2018 IEEE 31st International Symposium on Computer-Based Medical Systems

Introducing Traceability Information Models in Connected Health Projects

Paulo E. S. Barbosa, Fabio Leite,
Alex F. Figueiredo and Katia Galdino
Center for Strategic Health Technologies – NUTES,
Campina Grande, PB. Brazil
{paulo.barbosa, alex.figueiredo, fabio.leite,
katia.galdino}@nutes.uepb.edu.br

Danilo F. S. Santos
Embedded Systems and Pervasive Computing Lab.,
Federal University of Campina Grande,
Campina Grande, PB. Brazil
danilo.santos@embedded.ufcg.edu.br

***Abstract*— In software-intensive products, evidences of regulation conformance and failure propagation mitigation are required. The use of Traceability Information Model (TIM) is an important practice to support system engineering artifacts tracing and links, becoming part of a project traceability planning. Despite the relevance of traceability in terms of guaranteeing quality attributes and certification of these products, improvements creating a link between safety and systems engineering artifacts have not been focus of attention in the connected health industry. The lack of reference practices for adequately applying traceability has hindered product certification. In this work, we move towards introducing a TIM practice for integrated functional and safety specifications in connected health projects. To achieve this, we performed a case study involving an infusion control system. Evidences show that considerable advantages can be obtained by introducing a TIM into the modeling process. Therefore, the main contribution of this paper is to present an approach for the connected health industry to start dealing with traceability challenges.**

Keywords- Connected Health; Software Traceability; Safety.

I. INTRODUCTION

It is reasonable to say that the combination of software, systems, and safety engineering methodologies in the medical industry still lags what is found in several other critical industries, such as avionics or automotive. However, regulations are changing their understanding regarding the software documentation of medical devices and more sophisticated techniques are currently being required by industry [1]. In this context, the documentation required by standards such as IEC 62304 [2] demands architectural specifications of systems and risk management processes, where several layers, viewpoints, and concerns should be addressed by designers and safety engineers [3].

In the scenario we are presenting this paper, we use the example of infusion control systems, which are responsible for precisely delivering drugs and/or other fluids to patients under certain critical conditions. Several identified risks are associated with the technologies employed in the design of these devices, and mitigation strategies for hazards have been a hot topic of discussion in recent years [4]. With the advances in connected health applications, manufacturers are trying to incorporate best candidate technologies for dealing with remote care and newly emerging hazards that may have a direct impact on all the device characteristics as well as on system integration.

In this work, we go further in the discussion about regulatory issues regarding connected health by performing a case study on infusion control systems for remote care. We show how a functional architectural specification can be released concerning mandatory safety issues. This specification intends to support the use of an identified Traceability Information Model (TIM) to support essential requirements when showing compliance with medical software standards. Such research is justified because current connected health regulations still do not allow any automated control of infusion systems or their settings, or significant reformatting of data [4] [5].

Standards like IEC 62304, require demonstration of traceability between system requirements, software requirements, software system test, and risk control measures implemented in the software. However, monitoring and interpreting the effects of adverse events using artifact traceability is not a trivial task [6] [7] [8].

In this sense, this work also informs the reader about how important it is for development methodologies to consider hazards and tracing information to user errors in order to derive risk mitigation policies, especially at earlier stages. This could bring benefits such as accelerated certification processes. It is also important to start formalizing best practices and avoiding replication of data that could induce errors, especially at later stages.

This paper is structured as follows. Section II gives the necessary background and discusses the main related works. Section III describes the problem of the lack of traceability solutions for the connected health domain. Section IV defines the proposed approach to establish traceability. Finally, in Section VI we draw conclusions and present challenges regarding the introduction of traceability.

II. BACKGROUND AND RELATED WORK

According to [9], traceability is the degree to which a relationship can be established between two or more products of the development process, especially products having a predecessor-successor or master-subordinate relationship with one another. Several approaches exist that focus on establishing traces among different types of artifacts, such as requirements traceability [10] [11], requirements-to-architecture [12] [13] [14], requirements-to-code [15], and so on.

2372-9198/18/$31.00 ©2018 IEEE
DOI 10.1109/CBMS.2018.00011

Traceability plays a fundamental role in establishing consistency among the project artifacts in the software development process and providing evidence that system specifications and implementations address identified hazards and their risk control measures. For example, in [16], the author proposes a set of techniques and approaches based on traceability models to increase requirements consistency and completeness for safety-critical systems.

Despite traceability's fundamental role in projects, the definition and maintenance of trace links have challenged industry and academy [17] [18]. In [8], the authors analyzed common traceability problems in several projects and defined six practices for strategic traceability: use of a TIM, tool support, model traceability queries, trace visualization, and continuous evaluation.

In the healthcare domain, plenty of actions are being taken to define standards and application scenarios for connected health systems. Lee et al. [19] introduce the concept of medical cyber-physical systems (MCPS), which are a combination of embedded software controlling devices, networking capabilities, and the complicated physical dynamics exhibited by patients' bodies. Several achievements have been made in this area, but traceability between the produced artifacts is still missing. In [20], architectural requirements for an integrated clinical environment (ICE) were defined, which are responsible for coordinating medical devices from different vendors and provide a conceptual architecture for MCPS. Some of the most common applications are data acquisition, safety interlocks, system integration, and distributed closed loop control. For example, OpenICE [21] and the MDCF [22] are integration platforms for medical applications and have been used in academic and industry research on plug & play medical systems.

The safety assurance community has started to focus on the topic of safety assurance for medical devices. The work developed in [23] defines a set of patterns for safety assurance cases and their arguments for model-based development, and [24] proposes a methodology for enhancing Model-Based System Engineering (MBSE) practices from the safety perspective to encourage the use of safety assurance cases and to provide guidance on how to show the correspondent traceability for the development artifacts. In [25], the authors propose a platform-oriented ecosphere with specific certification processes designed to support the assurance of plug & play medical systems and define assurance argument patterns that exploit the design of a so-called ecosphere.

III. TRACEABILITY IN CONNECTED HEALTH

A TIM is an important piece when strategically planning traceability in early development phases and documenting it [8]. Traceable artifacts are generally software items, such as code classes, test cases, and requirements, plus failure propagation elements, etc.

In our prospective analysis, we included our previous experiences with standalone scenarios involving medical devices (not connected health), where our tools developed for traceability have proven to be a hot topic for slightly different applications [26] [27] [28]. However, although this practice is feasible, the problem is that the connected health industry imposes challenges that arise from the characteristics of being extremely multidisciplinary and because hazard analysis covers a long and diverse list of items, where the role of the software is not clear in every case. For example, regulators should ask questions about whether there are implantable products in the integrated clinical environment, whether they are in direct contact with the patient, whether energy is released or extracted from the patient, whether substances are released or extracted, whether measurements are realized, etc. In this sense, the experience reported by some industrial partners and regulators shows that traceability for new connected scenarios also involves a product's usage location, its state, whether it is being used in a critical hospital environment or in homecare. All these items contribute to confusing the analyst who is responsible for defining a TIM.

Consider the safety interlock use case scenario. It aims to mitigate the risk of *overinfusion* by defining the wrong dosage in an infusion pump, as illustrated in Figure 1. In one use case scenario, a physician may prescribe drugs to be injected for a certain period. After inserting the infusion data into the infusion pump, the pump sends all the information to the Integrated Clinical Environment (ICE) so that the infusion can be evaluated by the ICE using historical data from the EHR (Electronic Health Records). Since all applications are part of a distributed architecture, which is coordinated by the ICE, the information inserted by the EHR application can be accessed by other authorized applications. One of these applications may use the information about the amount of the drug and the period of infusion to calculate the appropriate infusion rate and use that data to control the infusion process.

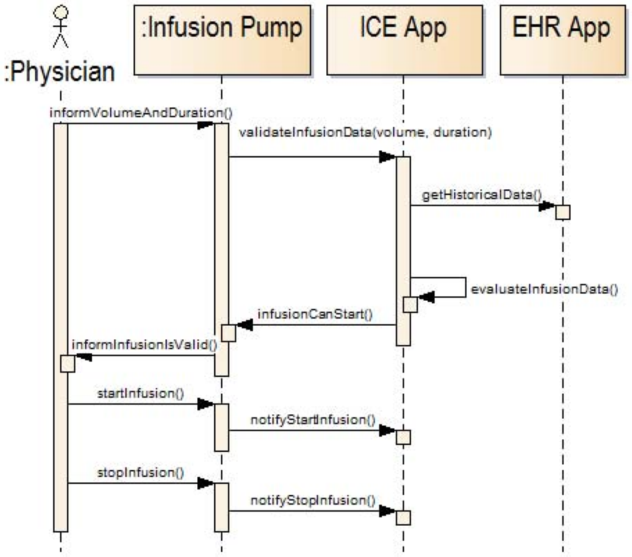

Figure 1. Remote infusion data evaluation diagram.

Thus, the main problem is that tracing artifacts is not straightforward in the connected health domain. Although safety-critical standards establish the role of traceability as a fundamental tool for providing evidence of good manufacturing practices and safety assurance, the lack of means for adequately applying traceability has hindered product certification of new platforms for medical devices that provide remote monitoring and actuating. This is confusing regulatory agencies due to the large number of heterogeneous technologies involved in connected health domain, as this example shows. Therefore, managing traceability in such scenarios is a difficult task.

Solving the problem of maintaining artifact traceability would bring the following benefits, among others:

1. Compliance with safety standards: in the connected health sector, medical devices must also comply with safety standards to allow operation. Traceability can be a means for the manufacturer to show fulfillment of the requirements of the standards. System suppliers might have to explicitly provide traceability specifications as part of the evidence of compliance.
2. Safety assurance: a fundamental need for any medical device, regardless of whether it must comply with some specific safety standard, is to ensure that its hazards have been mitigated (or avoided). Otherwise, system safety might not be achieved. Maintaining traceability is essential for showing that hazard mitigation strategies have been developed, validated, and verified.

Therefore, we demonstrate, through a case study, that a TIM is a suitable tool to support functional architectural specification and solve mandatory safety issues. Thus, it carries the objective of developing a model-based approach that could be used to manage traceability into any development process of a connected health certifiable system. For the investigation through the case study, we used the best practices proposed for conducting case studies presented in [29].

IV. An Approach for Establishing Traceability among Connected Health Artifacts

The presented approach focuses on hazard identification activities associated with the user and system operations. In this sense, we followed practices from previously proposed safety decomposition methodologies, as described in [26] [27] [28], which were all applied to standalone devices. In this section, we further explain the case study as the Health Aggregator Manager project[1] and then how we established the links between the hazard and risk assessment elements and system architecture artifacts.

A. *Case study: specification of scenarios and realization*

According to the presented simplified architecture in the Figure 2, the *Health Aggregator Manager* plays the central role in the architecture. It is responsible for gathering and integrating data from the different devices (personal health, medical devices, etc.) and dispatch commands for the medical devices.

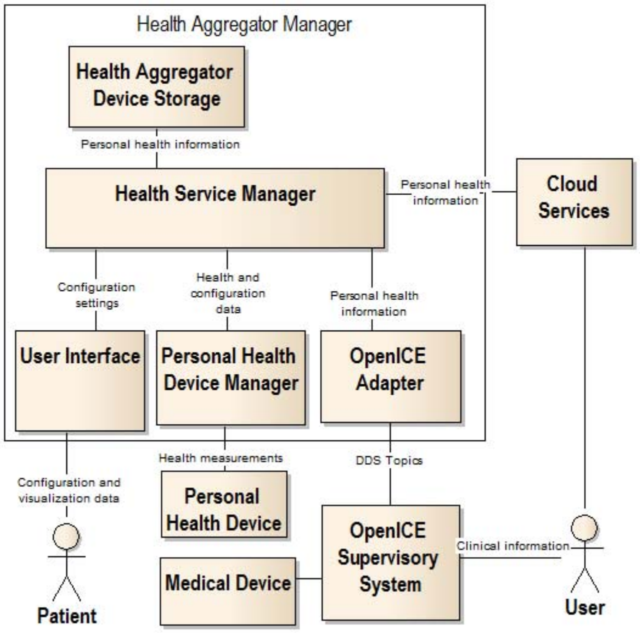
Health Aggregator Manager
Health Aggregator Device Storage
Personal health information
Health Service Manager
Personal health information
Cloud Services
Configuration settings
Health and configuration data
Personal health information
User Interface
Personal Health Device Manager
OpenICE Adapter
Health measurements
DDS Topics
Configuration and visualization data
Personal Health Device
OpenICE Supervisory System
Clinical information
Medical Device
Patient
User

Figure 2. Functional architectural specification overview.

Patients and caregivers are the main actors in the system context. Patients are connected to the system throughout a set of *Personal Health Device* or *Medical Device* (in the example, it is an infusion pump). Patients also can configure and visualize data from the *User Interface*. The configuration settings obtained from the *User Interface* are processed by the *Health Service Manager*, which, through managers (*Personal Health Device Manager* and *OpenICE Adapter*), obtains and processes health measurements and vital sign data, respectively. The *Health Service Manager* sends personal health information to a *Health Aggregator Device Storage*. Then it integrates such personal health information with a third-party supervisory system, namely *OpenICE Supervisory System* (instance of ICE App from Figure 1), in order to communicate with solutions from the Medical Device Plug and Play initiative.

As the main architectural scenario, we present the *Infusion Control Programmer* application (*User Interface* instantiation). The main application's aim is to program the infusion pump settings according to previously gathered data and *Medical Device Manager*. Once the infusion is configured, the infusion pump sends a logging event to the *Medical Device Manager* to notify it that the infusion has been started for the patient. The *Infusion Pump Programmer* processes all the alarms and solves possible issues with the pump.

B. *Deriving traceability through the proposed TIM*

To establish appropriate traceability between safety assurance artifacts and system engineering artifacts, we use our findings in the design of new prototypes under

[1] nutes.uepb.edu.br/ham

development, applying our modeling productivity tools and the NUTES Quality Model for medical devices [3].

During the hazard and risk analysis phase, engineers develop a complex set of models, documents, and artifacts. From among these, we have evaluated a minimal subset to define traces to the systems engineering artifacts. First, given the high complexity of connected health systems, we identified that a set of environmental assumptions must be made by the safety engineer. In the following, we list some of those identified in the case study.

- *Environmental Assumptions*: main system hypothesis from which further artifacts should be derived:

  [EA1]: There is high integrity, authenticity, and confidentiality of the communication channels between the infusion control system, the ICE instance (openICE), the Integration System, and other medical devices;
  [EA2]: The Integration System clock is periodically synchronized with the ICE, allowing decisions to be correct based on timestamps;
  [EA3]: The measurements are sent to the Integration System to be decoded;
  [EA4]: Other systems that should be connected to the infusion control system also have safety management policies according to medical device standards;
  [EA5]: Safety interlock situations are identified and managed by the ICE application.

Hazard identification techniques and system engineering practices strongly rely on precise identification of the system context. Although establishing traces between the system context and accident elements is an open gap not addressed by this work, we have concrete evidence showing its importance [26] [27].

In the next step of the TIM definition, after identifying hazards, analyzing their causes, and assessing the related risks, the next challenge for the safety engineers is to express the safety goals and safety requirements to mitigate the system's usage risks. The system engineers accordingly define the architectural decisions to implement the safety design strategies needed to realize the safety requirements. In the following we show excerpts filtered from the safety analysis.

- *Accidents*: undesired or unplanned events that result in loss, including human lives, mission failure, environmental damage, etc.:
  [A1]: Death or health damage caused to the patients by the actuation of the infusion control system;
  [A2]: Inconsistency of the data collected by the infusion control system regarding the patient profile and the associated therapy;
  [A3]: Infusion control damage;
  [A4]: Leakage of confidential information about patients, health professionals, and hospitals;
  [A5]: Loss of information about the measurements collected from the patients.

- *Hazard*: a system state or set of conditions that together with worst-case environmental conditions could lead to an accident:
  [H1]: The interval between infusion requests is lower than the minimum allowed interval, leading to [A1];
  [H2]: Incompatibility of protocols for data exchange between the Integration System, the infusion control system, and other medical devices, leading to [A2];
  [H3]: Inability to detect with which patient a measurement is associated, leading to [A2] and [A4];
  [H4]: Inability to detect the timestamp of a measurement, leading to [A3];
  [H5]: Low quality of service of the medium, leading to [A2] and [A5].

- *Safety Requirements*: requirements that shall be assured by the system to show an acceptable safety integrity level while the system performs its intended mission:

  [SR1]: The infusion control system shall not allow infusion requests to be accepted in shorter intervals than the minimum allowed, avoiding [H1];
  [SR2]: The Integration System shall identify and flag which medical devices with different data exchange protocols are available, avoiding [H2];
  [SR3]: The Integration System shall be capable of performing automated test routines periodically, identifying damaged medical devices, avoiding [H3];
  [SR4]: The Integration System shall discard measurements without timestamp identification and request new measurements. The system should stop the operation if it reaches a predefined number of unidentified timestamp measurements, avoiding [H4];
  [SR5]: The performance of the communication channels shall be monitored and the operation of the infusion control system shall be finalized in terms of measurable attributes with lower performance according to predefined parameters, avoiding [H5].

For instance, in Figure 3, using elements that will be incorporated to the TIM identified during the hazard and risk analysis phase. All the stereotypes represent concepts that must be traceable during the project. We identified a hazard associated with overinfusion caused by too frequent bolus requests by the patient. Requesting bolus is part of a foreseeable sequence of events specified for a use case scenario of the infusion pump. Such a foreseeable sequence exposes the patient to a hazardous event, namely Delivery of bolus infusion. When we analyze possible consequences, in the worst-case scenario, this situation might result in harm to the patient's health. In Figure 3, there are two possible kinds of harm that may be the result of a related accident.

On the other hand, hazards also need to have their causes investigated. Particularly when we look at the causes of the considered hazard, we identified that the main cause is the patient's behavior (requiring too many bolus doses). Therefore, the system should prevent this situation through safety measures or mechanisms such as the assurance of a minimal time interval between two bolus deliveries. Safety engineers have to state a clear safety requirement to avoid this situation. For example, the safety requirement in Figure 3 establishes that the system shall not allow infusion releases at time intervals shorter than a minimum defined by a physician. Consequently, system engineers defined an architectural decision to meet the safety requirement by keeping a log with timestamps for all infusion releases and requiring the system to consider these timestamps in order to calculate the time between the releases.

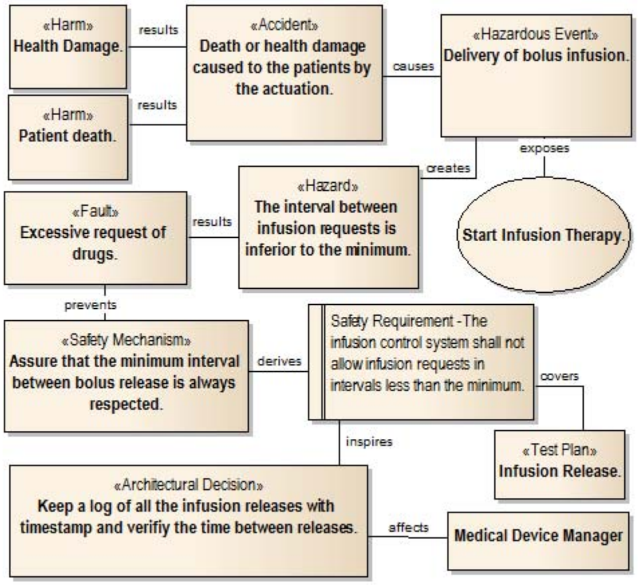
«Harm»
Health Damage.
results
«Accident»
Death or health damage caused to the patients by the actuation.
causes
«Hazardous Event»
Delivery of bolus infusion.
«Harm»
Patient death.
results
exposes
creates
«Hazard»
The interval between infusion requests is inferior to the minimum.
Start Infusion Therapy.
«Fault»
Excessive request of drugs.
results
prevents
«Safety Mechanism»
Assure that the minimum interval between bolus release is always respected.
derives
Safety Requirement - The infusion control system shall not allow infusion requests in intervals less than the minimum.
covers
inspires
«Test Plan»
Infusion Release.
«Architectural Decision»
Keep a log of all the infusion releases with timestamp and verifiy the time between releases.
affects
Medical Device Manager

Figure 3. Demonstration of traceability analysis for a predicted accident.

Figure 4 presents a general overview of the TIM extracted from the experiences collected in this work. We illustrated the extraction of its main structure through the association of Hazards with Architectural Decisions. The reader should note that after extracting this structure, it is clear how to add other system and software engineering artifacts such as Test Plan, Test Log, System Module, and Class. It also include patterns identified by us in the main medical device standards, such as ISO 14971, which is the main risk analysis compliance standard required by regulators worldwide. The TIM could be further refined and it is ready to be associated with domain specific items from the connected health domain. In this sense, the model in Figure 3 becomes an instance of the TIM presented in Figure 4, with specific connected health artefacts requested when showing compliance with ISO 14971 (*e.g.* Accident, Hazardous Event, Foreseeable Sequence, and Harm). A toolset using UML was implemented as an Add-in for the Enterprise Architect, intending to support software engineers when specifying such models [28].

In this case study we used artifacts from technological transfer processes at the Brazilian public research institute NUTES (Center for Strategic Health Technologies), which is currently acquiring capabilities for manufacturing connected medical devices. In this scenario, new tools and development methodologies are necessary and are under development. These will incorporate previous practices from existing products and evolve them in various ways with regard to connected health and technological advances. As an example of this evolution, one issue being dealt with is connectivity according to the ISO/IEEE 11073 standard, following the Continua Health Alliance[2] design guidelines. Thus, an efficient approach for managing existing artifacts, dealing with newly introduced hazards, and handling cyber-physical environments and certification requirements is mandatory. The TIM is supporting directly in the definition association of development artefacts with ontologies from the health domain.

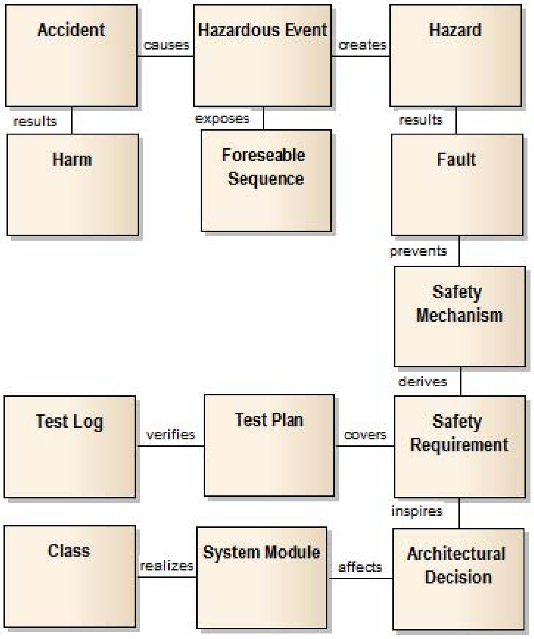

Figure 4. The identified traceability information model.

V. Conclusions

This paper presented a model-based approach for artifact traceability management in the context of connected health applications. The presented approach was evaluated in a concrete case study throughout a remote infusion control system and provided efficient support to trace artifacts related to hazard analysis, risk assessment, requirements, architectural elements, and verification. This made it possible to integrate contextual risk management specifications involving software and hardware with the TIM specification of a project. The traces established in the hazard analysis process allows keeping engineers informed about how simple architectural elements, regardless of whether they are realized in software or in hardware, affect the general safety of the system. These findings bring benefits for safety assurance and certification by providing the evidences required by regulatory agencies and standards, which is an issue that, most of the time, is considered very abstract for the software engineering field.

We claim this work is rich in this sense and we expect to be providing a good contribution to academic research by showing how these multidisciplinary issues can be

[2] http://www.pchalliance.org

addressed. Regarding further developments, this traceability information model is also being explored in other scenarios.

This approach is currently being employed in the OCARIoT project[4], that uses the IoT potential to develop obesity coaching plans for children and educators. In future work, we will focus on refinement of the models and supporting tools based on the TIM. Moreover, we intend to enhance the approach and the models in to a derive a generic framework based on the experience made in applying them in several connected healthcare scenarios.

Acknowledgments

The research has been performed under OCARIoT project, funded from the European Union`s HORIZON 2020 Programme (2014-2020), ID 777082, and from the Brazilian Ministry of Science, Technology and Innovation through Rede Nacional de Ensino e Pesquisa (RNP), ID 003008.

REFERENCES

[1] P. Barbosa, M. Morais, K. Galdino, M. Andrade, L. Gomes, F. Moutinho, and J. Figueiredo, “Towards medical device behavioural validation using petri nets,” in Proceeding of the IEEE CBMS’13, 2013.

[2] ANSI/AAMI/IEC 62304:2006, "Medical Device Software—Software Life Cycle," Assoc. Advancement Medical Instrumentation, 2006.

[3] F. Leite, P. O. Antonino, P. Barbosa, S. Kemmann, and R. Mendonca. Are the current architectural practices suitable for safety aspects of medical devices? an exploratory investigation. In Healthcom 2014, pages 477–482, 2014.

[4] J. T. James, “A new, evidence-based estimate of patient harms associated with hospital care,” Journal of patient safety, vol. 9, no. 3, pp. 122–128, 2013.

[5] J. Hatcliff, A. Wassyng, T. Kelly, C. Comar and P. Jones (2014) Certifiably safe software-dependent systems: challenges and directions, in FOSE 2014, Hyderabad, India, 2014.

[6] J. Cleland-Huang, H. Mats, J. Huffman Hayes, R. Lutz and P. Mäder (2012) Trace queries for safety requirements in high assurance systems, in REFSQ'12, Essen, Germany, 2012.

[7] M. P. E. Heimdahl, "Safety and Software Intensive Systems: Challenges Old and New," in FOSE '07 - ICSE 2007, Minneapolis, 2007.

[8] Patrick Mäder, Paul L. Jones, Yi Zhang, and Jane Cleland-Huang (2013) Strategic Traceability for Safety-Critical Projects. IEEE Software 30(3), pp. 58-66, 2013.

[9] IEEE: IEEE Standard Glossary of Software Engineering Terminology. (1990).

[10] Kannenberg A, Saiedian H (2009) Why Software Requirements Traceability Remains a Challenge. Cross Talk: The Journal of Defense Software Engineering 22 (5):14 -17.

[11] O. Gotel and A. Finkelstein, “An analysis of the requirements traceability problem”, RE 1994, pp. 94-101.

[12] G. Spanoudakis and A. Zisman, “Software traceability: a roadmap”, in Handbook of Software Eng. and Knowledge Engineering, 2005.

[13] P. O. Antonino, M. Trapp and A. Venugopal, "Automatic Detection of Incomplete and Inconsistent Safety Requirements," in SAE International, Detroit, USA, 2015.

[14] Rempel, P. et al.: Mind the gap: assessing the conformance of software traceability to relevant guidelines. Proc. 36th Int. Conf. Softw. Eng. - ICSE 2014. 943–954 (2014).

[15] Mäder, P., Egyed, A.: Do developers benefit from requirements traceability when evolving and maintaining a software system?, (2014).

[16] Improving the Consistency and Completeness of Safety Requirements Specifications PhD Theses in Experimental Software Engineering , Band 55. Pablo Oliveira Antonino de Assis Hrsg.: Fraunhofer IESE, Kaiserslautern 2016, 232 p., num. illus. and tab., Kartoniert Sprache: engl.

[17] J. Cleland-Huang, O. Goetel, J. Huffman Hayes, P. Maeder and A. Zisman, "Software traceability: trends and future directions," in Future of Software Engineering - FOSE 2014, Hyderabad, India, 2014.

[18] Nair, S. et al.: A review of traceability research at the requirements engineering conferencere@21. 2013 21st IEEE Int. Requir. Eng. Conf. RE 2013 - Proc. 222–229 (2013).

[19] Insup Lee et al.: Challenges and Research Directions in Medical Cyber-Physical Systems. Proc. IEEE. 100, 1, 75–90 (2012).

[20] Hatcliff, J., King, A., Lee, I., MacDonald, A., Fernando, A., Robkin, M., Vasserman, E., Weininger, S., Goldman, J.M.: Rationale and architecture principles for medical application platforms. In: 2012 IEEE/ACM Third International Conference on Cyber-Physical Systems (ICCPS), pp. 3–12. IEEE (2012).

[21] Plourde, J. et al.: OpenICE: An open, interoperable platform for medical cyber-physical systems. In: 2014 ACM/IEEE International Conference on Cyber-Physical Systems (ICCPS). pp. 221–221 IEEE (2014).

[22] King, A. et al.: Prototyping closed loop physiologic control with the medical device coordination framework. In: Proceedings of the 2010 ICSE Workshop on Software Engineering in Health Care - SEHC ’10. pp. 1–11 ACM Press, New York, New York, USA (2010).

[23] Ayoub, A., Kim, B.G., Lee, I., Sokolsky, O.: A safety case pattern for model-based development approach. In: Goodloe, A.E., Person, S. (eds.) NFM 2012. LNCS, vol. 7226, pp. 141–146. Springer, Heidelberg (2012).

[24] Ruiz, Alejandra; Barbosa, Paulo; Medeiros, Yang; Espinoza, Huascar. Safety Case Driven Development for Medical Devices. Lecture Notes in Computer Science. 1ed.: Springer International Publishing, 2015, p. 183-196.

[25] King, A.L. et al.: Towards Assurance for Plug and Play Medical Systems. In: 34th International Conference, SAFECOMP. pp. 228–242 Springer International Publishing, Delft, The Netherlands (2015).

[26] Antonino, Pablo; Trapp, Mario; Barbosa, Paulo; Sousa, Luana. The Parameterized Safety Requirements Templates. In: Proceedings of 8th ICSE International Symposium on Software and Systems Traceability, Firenze, Italy, 2015.

[27] Antonino, Pablo; Trapp, Mario; Barbosa, Paulo; Gurjao, Edmar; Rosario, Jeferson. The Safety Requirements Decomposition Pattern. Lecture Notes in Computer Science. 1ed.: Springer International Publishing, 2015, p. 269-282.

[28] Barbosa, Paulo E.S.; Leite, Fabio; Mendonca, Raphael; Andrade, Melquisedec; Sousa, Luana; Antonino, Pablo O. RAwTIM – A Tool for Traceability during Risk Analysis (Uma Ferramenta para Rastreabilidade da Informação em Análises de Riscos). In: Brazilian Conference on Software: Theory and Practice – Tools Section, 2015, Belo Horizonte, Brazil. Proceedings of Brazilian Conference on Software: Theory and Practice, 2015.

[29] P. Runeson and M. Höst (2009) Guidelines for conducting and reporting case study research in software engineering, Empirical Software Engineering, vol. 14, no. 2, pp. 131–164, Apr. 2009.

[4] ocariot.com