Introducing Safety Cases for Health IT

George Despotou
Department of Computer Science
University of York, UK
george@cs.york.ac.uk

Tim Kelly
Department of Computer Science
University of York, UK
tim.kelly@cs.york.ac.uk

Sean White
National Health Service,
Connecting for Health (NHS-CfH), UK
sean.white@nhs.net

Mark Ryan
Rotherham Hospital
Rotheram NHS Foundation Trust, UK
mark.ryan@rothgen.nhs.uk

***Abstract*— Introduction of IT in the health domain can potentially benefit the quality of the delivered healthcare, also contributing to increase safety. However health IT systems themselves can have safety implications and can result in accidents. Creating a safety case has been in practice in numerous domains and is starting to be adopted in the health IT domain with the most notable example, that of the UK National Health Service (NHS), Information Standards Board for Health and Social Care (ISB) standards (formerly DSCN 14/2009 & DSCN 18/2009). Safety cases can be thought of as a defensible, comprehensible and clear argument, supported by evidence that a system is acceptably safe in its operational context. This paper presents the main areas of safety case practice and its implication for the health IT development and stakeholders.**

Keywords-Safety case, hazard analysis, evidence, software safety, software assurance, ISB, DSCN

I. INTRODUCTION

There has been an increasing use of IT in healthcare, aiming to improve the healthcare quality as well as safety delivered to patients. IT can contribute to improvement on meeting the performance targets of hospitals by offering functionality for the management of patients. Furthermore, a number of studies have indicated that IT can also favour the safety of the delivered healthcare; for example electronic prescribing systems are considered to considerably reduce potential human error that may result in adverse effects for the patient [1]. However, use of health IT is not innocuous; erroneous use of health IT and faults in the health IT system itself may deviate its intended operation, resulting in conditions that may pose a risk to the patient. Quality, is an aspect of IT that can be apparent in all every day devices, but in healthcare, poor IT quality may have safety implications. For example, consider an electronic prescribing system that corrupts data about the dose of a prescription, or a patient management system that stores incorrect information about the criticality of the condition of the patients. Health IT may affect all aspects of healthcare ultimately affecting the overall clinical safety. Risks from health IT may be the result of both faults in the software itself, introduced during development or, unintended operation of the software by the user.

Over the years, and motivated by a number of accidents, there has been significant interest on behalf of contractors, customers and independent authorities, in being able to capture and communicate assurance about the safe operation of a system. This has resulted in a number of standards requiring a system to be accompanied by a safety case. The safety case communicates an argument, supported by evidence that a system is acceptably safe in a given operational context. Safety cases explicitly capture a position about the safety of a system, and explain how the available evidence collected during the various phases of the system development supports this position. Safety cases have been used in a number of domains [2], from control software in the nuclear industry to software onboard automotive vehicles. In the NHS, clinical safety of health IT systems is managed through two complimentary standards [3,4]:

- ISB 0129 (DSCN 14/2009): Application of clinical risk management to the manufacture of health software
- ISB 0160 (DSCN 18/2009) Guidance on the management of clinical risk relating to the deployment and use of health software

Both standards are issued by the Information Standards Board for Health and Social Care (ISB) on behalf of the NHS and Adult social Care in England. These standards consider the clinical risk management processes required to ensure patient safety both in terms of manufacture and deployment of IT based health systems; and establish the requirements for an iterative safety case to be developed as the primary mechanism to organise, evidence and communicate the safety characteristics of the developed and deployed health IT system. ISB 0129 focuses on the product delivered by the supplier, whereas the scope of ISB 0160 includes the product, and the socio-technical aspects of its operation (e.g. users such as doctors and nurses), all of which constitute the system under safety analysis. Nevertheless, despite the different focus, the two ISBs should be seen as complementary to each other covering safety aspects of IT software from its inception to the deployment within a larger

clinical operating environment. The standards are based upon "ISO 14971 Medical devices -- Application of risk management to medical devices" thereby maintaining a consistent approach to risk management in the healthcare domain.

II. What Is a Safety Case

A safety case exists to communicate an argument. It is used to demonstrate how someone can reasonably conclude that a system is acceptably safe from the evidence available. A safety case is a device for communicating ideas and information, usually to a third party (e.g. a regulator). In order to do this convincingly, it must be as clear as possible. Safety case definition may bear differences in different domains, but all definitions converge to the aforementioned characteristics. Similarly, the ISB defines the safety case as :

> "*Accumulation, through, the lifecycle of the health software system, of product and business process documentation and of evidence structured such as to enable a safety argument to be developed to provide a compelling, comprehensible and valid case that a system is, as far as the clinical risk management process can realistically ascertain, free from unacceptable clinical risk for its intended use"*

A safety case consists of an argument and evidence supporting it. An argument is described as being "a connected series of statements or reasons intended to establish a position" [5]. An argument consists of [9]:

- Claims: Claims represent statements about the system made at various abstraction levels. The highest level claim (e.g. system X is acceptably dependable) assumes the role of the overall position of the argument. Consequent claims establish (at different abstraction levels) specific positions about the system's operation (e.g. system X implements authentication, subsystem Y responds in timely manner), which contribute to establish the overall argument. Claims are true/false propositions capturing assertions that can be made about the system, In contrast to 'traditional' requirements, which make should/must demands. Furthermore they represent intent rather than a specification of the system.

- Inferences: Claims are interconnected with each other, based on inference rules, forming a network. An inference is described "as the drawing of a conclusion from known or assumed facts or statements" [8]. In the case of argumentation, the known or assumed facts being the existing claims about the system, and the conclusions being the new claims about the system (or child claims). Evolution of an argument usually takes place until the claims can be directly supported by the available evidence.

Insufficiency of evidence to support the developed argument structure reveals the need for activities that will produce additional evidence (e.g. additional system testing). Both argument and evidence are crucial elements of a safety case. An argument without evidence is unfounded whereas evidence without argument is unexplained. A set of evidence does not constitute by itself a safety case. Argumentation can be used to capture any position about the system that the stakeholders choose to communicate, irrespective of the framework that was followed to establish this position. Establishing a (safety) case can provide a number of advantages, regardless of it being imposed by standards:

- Completeness on safety management: Allows for a systematic and thorough examination of the system behaviour. Particularly in presence of diverse evidence and implicit arguments.
- Rationale behind prescriptive requirements that may be missing (e.g. procedures).
- Communication of knowledge. There is an expected imbalance between system stakeholders (e.g. developers and regulators).
- Explanation of the role of evidence in the case which can often be unclear.
- Documentation of assumptions and implicit judgments that need to be explicitly captured.

Evolving in parallel to the system development, the argument claims are decomposed until they can be directly supported by evidence collected during the development and testing phases of the system.

III. Risk Assessment and Hazard Management

Hazard analysis and risk assessment can be thought of as two of the most fundamental activities of safety. During hazard analysis the system stakeholders identify conditions arising during the operation of the system that can have safety related consequences. Contrary to other system requirements, hazards can constitute negative requirements – i.e. something that needs to be avoided during a system's operation.

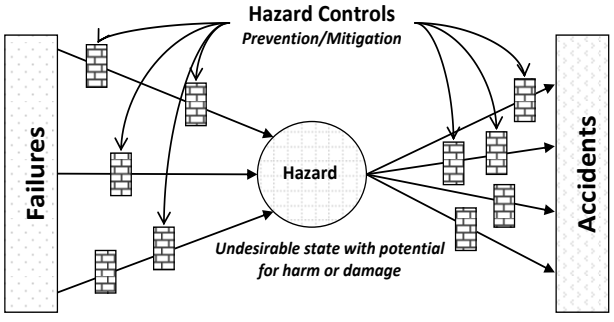

Figure 1 – Bow Tie diagram

Risk assessment is the process during which the hazards are assessed in terms of their severity and likelihood. Various frameworks are used assess the risk of a system.

With respect to the health domain, these activities provide a number of difficulties. The challenges of hazard analysis relate to the activity itself as well as to how the hazards are then managed. Hazard analyses need to be complete in identifying all possible hazards. This is difficult on its own right as hazard analysis requires a high degree of domain knowledge. A number of the identified hazards can be identified based on domain expertise whereas others will be identified after analysis of the operation of the system. The latter involves identification of how the functionality of the system can result in hazardous conditions within the system's operational context. Hazards are defined in relation to consequences during the operation of a system and thus they should be distinguished from faults. For example in the context of health IT a hazard could be 'administering the wrong dose to patient' as it can directly result in an accident, whereas a 'corrupt data record' can be thought of a system specific fault and contributing factor to the hazard. Whereas a fault within the system may be a contributing factor to an accident, it may not necessarily be a hazard. Hazards are the states that result from the manifestation of one or the combination of more than one fault. Furthermore, in complex systems hazards may occur without the presence of a fault. Figure 1 shows how system failures result in hazards and accidents (also known as the bow-tie diagram). Managing the hazards involves both prevention (e.g. checklists) and mitigation (e.g. recovery procedures) of hazard.

Risk assessment is a difficult activity during the safety lifecycle of a system as it is very difficult to quantify the likelihood of an event; in particular more scarce events. Whereas the likelihood of more frequent events can be captured, calculation of the likelihood of more rare events depends on a number of assumptions. This can result in inconsistencies of risk between different developers, whereas ideally there should be consistent approaches that can be communicated to and evaluated by the regulator easily.

One caveat with risk analysis is the scope in which it is performed and its acceptability. The scope of the risk analysis should not be localised but should consider the entire operation of the system. The residual risk to the affected (by the health IT system) healthcare pathways should be identified as whole. Furthermore, the acceptability of risk will need some informed decisions. Claiming the achievement of absolute safety is not realistic. Instead the risk of a system should be considered in relation to the benefits received by that system. It can be the case that a software system is considered to have high risk but, the benefits that it offers are disproportionate, thus justifying its deployment. Furthermore there can be trade-offs between risk and cost, whereas a certain risk level is considered acceptable when considering the cost of reducing it. Although this may seem as a crude approach risk and cost trade-offs are not a simple cost benefit analysis [6] but needs justification. Furthermore, risk trade-offs should be considered at the organisation level as the resources to reduce a certain risk may be redirected for a higher improvement to another aspect of healthcare thus reducing the overall risk to the patient. Regardless of the criteria for the acceptability of risk, the operator should be in a position to make informed decisions with acceptable levels of uncertainty, the identification of which is enabled by the practice of safety cases.

IV. The Lifecycle of the Clinical Safety Case

Based on accumulated experience from the safety domain, many standards (such as [7]) suggest that a (safety) case should be constructed in parallel with the system and not at the end of the system lifecycle. Creating an argument about the achievement of safety in retrospect, after the end of the system development process can be problematic. The design of the system may not be optimised for developers to create a strong argument about its safety. Instead, developers will eventually 'force' the argument to support the overall claim often resulting in a weak argument (e.g. relying on operational constraints). Thus the case will not be able to communicate a satisfactory argument about assurance of the system's safety. This means that either the system will be delivered with a case that doesn't provide a satisfactory degree of assurance, something that may result in rejection of the system; or parts of the design will need to be reconsidered so that a stronger argument can be created. There are limited opportunities during the later stages of the lifecycle, for developers to easily (and cost effectively) revise the system to meet the required safety objectives. Evolution of the system and the argument involves making decisions about the architecture and the design of the system that needs to be justified and documented.

Interaction between the argument and the design process exists during the evolution of the system. The evolving argument should serve to evaluate the design's fitness to satisfy the stated safety goals. A design that directly addresses the stated goals will result in a strong argument. If the stakeholders involved in the development of the argument deem that the argument is not satisfactory, changes to the design will have to be made. The safety case should be seen as a 'living artefact' constantly being informed by the processes of the system lifecycle. At each stage of the evolution of the safety case, the argument is expressed in terms of what is known about the system being developed. In the early stages of project development the argument is related to the requirements as conceived by the stakeholders, based on the intended operational concept of the system and in some cases a high level conceptual design. In the same time, the overall safety objectives and the hazards arising from the operation of system are identified (based on the conceptual design), along with a statement of intended (safety) controls.

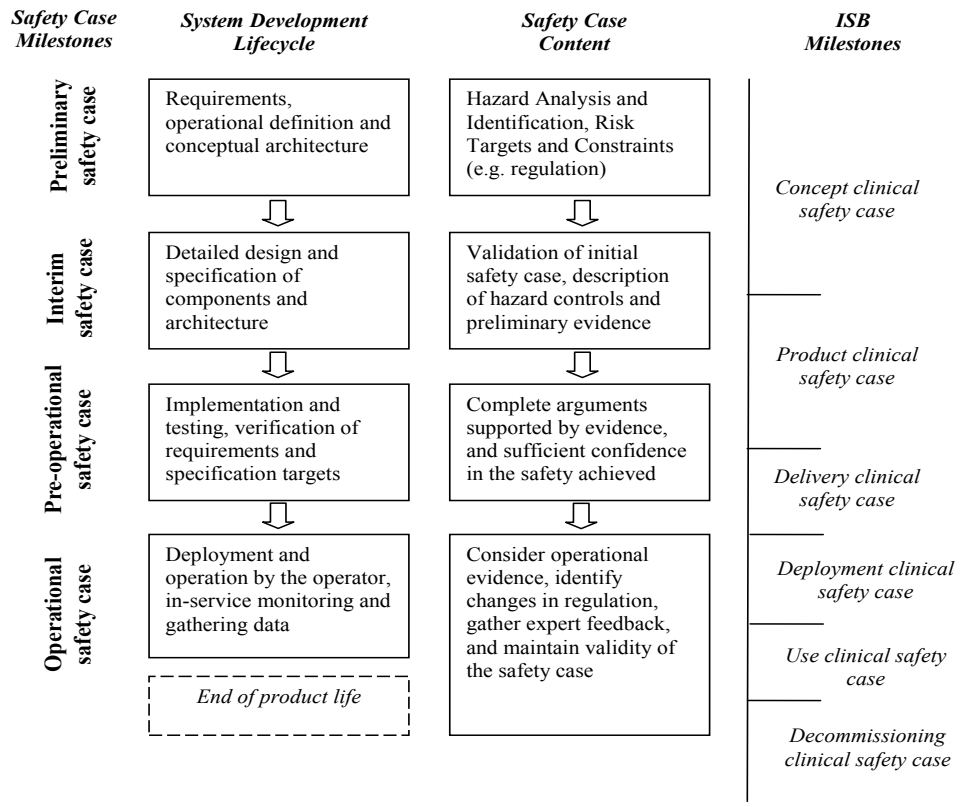

Figure 2 – The Lifecycle of a Safety Case

As design knowledge increases during the project, these objectives (and the corresponding argument) can be expressed in more tangible and specific terms. During this stage the safety controls are validated, are designed in detail and safety related specifications are assigned to the various parts of the system. Following implementation, the system is tested and the derived specification is verified. During this stage the evidence available to support the safety claims is identified and the argument that explains how the evidence supports the claims is finalised.

Following deployment, operational data can be collected and maintenance updates can be deployed. Operational data can provide further evidence to the safety case, but they can also provide counter-evidence for parts of the safety case. For example, consider a task that a particular role (e.g. nurses) needs to perform that adds another check when using the system, which however is difficult to do because the system is not optimised (e.g. not appropriate menus) for this. This may result in needing (what could be considered) disproportionate time, eventually being (informally) dropped. This would provide counter-evidence that the controls that were put in place are not followed, thus the claims in the safety case becoming unsupported. In such an occasion the safety case would need to be revisited and alternative (more realistic) controls to be implemented. Furthermore, in terms of the safety case, post-deployment monitoring is necessary to argue the on-going validity of the safety case. Changes to the regulatory frameworks as well as to the operation of the system will affect the assumptions about the operational context, made during the safety case development thus compromising the relevance of the argument encapsulated in the safety case. Furthermore, changes to operation may introduce new hazards. Unless the introduced changes are analysed to understand their impact, the uncertainty about the safety operation of the system will increase. In addition to these changes, updates and upgrades from the developer will affect the safety case in the same manner, thus requiring analysis.

The development stages of the system and the safety case are rarely discrete, and progression through each stage is gradual, and often iterative. In order to allow the monitoring of the safety case throughout the system's lifecycle, a number of safety case reports are produced. It should be noted that the safety case is the entirety of the reasoning and evidence either implicitly or explicitly documented, whereas the safety case report is the artefact offering a snapshot of the safety case at any given time (however, the term safety case is often used to refer to the safety case report) [8].

ISBs suggest a way in which the development of the safety case (in terms of deliverables) can be structured. ISBs identify safety case stages according to the (sources of) information about operation available during the development of the system. Initially the safety case is based

on the assumptions of the developer about the operation of the system. The delivery and deployment safety cases reflect the input of the user, explaining why the system is safe in that particular deployed operational context. Finally, the use and deployment safety cases are more focused on input from the user and the data collected during operation. The decommissioning safety case in particular, explains the safe decommission of the system, including impact on the safety of the services delivered by the system being withdrawn and analysis of potential functionality gaps.

Although it can be argued whether the ISB milestones have been designed in an optimum manner, it is necessary to highlight that the sequence of milestones represents the importance of clinical input to the safety case. Developers often assume a representative generic set of requirements aiming to cover most cases. However, requirements may change for each particular use of the system, making clinical input (i.e. the users' input) critical to identify operational assumptions. For example, in the UK (contrary to the US) there is some overlap (in terms of system functions accessed) between nursing staff and doctors. Software optimised for the US market assumes the clear distinction between the roles, completely separating the functions accessed; this approach would require nurses needing to have both a nurse's account and a doctors account, thus needing process controls to mitigate inadvertent use of doctors functionality, whereas an access control approach might be more suitable.

Finally, during the evolution of the safety case, the developer and the user can identify the types of information likely to be required by the end of the safety case development, and adjust their processes to generate it. For example, a developer can identify early that a particular type of testing might be required, whereas the user may identify early in the process that a particular type of measurement or implementation of procedures will be required.

V. The Arguments in the Health IT Safety Case

Figure 3 presents a potential structure, and the arguments that should be encapsulated by the safety case. These are the hazard management argument, the confidence argument and the compliance argument. Each of the arguments contains a set of evidence that has been considered suitable to support the claims in the argument. Each of the arguments is not stated in isolation, but they could also be merged. However this modular approach, allows separating the concerns, assists with the clarity and comprehensibility and allows developers to focus. There are dependencies between the arguments and they should be seen as being stated in context of each other.

A. Hazard Management Argument

This argument communicates the reason as to why the system is safe, or in other words the effective management of hazards. It acknowledges the safety objectives for the system, identifies the hazards to the operation of the system, and claims sufficient hazard management, supported by appropriate evidence.

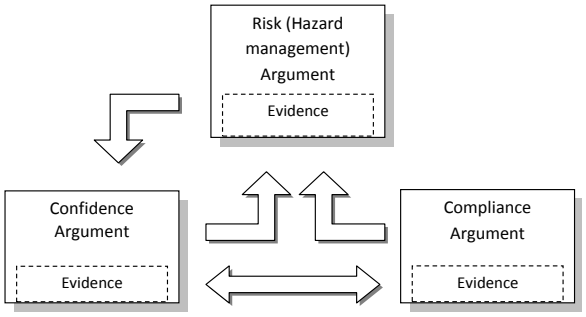

Figure 3 – Overview of the Safety Case Arguments

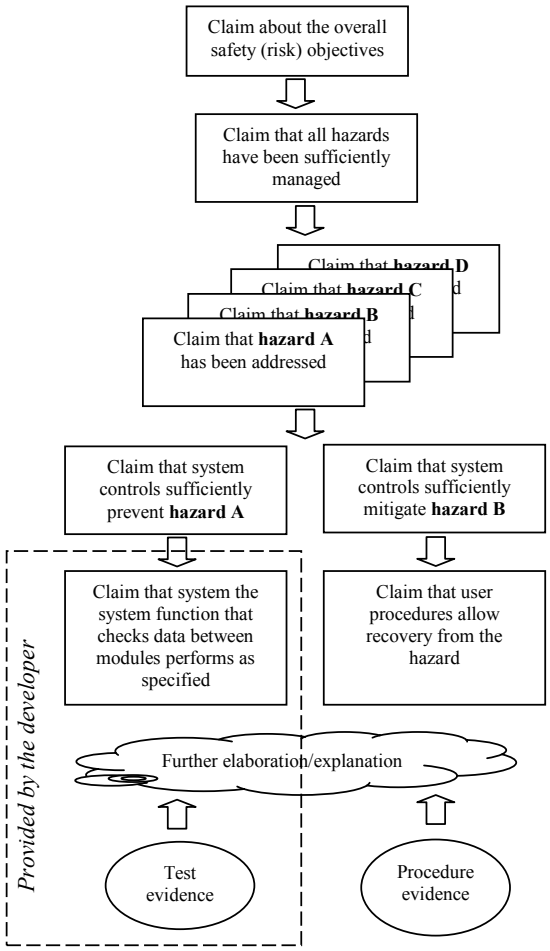

Figure 4 – Exemplar Hazard Management Argument

This can be considered the heart of the safety case as it is this argument that actually explains how the available evidence supports the claims about hazard management (Figure 4). Hazard management can include both prevention and mitigation, hence may require different types of evidence for each type of claims.

A distinction should be made between the evidence provided by the operator and the evidence provided by the developer. For example, if a hazard is controlled by means of a process implemented by the clinical staff (e.g. a checklist) the evidence is provided by the user. Whereas, if a hazard is controlled by means implemented by the system itself (e.g. data checks between software modules) then the evidence is expected to be provided by the developer. Overall, the hazard management argument should be very focused on explaining how the evidence supports the claims made about safety.

B. Confidence Argument

The confidence argument provides an explanation as to why one should place their confidence in the hazard management argument. A simple way to understand the role of the confidence argument in the safety case is to consider that it mainly answers the question: "why should one believe a hazard management argument?" The hazard management argument leaves a number of unanswered questions that may undermine its credibility, about issues (among others) such as: completeness of the identified hazards, quality and trustworthiness of the evidence in the hazard argument, quality of the safety and safety case development processes used, and strength of evidence.

C. Compliance Argument

The compliance argument explains how the safety case has met the requirements of a particular regulatory framework (e.g. a standard). The compliance argument provides:

- A clear perspective for the regulator
- Allow the main hazard and confidence argument to be standard agnostic
- Helps avoid entanglement of process and hazard management argument

In the case of ISB, the compliance argument would explain how the requirements of the applicable ISB standard have been met. The compliance argument can be a very useful view for the regulator as it shows how the requirements of a standard have been met. Moreover it helps to avoid a common mistake in safety, which is the assumption that following a particular (albeit often a rigorous) process will result in a safe system. This is an implicit argument that makes an assumption about the safety of the delivered system by appeal to the rigour of the processed followed. But such an argument tells little about the actual management of hazards and thus is weaker than an argument that would provide direct evidence about hazards management. By using a compliance argument a developer or operator will have some degree of flexibility to argue about safety as they see fit, and then still have an opportunity to explicitly claim how they have met the requirements of the standard. The compliance argument can be as simple as a traceability matrix of a standard's clauses and a reference to the rest of the safety case where the clauses have been implemented. For example, ISB 0160 asks for competent personnel (§5.3) which would reference the claim about personnel competency (in the confidence argument).

VI. Safety Case Stakeholders

As described, the safety cases consists of the entirety of the (safety related) including a large amount of reasoning and information about the system. Different stakeholders are interested in the safety case for different reasons.

A. Regulator

A well structured safety case and its associated safety case report provides an effective mechanism of communication within and between organisations. The underlying philosophy in the ISBs, is that a manufacturer should make their safety case available to the deploying Health Organisation. In doing so they are able to declare any assumptions made about the operational context that their system will be used in and communicate any potential residual risk associated with their system. In turn, the Health Organisation should use this safety case as a starting point in their safety activities. They should review it and satisfy themselves that any assumptions made or controls cited hold true in their operational context. Where weakness or gaps are identified the Health Organisation can then plan to address them.

B. Operator

Understanding the context that a piece of health software was designed for (and consequently not designed for), can greatly help the deploying organisation understand the benefits and limitations of the software they are intending to use. Using the safety case set out by the software designers in ISB 0129, can greatly help anticipate which areas of use may present problems and shortcomings in terms of functionality, and also allow cross-referencing between the designers intended use and operator's intended use, in the same time identifying any incompatible or different assumptions. In many organisations there isn't always an exact match between these two intentions and nor is it possible, without a bespoke package, for the expectations in functionality to completely align with local processes.

In turn, the deploying organisation must feel assured that these misalignments are sufficiently mitigated in terms of risk, as it is their duty of care to patients to protect them from any potential increased risk imposed by a change of practice involving an IT system. In order to do this they should develop their own safety case as set out in ISB 0160. This might be achieved by:

- Developing the criteria which are considered a requirement for safety assurance

- Explicitly developing a safety argument which claims and shows how those criteria can be met
- Evidencing through the existence of hardware requirements, software routines and test results that automated processes are consistent and reliable
- Evidencing that where software and/or evidence does not exist, that an alternative policy outlining the process that should be followed to ensure safety, does exist
- Explicitly stating the context that the claims hold true, and any assumptions made

In doing so, the visibility of the areas of weakness in evidence or assumptions, and potentially in safety, are highlighted. Lack of evidence doesn't directly mean lack of safety, but it does mean that uncertainty about safety.

In determining the areas of weakness in a safety case during its development, it allows the organisation to closely examine and attempt to quantify the likelihood and probability of that breach in safety, and thus rationalise the resources to the areas of highest risk. In an integrated health care software system where data flows from one area of the organisation to another and is used to make clinical judgement, assurances must be made of the quality, reliability and accessibility of that data at the time that a clinical decision needs to be made. An explicit safety case allows assurance to the receiver, and interpreter, of the information outlining the providence and dependability of the information received, and whether or not it can be used for key decisions in a particular clinical scenario. Understanding the key areas of strengths and potential weaknesses in safety in a software system, does give the organisation the ability to revise, or create new policies. These can then be used to bridge the gap with human processes and where required develop additional safety checklists.

C. Developer

Establishing a safety case can help the developer substantiate the guarantees they offer to the deploying organisation. The safety case allows the deploying organisation to understand the claims made about the operation of the system and the operational context in which these claims were made. Furthermore, the safety case will highlight any potential gaps (assurance deficits) that may be present; for example, insufficient evidence to support a claim. The developer will then be able to plan at an appropriate time the generation of further evidence or to re-evaluate their design.

VII. SUMMARY

A safety case exists to communicate an argument. It is used to demonstrate how someone can reasonably conclude that a system is acceptably safe from the evidence available. A safety case consists of an argument (claims and inferences) and evidence supporting it. Evolving in parallel to the system development, the argument claims are decomposed until they can be directly supported by evidence collected during the development and testing phases of the system. Creating an argument about the achievement of safety in retrospect, after the end of the system development process can be problematic. The safety case consists of a number of self-contained arguments: the hazard management argument, the confidence argument and the compliance argument. Each of the arguments contains a set of evidence that has been considered suitable to support the claims in the argument. A safety case can provide a useful insight to the safety related reasoning for each system stakeholder, namely the regulator, the developer and the operator of the system.

Regardless of whether a safety case is imposed by any regulatory framework, it should be seen as a vehicle to systematically structure available information about safety in a clear and comprehensible way, supported by evidence, thus resulting in identifying and subsequently reducing areas of high risk or uncertainty about the risk during the operation of a health IT system.

ACKNOWLEDGMENTS

This work was carried out as part of the Large Scale Complex IT Systems (LSCITS) project, - http://lscits.cs.bris.ac.uk/

REFERENCES

[1] Kohn T. L., Corrigan M. J., Donaldson S. M., (Editors), To Err is Human: Building a Safer Health System, Committee on Quality of Health Care in America, Institute of Medicine, National Academy Press, 2000

[2] Kelly T., Arguing Safety, a Systematic Approach to Managing Safety Cases. PhD Thesis, Department of Computer Science, University of York, 1998

[3] NHS CfH, Health informatics — Guidance on the management of clinical risk relating to the deployment and use of health software Formerly ISO/TR 29322:2008, http://www.isb.nhs.uk/documents/isb-0160/dscn-18-2009

[4] NHS CfH, Health Informatics — Application of clinical risk management to the manufacture of health software Formerly ISO/TS 29321:2008(E) DSCN14/2009, http://www.isb.nhs.uk/documents/isb-0129/dscn-14-2009/

[5] Oxford English Dictionary, Oxford Press, http://www.oed.com

[6] Despotou G., Kelly T., An Argument Based Approach for Assessing Design Alternatives and Facilitating Trade-offs in Critical Systems, Journal of System Safety Vol.43 No.2 March-April 2007, System Safety Society, ISSN-0743-8826

[7] Ministry of Defence, JSP 430 - Ship Safety Management System Handbook, MoD 1996

[8] Kelly T., Habli I., Safety Case Depictions vs. Safety Cases - Would the Real Safety Case Please Stand Up?, 2nd IET International Conference on System Safety, published by the IET, November 2007, London

[9] Weaver R. , Despotou D., Kelly T., McDermid J, Combining Software Evidence - Arguments and Assurance, Workshop in Realising Evidence Based Software Engineering (REBSE),, ACM 2005. ACM SIGSOFT Software Engineering Notes , Volume 30 , Issue 4 (July 2005), ISBN:1-59593-121-X