A Quality Requirements Safety Model for Embedded and Real Time Software Product Quality

KHALID T. AL-SARAYREH
Department of Software Engineering
Hashemite University
Zarqa 13115, Jordan
khalidt@hu.edu.jo

Abstract— safety requirements quality is typically expressed at the software system levels, which may lead to specific safety-related functions that may be applied in either or both hardware and software. According to the Previous work of academia and industry standards such as ISO 25021 square series safety requirements can be defined and measured internally and externally; the internal safety concepts could include control software hazards, safety levels of software integration and critical software catastrophic, while the external safety could include software safety functions, safety failure mechanism and safety switching of redundant items. This paper collects and organizes these safety-related requirements into a quality requirements safety model for identifying and evaluation of the quality safety requirements of embedded and real time systems..

Keywords: Safety Requirements, Software Product Quality, ISO 2502n Squares

1. Introduction

A quality requirements model for safety is a key to the success of the embedded and real time software systems. The development of such quality model of requirements early in the software life cycle is therefore of prime importance by defining a comprehensive specification and evaluation of software product quality.

The qualities of the safety requirements as currently defined can be considered subjective since they can be viewed, interpreted and evaluated differently by different people; when the safety requirements are stated briefly and vaguely the problem is compounded [1]. Safety requirements and their qualities can also be defined relatively, since the interpretation and importance of such kind of safety requirements may vary depending on the particular system being considered.

Furthermore, if the safety requirements are not addressed, then it may lead to [2] "Software which is inconsistent and of poor quality. User, developer and clients who are unsatisfied and time and cost overruns to fix software, which was not developed with such requirements".

However, safety requirements may also impact considerably project effort and should also be taken into account when doing project benchmarking. It is however challenging to take these requirements into account in software estimation and software benchmarking.

The aim of this paper is to propose a procedure for describing, and next quality, software safety using a strategy based neither on our own views nor on individual researchers view of such type of safety requirements, but on a consensual view documented in international standards of software safety as quality requirements.

For the purpose of this research, the set of European standards have been selected: [3-6], ISO 2502n [7] square standards and a previous published work for academia.

This paper is organized as follows. Section 2 presents related work. Section 3 presents a standard-based procedure to develop a model of requirement for software safety. A conclusion is presented in Section 4.

2. Related Work

The European Standard series [3-6] present software safety requirement for real-time and embedded software: in these standards, the safety requirements are described as system state where an acceptable level of risk is not exceeded with respect to fatality, injury or occupational illness, damage to launcher hardware or launch site facilities, damage to an element of an interfacing manned systems, the term “safety” is described differently in ISO/IEC Guide 2, as “freedom from unacceptable risk or harm”. The ISO 8402 [8] was replaced by ISO 9000, but the definition of safety was not maintained.

According to safety requirements shall be identified and traced from the system level into the

design and then allocated to the lower levels as well as the identified safety requirements shall be justified in the design and presented in an appropriate document. [3-4] describe the the mandatory aspects for safety requirements of a system safety programme to ensure that all safety risks associated with the design, development, production and operations of space product are adequately identified, assessed, minimized, controlled and finally accepted through the implementation of a safety assurance programme.

The [3-6] safety policy is applied by implementing a system safety program, supported by risk assessment, which can be summarized as follows:

- Hazardous characteristics (system and environmental hazards) and functions with potentially hazardous failure effects are identified and progressively evaluated by iteratively performing systematic safety analyses;
- The potential hazardous consequences associated with the system characteristics and functional failures are subjected to a hazard reduction sequence whereby: hazards are eliminated from the system design and operations; hazards are minimized and hazard controls are applied and verified.
- The risks that remain after the application of a hazard elimination and reduction process are progressively assessed and subjected to risk assessment, in order to: show compliance with safety targets; support design trade-offs; identify and rank risk contributors; support apportionment of project resources for risk reduction; assess risk reduction progress; and support the safety and project decision-making process (e.g. waiver approval, residual risk acceptance).
- The adequacy of the hazard and risk control measures applied are formally verified in order to support safety validation and risk acceptance;
- Approval obtained from the relevant authorities.

ISO series [9] defined Safety specifications as equipment/system design features, performance specifications, and training that reduce the potential for human or machine errors or failures that cause injury or death within the constraints of operational effectiveness, time, and cost throughout the equipment/system life cycle as well as describe the Safety Plan as the approach and methods for conducting safety analysis and assessing the risk to operators, the system, the environment, or the public.

ISO standards [7-9] list the Safety measurements to assess the level of risk of harm to people, business, software, property or the environment in a specified context of use. It includes the health and safety of the both the user and those affected by use, as well as unintended physical or economic consequences.

The IEEE Standard for Software Safety described software safety as falls into one or more of the following categories:

- Software whose inadvertent response to stimuli, failure to respond when required, response out-of-sequence, or response in combination with other responses can result in an accident
- Software that is intended to mitigate the result of an accident
- Software that is intended to recover from the result of an accident

However, neither ECSS nor IEEE series propose a way to measure such these safety requirements, while ISO 25021 presents measures of the outcome of safety management as a quality of the software product quality, not of the safety requirements that have to be built into the software thereby not allowing for measuring the functional size of such software safety requirements: without measurement it is of course challenging to take such an NFR as a quantitative input in an estimation process or in productivity benchmarking.

This paper reports on the work carried out to define safety requirements on the basis of international standards and on the safety requirements foundation results in the academia.

3. The Proposed Quality Procedure Model

This section illustrates the proposed procedure [10-16] for building a quality model for safety requirement as follows:

3.1 A Definition of Safety Requirements

- The safety requirements are defined as system state where an acceptable level of risk is not exceeded with respect to fatality [3-6].
- The safety defined as human or machine errors or failures that cause injury or death within the constraints of operational effectiveness, time, and cost throughout the equipment/system life cycle [3-6].
- The safety is the levels of risk of harm to people, business, software, property or the environment in a specified context of use [7-9].
- Safety is a freedom from software hazards or a systematic approach to reducing software risks [7-9].

3.2 Types of Safety Requirements

This section illustrates the types of safety requirements as follows:

- **Safety mandatory categories**
  - Catastrophic hazards
    - Loss of life, life-threatening or permanently disabling injury or occupational illness, loss of an element of an interfacing manned flight system.
    - Loss of launch-site facilities or loss of system.
    - Severe detrimental environmental effects.
  - Critical hazards
    - Temporarily disabling but not life-threatening injury, or temporary occupational illness.
    - Major damage to flight systems or loss or major damage to ground facilities.
    - Major damage to public or private property.
    - Major detrimental environmental effects.
- **Safety non-mandatory categories**
  - Marginal hazards
    - Minor injury, minor disability, minor occupational illness, or minor system or environmental damage.
  - Negligible hazards
    - Less than minor injury, disability, occupational illness, or less than minor system or environmental damage.

3.3 Safety Requirement Entities

This section illustrates the safety requirements entities as follows:

External entities of Safety

- Software Safety Functions
- Safety Failure Mechanism
- Safety Switching of Redundant items

Internal entities Safety

- Safety Related Software
- Safety Levels of Software Integration
- Safety Audit Software
- Control Software Hazards
- Critical Software Catastrophic

3.4 Identification of the Entity types of Safety

This section illustrates the entity types of safety requirements and as follows:

- **Entity Type 1: Software Safety Functions**
  - Each software safety function shall receive or send with at least one functional process from/to safety related software.

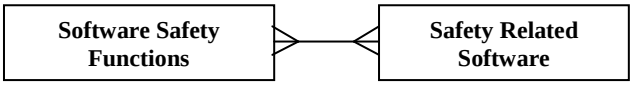

Figure 1: **Software Safety Functions**

- **Entity Type 2: Safety Failure Mechanism**
  - Each failure mechanism could send with at least one functional process to one or more safety levels of software integration.

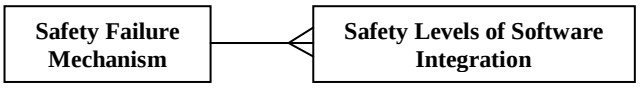

Figure 2: **Safety Failure Mechanism**

- **Entity Type 3: Safety Switching of Redundant items**
  - Each failure mechanism could send or/and receive with at least one functional process to one or more items in safety audit software.
  - Each failure mechanism could send or/and receive with at least one functional process to check one or more items in redundancy status information.

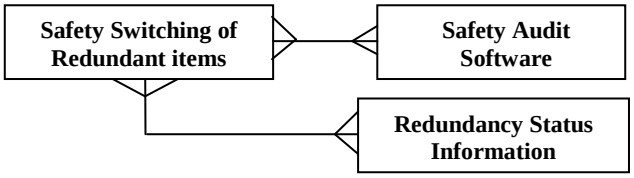

Figure 3: **Safety Switching of Redundant items**

- **Entity Type 4: Safety Related Software**
  - Each safety related software shall receive or/and send with at least one functional process from/to software failure data group.
  - Each safety related software shall receive or/and send with at least one functional process from/to the allocated failure in the safety levels of software integrations.

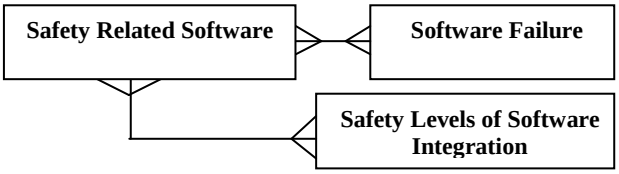

Figure 4: **Safety Related Software**

- **Entity Type 5: Safety Levels of Software Integration.**

- Each safety level of software integration shall receive or/and send with at least one functional process from/to fault tolerance data group.
- Each safety level of software integration shall receive or/and send with at least one functional process from/to safety related software to allocated the fault.
- Each safety level of software integration shall receive or/and send with at least one functional process from/to safety software audit data to check the integration with data.

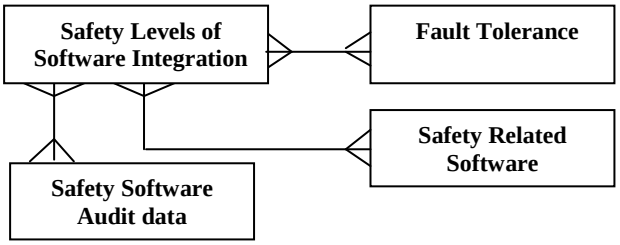

Figure 5: **Safety Levels of Software Integration.**

- **Entity Type 6: Safety Audit Software**
  - Each safety audit software shall receive or/and send with at least one functional process from/to redundancy status information.
  - Each safety audit software shall receive or/and send with at least one functional process from/to safety levels of software integration.

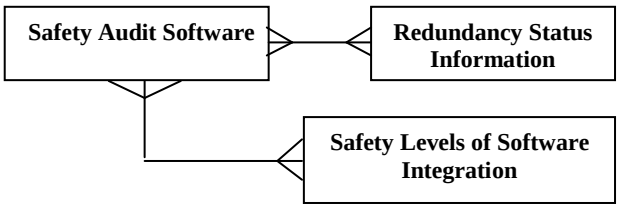

Figure 6: **Safety software audit data.**

- **Entity Type 7: Control Software Hazards**
  - Each safety control software hazards shall receive or/and send with at least one functional process from/to failure tolerance data group to control this kind of faults.
  - Each safety control software hazards shall receive or/and send with at least one functional process from/to software failure data group to control this kind of error.
  - Each safety control software hazards shall receive or/and send with at least one functional process from/to critical software catastrophic to check if the defect is harm or not.

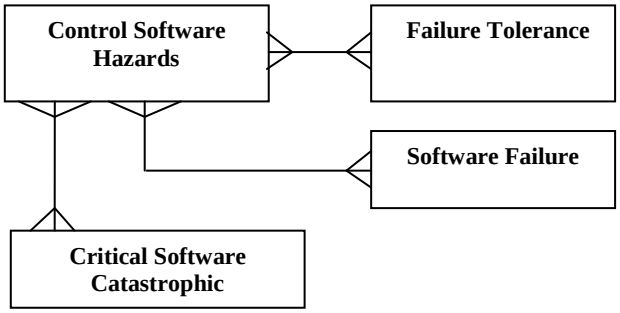

Figure 7: **Control Software Hazards.**

- **Entity Type 8: Critical Software Catastrophic**
  - Each safety critical software catastrophic shall receive or/and send with at least one functional process from/to failure tolerance data group to check this kind of faults before exchange processes with critical software catastrophic.
  - Each safety critical software catastrophic shall receive or/and send with at least one functional process from/to redundancy status information data group to check if the critical situation are caused by redundant data or not.
  - Each safety critical software catastrophic shall receive or/and send with at least one functional process from/to control software hazards to identify the source and the degree of the defects.

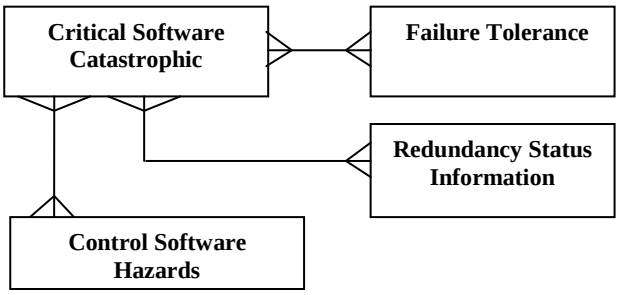

Figure 8: **Critical Software Catastrophic**

3.5 Model of the Requirements for safety requirements

In the following design of the safety requirements model:

- **Entity type 1** can be used to measure the external safety for the software safety functions from the received/send data movement from/to safety related software such as software operation, design and configuration risk- see-figure 9.

ISBN: 978-1-61804-297-2

- **Entity type 2** can be used to measure the external safety for the safety failure mechanism from the received/send data movement from/to safety levels of software integration such as loss of operation, failure detection and failure isolation - see-figure 9.
- **Entity type 3** can be used to measure the external safety for the safety switching of redundant items from the received/send data movement from/to safety software audit data and redundancy status of information such as duplicate or corrupted data - see-figure 9.
- **Entity type 4** can be used to measure the internal safety for the safety related software from the received/send data movement from/to software failure data group list and safety levels of software integration such as loss of operation, failure detection and failure isolation - see-figure 9.
- **Entity type 5** can be used to measure the internal safety for the safety levels of software integrity from the received/send data movement from/to software failure tolerance data group list and safety related software safety software audit data - see-figure 9.

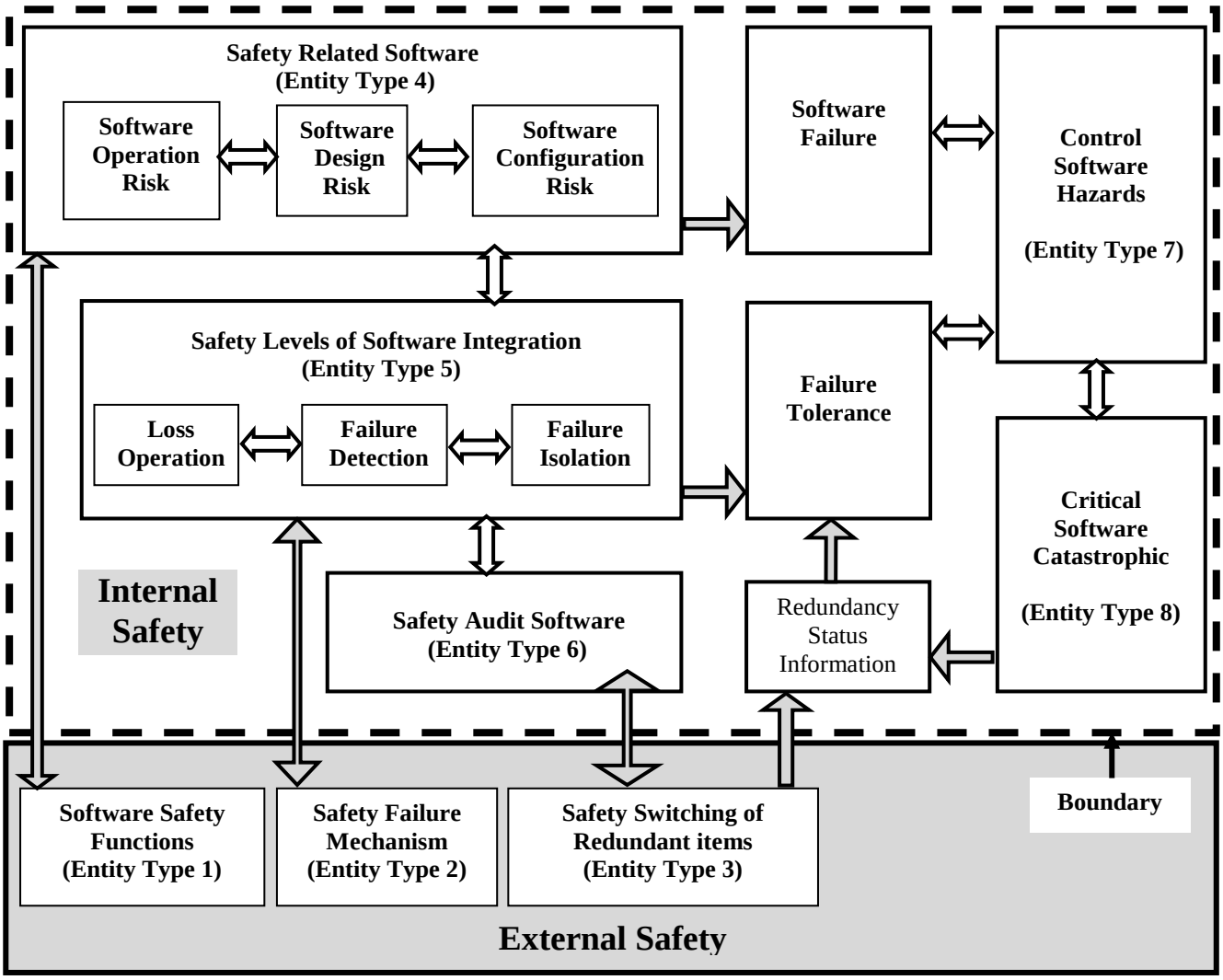

Figure 9. A Quality Requirements Safety Model

- **Entity type 6** can be used to measure the internal safety for the safety software audit data from the received/send data movement from/to safety levels of software integrity and redundancy status information data group list - see-figure 9.
- **Entity type 7** can be used to measure the internal safety for the control software hazards from the received/send data movement from/to failure tolerance data group list and critical software catastrophic - see-figure 9.

- **Entity type 8** can be used to measure the internal safety for the critical software catastrophic from the received/send data movement from/to failure tolerance data group list and control software hazards - see-figure 9.

4. Conclusion

This paper introduced a procedure for measuring requirements for internal and external safety required for system safety requirement. This included a proposed generic model for safety requirement using standard based identification of three set of international standards, this model is independent of the software type or languages in which these safety requirements will be implemented.

It is important to remark that the design measurement procedure for safety requirements for the embedded software have been developed to apply the measurement methods defined by academia to the safety requirements in order to obtain the quality of the software-safety as a separate piece of a software in early stages of the software development process.

The advantages and the limitations of the model are left as a future work to enhance the proposed model and to applicable to use it in the industry.

Furthermore, the main contribution of this paper is the proposed generic safety requirements model of the safety requirements. The proposed generic model is considered as kind of a a standard-based model that is being used for the measurement of the software-safety.

The proposed generic model of the software-safety requirements is a bridge between the system and software functional requirements to provide a basis for both describing in a standard way and in the future for the measurement of the functional size of the software based on the set of definitions and concepts of system safety requirements in European international standards as follows: The interrelations between the internal and external requirements of safety are defined, for example each process between the internal and external safety requirements and in the future to measure the functional size of software safety requirements for the all functional processes (internally and externally)

References

[1] L. Chung and J. do Prado Leite, "On Non-Functional Requirements in Software Engineering," in *Conceptual Modeling: Foundations and Applications*, Lecture Notes in Computer Science, Springer Berlin / Heidelberg, vol. 5600, pp. 363-379, 2009.

[2] W. Ma, L. Chung, and K. Cooper, "Assessing Component's Behavioral Interoperability Concerning Goals," in *On the Move to Meaningful Internet Systems: OTM 2008 Workshops*, Lecture Notes in Computer Science, Springer Berlin / Heidelberg, pp. 452-462, 2008.

[3] ECSS-E-40-Part-2B, Space Engineeing:Software-part 2 Document Requirements Definitions, *European Cooperation for Space Standardization, The Netherlands*, 2005.

[4] ECSS-ESA, Tailoring of ECSS, Software Engineering Standards for Ground Segments, Part C: Document Templates, *ESA Board of Standardization and Control (BSSC)*, 2005.

[5] ECSS-E-ST-10C, Space engineering: System engineering general requirements, *Requirements & Standards Division Noordwijk, The Netherlands*, 2009.

[6] ECSS-Q-ST-80C, Space Product Assurance: Software Product Assurance, *Requirements & Standards Division Noordwijk, The Netherlands*, 2009.

[7] ISO/IEC-2502n, Software Engineering -- Product Quality -- Part 1: Quality Model 2502, *International Organization for Standardization,Geneva (Switzerland)*, 2012.

[8] ISO 24765, "Systems and software engineering vocabulary," *British Standards Institution*, 2008.

[9] ISO 2382-1, "Information technology -- Vocabulary -- Part 1: Fundamental terms," *International Standards for Business, Government and Society*, 1993.

[10] Abran, A., K. T. Al-Sarayreh, and J. J. Cuadrado-Gallego, " A Standards-based Reference Framework for System Portability Requirements", Computer Standards and Interface, Elsevier, 2013. http://dx.doi.org/10.1016/j.csi.2012.11.003

[11] Al-Sarayreh, K. T., A. Abran and and J. J. Cuadrado-Gallego," A Standards-based model of system maintainability requirements", Journal of Software: Evolution and Process, John Wiley & Sons, Ltd, 2012. http://dx.doi.org/10.1002/smr.1553

[12] K. T. Al-Sarayreh, A. Abran, and J. J. Cuadrado-Gallego, "Measurement Model of Software Requirements Derived from System Portability Requirements," *9th International Conference on Software Engineering Research and Practice (SERP 2010)*, Las Vegas, USA, pp. 553-559, 2010.

[13] K. T. Al-Sarayreh and A. Abran, "A Generic Model for the Specification of Software Interface Requirements and Measurement of Their Functional Size," *8th ACIS International Conference on Software Engineering Research, Management and Applications, SERA 2010*, Montreal, Canada, pp. 217-222, 2010.

[14] K. T. Al-Sarayreh and A. Abran, "Measurement of Software Requirements Derived from System Reliability Requirements," *24th European Conference on Object-Oriented Programming (ECOOP 2010)*, ACM Digital Library, Maribor, Slovenia, 2010.

[15] K. T. Al-Sarayreh and A. Abran, "Specification and Measurement of System Configuration Non Functional Requirements," *20th International Workshop on Software Measurement & International Conference on Software Measurement, IWSM/Metrikon/Mensura, Stuttgart, Germany*, pp. 141-156, 2010.

[16] A. Abran, K. T. Al-Sarayreh, and J. J. Cuadrado-Gallego "Standards-based Model for the Specification and Measurement of Maintainability Requirements," *22nd International Conference on Software Engineering and Knowledge Engineering (SEKE 2010)*, Redwood City, California, USA, pp. 153-158, 2010.