hazards and safety argument structure) and some others can be extracted from existing sources, like system structure and interactions specified in design models. A set of applied perspective information can be adjusted to represent relations appropriate for change impact analysis of a given system.

Usually change management is an element of the overall project configuration management process and it is not our objective to supersede it with EF metamodel. The goal of the approach it to strenghten change management of the evidence framework and stress the fact that evidence framework consistency is an attribute that has to be taken into account in the system configuration management process.

The proposed evidence framework metamodel uses simple data structures and algorithms which can be easily extended to other perspectives when necessary. We avoid creating specific data structures for a given type of evidence relations. In practice the approach can be implemented as a hierarchical extension for TaggedValue defined by OMG SACM standard [7]. The solution is flexible and easy to use. Tags can be used for searching and filtering safety case and evidence items by safety engineers. The metamodel even if not used directly to change management helps with maintaing evidence framework and safety case consistency.

References

1. Haddon-Cave, C.: The Nimrod Review: An independent review into the broader issues surrounding the loss of the RAF Nimrod MR2 Aircraft XV230 in Afghanistan in 2006, House of Commons Stationary Office, London (2009).

2. Kelly, T.P., McDermid, J.A.: A systematic approach to safety case maintenance, Reliability Engineering and System Safety, vol. 71, pp. 271-284, Elsevier (2001)
3. Górski, J., Jarzebowicz, A., Leszczyna, R., Miler, J., Olszewski, M.: Trust case: justifying trust in IT solution, Proc. Safecomp Conference, Reliability Engineering and System Safety, Elsevier, vol. 89/1, pp. 33-47 (2005)
4. Infusion Pumps Total Product Life Cycle Guidance for Industry and FDA Staff, U.S. Department of Health and Human Services, Food and Drug Administration (2014)
5. IEC 62741:2014, Demonstration of dependability requirements - The dependability case, International Electrotechnical Commission (2014)
6. Denney, E., Naylor, D., Pai1, G.: Querying Safety Cases, in A. Bondavalli and F. Di Giandomenico (Eds.): SAFECOMP 2014, LNCS 8666, pp. 294309, Springer International Publishing Switzerland (2014)
7. Structured Assurance Case Metamodel (SACM), Object Management Group (2013)
8. ISO/IEC 15026-4, Systems and software assurance – Part 4: Assurance in the life cycle, International Organization for Standardization, International Electrotechnical Commission (2012)
9. R. Larson, B.R., Hatcliff J.: Open Patient-Controlled Analgesia Infusion Pump System Requirements, Draft 0.11, SAnToS TR 2014-6-1, Kansas State University (2014)