Specification and Measurement Derived from System Operations Non Functional Requirements

Alain Abran, Khalid T. Al-Sarayreh

Software Engineering Department, University of Quebec (ETS)
1100 Notre-Dame West Montréal, Québec H3W 1T8, Canada

alain.abran@etsmtl.ca, khalid.al-sarayreh.1@ens.etsmtl.ca

Abstract:

An operation is typically described initially as a non functional requirement at the system level. Systems engineers must subsequently apportion these system requirements very carefully as either software or hardware requirements to conform to the operations requirements of the system. A number of concepts are provided in the ECSS and IEEE standards to describe the various types of candidate operations requirements at the system, software, and hardware levels. This paper organizes these concepts into a generic standards-based reference model of the requirements at the software level for system operations. The structure of this reference model is based on the generic model of software requirements proposed in the COSMIC – ISO 19761 model, thereby allowing the measurement of the functional size of such operations requirements implemented through software.

Keywords

Operations Requirements, Non functional requirements – NFR, Functional size, COSMIC – ISO 19761, ECSS International Standards, IEEE-830 standard, Software Operations Measurement

1 Introduction

Non-functional requirements (NFR) play a critical role in system development, including as selection criteria for choosing among alternative designs and ultimate implementations. NFR may also have a considerable impact on project effort, and should be taken into account for estimation purposes and when comparing project productivity.

Typically, these NFR are described at the system level and not at the software level, and there is no consensus yet on how to describe and measure these system NFR. In practice, NFR can be viewed, defined, interpreted, and evaluated differently by different people, particularly when they are stated vaguely and only briefly [1-3]. Therefore, it is challenging to take them into account in software estimation and software benchmarking: NFR have received less attention in the software engineering literature and are definitely less well understood than other cost factors [3] . Without measurement, it is challenging to take them as quantitative inputs into an estimation process and productivity benchmarking.

In practice, the requirements are initially typically addressed at the system level [4-10] either as high-level system functional user requirements (system FUR) or as high-level system non-functional requirements (system NFR). The latter must usually be detailed, allocated and implemented in either hardware or software, or both, as software FUR [11-15], for instance – see Fig. 1.

For example, a system FUR will describe the required functions in a system, while a system NFR will describe how the required functions must behave in a system. In the software requirements engineering step, system NFR can then be detailed and specified as software FUR to allow a software engineer to develop, test, and configure the final deliverables to system users.

The term "functional" refers to the set of functions the system (including the software) has to offer, while the term "non-functional" refers to the manner in which such functions are performed. FUR is typically phrased with subject or predicate constructions (i.e. noun/verb), such as: "The system must print 5 reports". NFR, by contrast, are typically phrased with adverbs or modifying clauses, such as: "The system will print 5 reports quickly" or "The system will print 5 reports with a high degree of reliability".

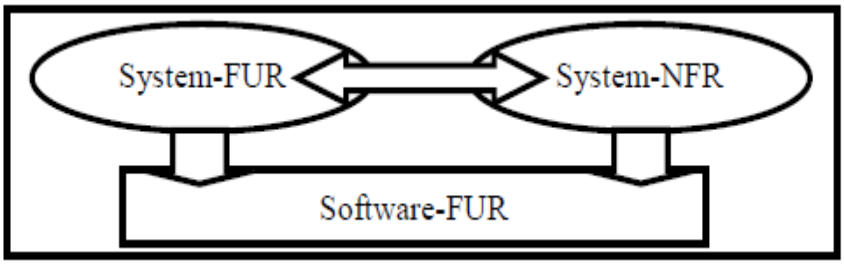

Fig. 1. Mapping system-FUR and -NFR into software-FUR

In the ECSS (European Cooperation on Space Standardization) standards for the aerospace industry [16-19] and IEEE 830 [20] standards, a number of concepts are provided to describe various types of candidate operations requirements at the system, software, and hardware levels. However, these standards vary in their views, terminology, and coverage of operations.

Currently, there exists no generic model for the identification and specification of software FUR for implementing system operations requirements (system NFR) based on the various views documented in international standards and in the literature. Consequently, it is challenging to measure these operations-related software FUR, and take them into account quantitatively for estimation purposes.

This paper focuses on a single type of NFR, that is, system operations requirements, and reports on the work carried out to define an integrated view of software FUR for system operations NFR, on the basis of international standards

including the use of the generic COSMIC – ISO 19761 [21] model of software FUR.

The paper is organized as follows. Section 2 presents the view of software FUR in ISO 19761. Section 3 identifies the standards describing operations requirements. Section 4 presents a standards-based definition of a generic model of requirements for software to implement system operations NFR. Section 5 presents the sizing of a reference instantiation of the generic model of operations software FUR. Finally, a discussion is presented in section 6.

2 A generic view of software-FUR in ISO

In the collection of ISO standards, it is specified in the ISO 14143-1 [22] that a functional size measurement method must measure the software functional user requirements (FUR). In addition, ISO 19761 – COSMIC [23] proposes a generic model of software-FUR that clarifies the boundary between hardware and software. Fig. 2 illustrates the generic flow of data from a functional perspective from hardware to software. From this generic model of software functional requirements in Fig. 2 the followings can be observed:

- Software is bounded by hardware. In the so-called "front-end" direction (i.e. left-hand side in Fig. 2), software used by a human user is bounded by I/O hardware such as a mouse, a keyboard, a printer or a display, or by engineered devices such as sensors or relays. In the so-called "back-end" direction (i.e. right-hand side of Fig. 2), software is bounded by persistent storage hardware like a hard disk and RAM and ROM memory.
- The software functionality is embedded within the functional flows of data groups. Such data flows can be characterized by four distinct types of data movements. In the "front end" direction, two types of movements (ENTRIES and EXITS) allow the exchange of data with the users across a 'boundary'. In the "back end" direction, two types of movements (READS and WRITES) allow the exchange of data with the persistent storage hardware.
- Different abstractions are typically used for different measurement purposes. In real-time software, the users are typically the engineered devices that interact directly with the software that is the users are the 'I/O hardware'. For business application software, the abstraction commonly assumes that the users are one or more humans who interact directly with the business application software across the boundary; the 'I/O hardware' is ignored.

As an FSM method, COSMIC is aimed at measuring the size of software based on identifiable FUR. Once identified, those requirements are allocated to hardware and software from the unifying perspective of a system integrating these two "components". Since COSMIC is aimed at sizing software, only those re-

quirements allocated to the software are considered in its measurement procedure.

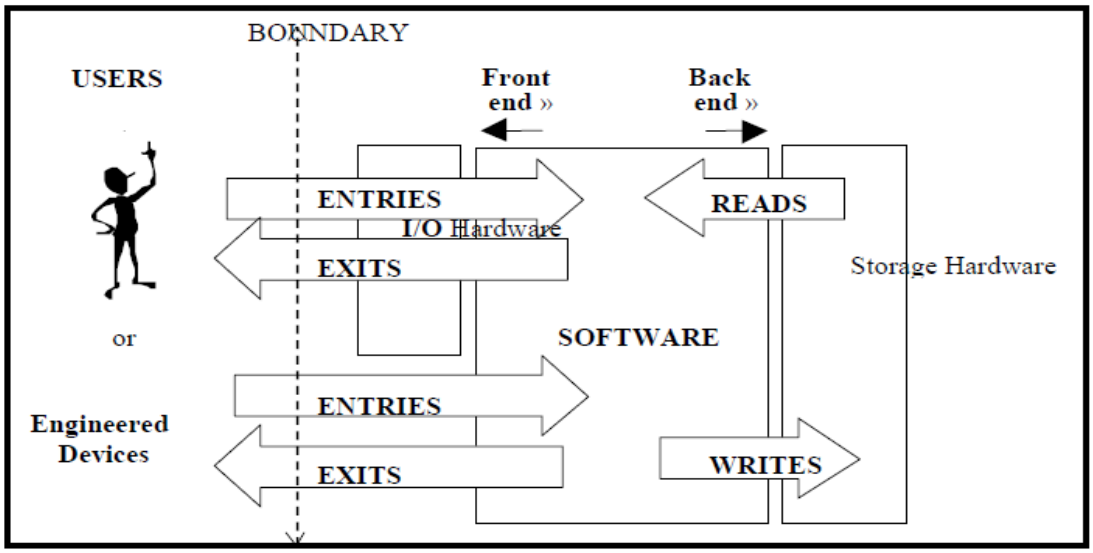

Fig. 2. Generic flow of data groups through software from a functional perspective in COSMIC – ISO 19761

3 Identification of standards for describing system operations requirements

This section presents a survey of the operations-related views, concepts, and terms in the ECSS and IEEE-830 standards. This section identifies which standards currently address aspects of the software FUR derived from system operations FUR and NFR – see Fig. 3.

The expected outcome is the identification of the various elements that should be included in the design of a standards-based framework for modelling software FUR for system operations. The elements of operations are dispersed in various system views throughout various ECSS standards and are expressed as either:

- System operations functional user requirements (operations system FUR), or
- System operations non-functional requirements (operations system NFR)

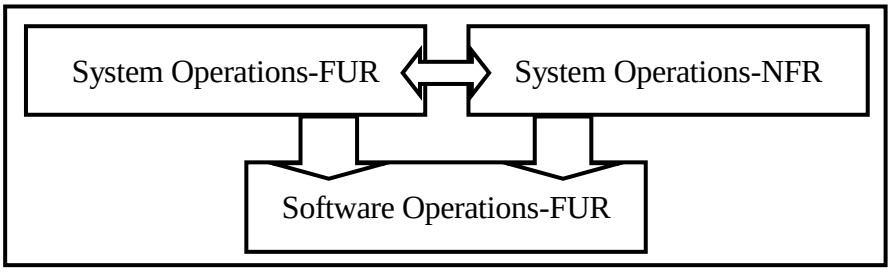

Fig. 3. Mapping system-requirements into software-FUR for operation requirements

3.1 Operations requirements in ECSS standards

Operations in the ECSS standards include any specified operations mode and mode transition for the software and, in case of man-machine interaction, the intended use scenarios and diagrams may be used to show the intended operations and related modes-transitions. Moreover, Operations engineering should be covering all operations activities through all phases of the life-cycle; i.e. operations preparation, operations validation, operation execution and disposal.

Table 1 presents a list of concepts and vocabulary used in the ECSS standards to describe system-related operations requirements. For instance, the ECSS specifies that, for system operations, analysis of operational functions and the transition mode (operational control interface and operational data interface) must be carried out. ECSS specify such requirements must be implemented in software, hardware, or a combination of the two.

Key views	Concepts and vocabulary
Operational mode and transition mode	• Operational Functions • Operational Control interface • Operational data interface • Operational mode • Transition mode • Operational scenario

Table 1. Operation requirements view and vocabulary in ECSS

While conducting the survey of all the operations concepts and terms described in the ECSS-E-40 and ECSS-Q series and in ECSS-ESA as the integrated standard for ECSS-E and ECSS-Q, it was observed that:

- These various operations elements are described differently, and at different levels of detail;
- The operational elements are dispersed throughout the various documents: there is, therefore, no integrated view of all types of candidate operations requirements;
- There is no obvious link between the operations requirements in ECSS-ESA [24] as the integrated standard and all the other ECSS standards that describe operations requirements.
- Functional description: a brief description of what the operation achieves and a list of cautions and warnings that apply to the operation.

It is also to be noted that the ECSS does not propose a way to measure such software operations requirements, and, without measurement, it is challenging

to take such an NFR either as a quantitative input to an estimation process or in productivity benchmarking.

3.2 Operations requirements in the IEEE-830

IEEE-830 [20] lists operations as one of the NFR type in their list and considers the various modes of operations as part of the user interfaces, but does not define it, nor does it provide guidance on how to describe and specify the operations requirements; of course, it does not provide guidance on how to measure any of these NFR either.

4 A standard based generic model of software-FUR for system operations requirements

This section identifies and assembles the concepts and vocabularies of operations dispersed throughout the ECSS and IEEE standards. These concepts are mapped next into a proposed model of operations software FUR – see Fig. 3, using the generic FUR model proposed in COSMIC. This COSMIC-based generic model then becomes a framework for describing the software FUR from system operations requirements based on the ECSS standards.

The initial generic model of software-FUR for system operations is not to develop a model for all the elements that could compose the security requirements, but only the elements that may be allocated to software and only those elements that are currently described in the standards selected in the first phase of the research.

Moreover, this research work is not attempting to measure quality attributes, and not attempting to develop measurement methods to define quality targets not to evaluate how such quality targets have been attained. The objective of this research work is to define the non functional requirements that translate into functional requirements allocated to software.

4.1 Mapping System Operations Views and Vocabulary from Standards

Table 2 presents the system operations requirements that are present as system requirements in the ECSS and IEEE standards in which these could be interpreted and specified, at times, as software FUR.

System operations requirements
1. Operational functions 2. Operational data interface 3. Operational control interface

Table 2. System operations–FUR

Types of operations requirements:
Next, two types of system-related operations requirements can be derived:

- System operations mode
- System transition mode

4.2 Software Operation Functions to be specified

The operations functions to be specified (and corresponding entities to be measured) are divided into external and internal operation functions that may be allocated to software – see Table 3. External operations refer to the expected control and data operations via the interface that could occur in the system, while internal operations refer to the expected operations for the executed functions occurring in the system.

Operational types	Functions type
Internal operation function	• Operational functions
External operation function	• Operational data interface function • Operational control interface function

Table 3. Software operation functions

4.3 Identification of the function types in software-FUR for system operations requirements

This section identifies the function types and functional relationships in the software-FUR for system operations requirements.

Operation Function Type 1: System Operational Control

– The control operational interface function sends at least one data group to the operational functions.
– The operational functions receive a data group from control operational interface

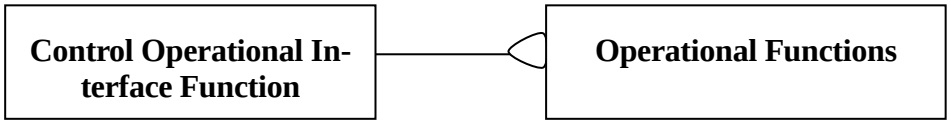

Fig. 4. System Operational Control

Operation Function Type 2: System Operational Modes

– Operational functions 1 to n send and receive a data group to/from each other.

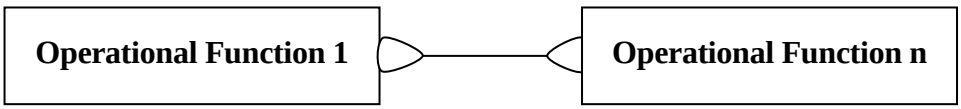

Fig. 5. System Operational Modes

Operation Function Type 3: System Operational Data

– The operational functions send and receive at least one data group to/from the operation data interface function.

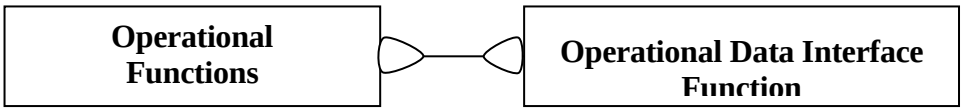

Fig. 6: System Operational Data

4.4 Identification of the functional relationships in the software FUR for system operations NFR

Figure 8 presents an overview of the relationships between the function types in the system operation FUR, using COSMIC for graphical representation. Specifically:

- The sub model of the operation function type 1 can be used to specify and measure the functional size of the external operation function for the system operational control from the received/send data groups from/to the control operational interface function and operational functions – see Figure 8.
- The sub model of the operation function type 2 can be used to specify and measure the functional size of the internal operation function for the system operational modes from the received/send data groups between operational functions, using their intermediary services– see Figure 8.
- The sub model of the operation functions type 3 can be used to specify and measure the functional size of the external operation function for the system operational data from the received/send data groups from/to the operational data interface function and operational functions – see Figure 8.

As mentioned in this paper the intermediary service used when a functional process of an application service in application A requires data that is available via an application service in application B, the former application service calls a functional process of the intermediary service, in this paper this symbol (⊗) used to represent the intermediary service – see Fig 7 and Fig. 8.

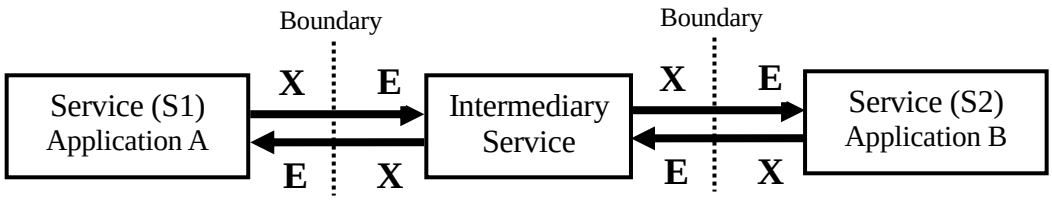

Fig. 7: Application services and interconnecting intermediary service

This model is referred to here as a generic model of software FUR for system operations.

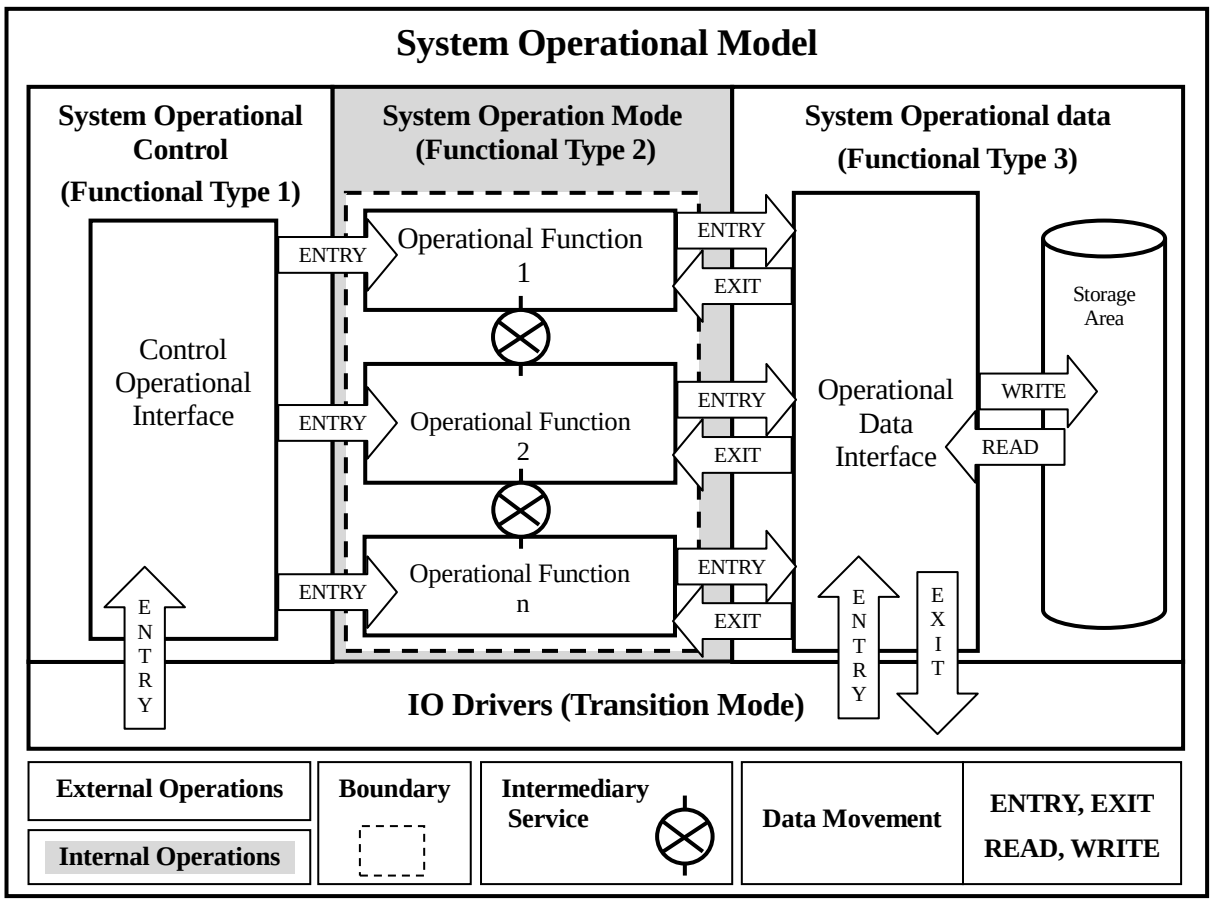

Fig. 8**.** COSMIC reference model of operations requirements allocated to software

5 Sizing a Reference Instantiation of the Generic Model of Software-FUR for System Operations

The specification of software FUR for system operations in any specific project is a specific instantiation of the proposed generic model described in Figure 8. When the software specification document is at the level of the movements of data groups, then these functional requirements can be directly measured using the COSMIC measurement rules.

Table 4 presents the measurement results using a specific instantiation of operations requirements which would have one of each of the operational function types and relationships described in section 4 and Figure 8. For example, for system operational control (Operation Function Type 1):

- The control operational interface function sends one data group to operation functions.

- The operational functions receive one data group from the control operational interface

The above requirements correspond to 2 COSMIC Exit and Entry data movements, for a functional size of 2 COSMIC Function Points (i.e. 2 CFP).

The corresponding total functional size of this specific instantiation of Figure 8 would therefore correspond to 19 data movements (of one data group each), which would then give a functional size of 19 CFP for this specific instantiation, using the COSMIC ISO 19761 measurement standard - see the bottom line of Table 4.

Operations Function Types	**Data Movement Description**	**Data Movement Type**
System Operational Control	The control operational interface function sends a data group to the operation functions.	X For each process
	Operational functions receive one data group from the control operational interface	E For each process
	The control operational interface function sends and receives data group to/from the I/O device.	E & X For each process
System Operational modes	Operational functions 1 to n send and receive data groups to/from each other by using intermediary service.	4E & 4X For each process
	Operational functions 1 to n send and receive data groups from control operational interface function	R & W For each process
System Operational Data	Operational data interface function send and receives data groups to/from the operational functions	E & X For each process
	Operational data interface function read and writes data groups to/from the operational functions	R & W For each process
	Operational data interface function receives data groups from the I/O device.	E For each process
Total Cosmic Functional Size (with one data group per process)		**19 CFP**

Table 4. Functional Size of a specific instantiation of the generic model of software-FUR for system operations

To use a generic model of system operations, for example, assume an I/O device send an operational command to execute, this command for instance needs for 2 operational functions in the system to complete this command, Find the functional size measurement for the operation.

Table 5, illustrates,the measurement solution of the instantiation case of the above example based on on figure 8 and table 4:

1. The I/O device sends one data movements to the control operational interface function.	X	1 CFP	2. The control operational interface function receive one data movements from I/O device	E	1 CFP
3. The control operational interface function sends 2 data movements, the first one operational function 1 and the sencond one to operational function 2	2X	2 CFP	4. The operational function 1 and operational functional 2, each one send a data movement to operational data interface.	2 E	2 CFP
5. The operational data interface read and write data movement to/from operational function 1	R & W	2 CFP	6. The operational data interface read and write data movement to/from operational function 2	R & W	2 CFP
7. The operational function 1 and 2 contact each other using intermediary service	4 X 4 E	8 CFP	8. The operational data interface send the results to the I/O device	X	1 CFP

Table 5. Measurement results using the COSMIC generic model of system operations

The measurement results for the instantiation case for the above example is 19 CFP for the all kinds of COSMIC data movements; This measurement result is composed of 8 Exits, 7 Entries, 2 Read and 2 Writes.

6 Discussion

This paper has introduced a procedure for specifying and measuring software requirements for the internal and external operations needed to address the system's operations requirements.
The main contribution of this paper is our proposed Generic Model of software FUR for system operations. This generic model can be considered as a kind of reference model for the identification of system operations requirements and their allocation to software functions implementing such requirements. System requirements allocated to hardware have not been addressed in this paper. Since the structure of the generic model is based on the generic model of software adopted by the COSMIC measurement standard, the necessary information for Measuring their functional size is readily available, and an example has been presented of a specific instantiation of this reference model.
Specifically, the generic model of operations presented in this paper is based on:

- the ECSS standards for the description of the NFR for system operations;
- The COSMIC measurement model of functional user requirements.

The model is independent of the software type and the languages in which the software FUR will be implemented. The proposed generic operations model (i.e. reference model) provides:

- A specification model for each type, or all types, of operations requirements: for example, the requirements to be allocated to software for the system operational control, modes and data.
- A specification measurement model for each type, or all types, of operations requirements.

The target of this paper is not the quality requirements as stated in the upcoming series, but the non functional requirements as stated in the ECSS series.

The developing group of the ISO 25000 series has not yet synchronized their work in progress with other existing industry standards and that such synchronization will be required in the future and should have been addressed by those developing these standards.

Future work includes documentation of the traceability of the elements of this generic model to the detailed elements of the ECSS standards, as well verification of this generic model to ensure full coverage of operations requirements. Discussions with groups of experts will be necessary to ensure its usefulness across various communities and to develop a consensus on further refinements of such a generic model which could be proposed eventually as a candidate for standardization.

References

1. Chung, L. and J. P. Leite, "On Non-Functional Requirements in Software Engineering" , in "Conceptual Modeling: Foundation and Applications, Essays in Honor of John Mylopoulos", 2009, Springer, Verlag Berlin Heidelberg, p. 363-379.
2. Chung, L., B.Nixon, E.Yu, J. Mylopoulos, " Non-Functional Requirements in Software Engineering", Springer, Heidelberg, 1999.
3. Mylopoulos, J., L.Chung, B.Nixon, "Representing and Using Nonfunctional Requirements: A Process- Oriented Approach", IEEE Transactions on Software Engineering, 1992. 18(6): p. 483-497.
4. Shaw, M., "Larger Scale Systems Require Higher-Level Abstractions", Software Specification and Design, IEEE Computer Society, 1989. 14 (3), p. 143-146.
5. Davis, A.M., " Software requirements: objects, functions, and states", 1993, Prentice-Hall, Inc. 521.
6. Jacobson, I., G., Booth, J.,Rumbaugl, "Excerpt from the Unified Software Development Process: The Unified Process", IEEE Software, 1999. 16(3), p. 96-102
7. Wiegers, K., "Software Requirements", 2nd edition, Microsoft Press, 2003.

8. Roman, G., "A Taxonomy of Current Issues in Requirements Engineering", IEEE Computer, 1985, p. 14-21.
9. Boehm, B.W., "Characteristics of software quality", Amsterdam, New York, North-Holland Pub. Co. , American Elsevier, 1978.
10. Antón, A.I., "Goal identification and refinement in the specification of software-based information systems", PhD Thesis, Georgia Institute of Technology, 1997.
11. Al-Sarayreh, K.T. 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, SERA10, 2010. Montreal, Canada.
12. Al-Sarayreh, K.T. and A. Abran, "Measurement of Software Requirements Derived from System Reliability Requirements", 24th European Conference on Object-Oriented Programming (ECOOP 2010), Maribor, Slovenia, EU, 2010.
13. Al-Sarayreh, K.T., A. Abran, and J.J. Cuadrado-Gallego, "A 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, 2010.
14. Abran, A., K.T. Al-Sarayreh, 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, 2010.
15. Abran, A. and K.T. Al-Sarayreh, "Standards-Based Model for the Specification of System Design and Implementation Constraints", 17th International Conference on European Systems and Software Process Improvements (EURO-SPI 2010), Industry track, Grenoble Institute of Technology, Grenoble, France, Sept. 2010.
16. ECSS-E-40-Part-1B, "Space Engineering: Software - Part 1 Principles and Requirements", in European Cooperation for Space Standardaization,The Netherlands, 2003.
17. ECSS-E-40-Part-2B, "Space Engineeing:Software- part 2 Document Requirements Definitions", European Cooperation for Space Standardaization, The Netherlands, 2005.
18. ECSS-Q-80B, "Space product assurance: Software product assurance", European Cooperation for Space Standardaization, The Netherlands, 2003.
19. ECSS-E-ST-10C, "Space engineering: System engineering general requirements", Requirements & Standards Division Noordwijk, The Netherlands, 2009.

20. IEEE-Std-830, " IEEE Recommended Practice for Software Requirements Specifications", 1993.
21. ISO/IEC-19761, "Software Engineering - COSMIC v 3.0 - A Functional Size Measurement Method", International Organization for Standardization, Geneva (Switzerland), 2003.
22. ISO/IEC-14143-1, "Information technology - Software measurement - Functional size measurement Part 1: Definition of concepts", International Organization for Standardization, Geneva (Switzerland), 1998
23. ISO-19759, "Software Engineering Body of Knowledge (SWEBOK)", IEEE Computer Society, 2004.
24. ECSS-ESA, "Tailoring of ECSS, Software Engineering Standards for Ground Segments, Part C: Document Templates", ESA Board of Standardization and Control (BSSC), 2005.