Best practices of interoperability among heterogeneous software systems: a Semat-based representation

Buenas prácticas para la interoperabilidad de sistemas de software heterogéneos: Una representación usando Semat

Boas práticas para a interoperabilidade de sistemas de software heterogêneos: Uma representação usando Semat

Diana María Torres-Ricaurte[*]
Carlos Mario Zapata-Jaramillo[**]

Abstract

Interoperability among heterogeneous software systems is a software quality sub-characteristic. Some methods for dealing with interoperability exhibit differences in aspects like generality, development method, and work products, among others. However, some authors understand interoperability as a non-functional requirement with general-purpose practices for identifying and specifying such requirement. Other authors assess and achieve interoperability by using work products falling beyond defined practices. Consequently, in this paper we propose four best practices in order to accomplish interoperability among heterogeneous software systems. Our best practices are represented with the Semat (Software Engineering Method and Theory) kernel, since it includes a language with simple and precise elements. Definition of interoperability best practices enables unification of the effort focused on software systems interoperability.

Keywords: best practices; heterogeneous software systems; interoperability; Semat.

Resumen

La interoperabilidad entre sistemas de software heterogéneos es una subcaracterística de la calidad del software; existen varios métodos que la abordan, los cuales se diferencia en aspectos como la generalidad, la metodología de desarrollo y los productos de trabajo, entre otros; sin embargo, en algunos de estos métodos la interoperabilidad se maneja como un requisito no funcional que se identifica y se especifica mediante prácticas generales. En otras propuestas la interoperabilidad se evalúa y se alcanza mediante la elaboración de productos de trabajo que no se enmarcan en prácticas definidas. En este artículo se proponen cuatro buenas prácticas para abordar la interoperabilidad de sistemas de software heterogéneos, las cuales se representan mediante los elementos del núcleo de *Semat* (Teoría y Método de la Ingeniería de Software), que proporciona un lenguaje con elementos claros y sencillos. La definición de buenas prácticas de interoperabilidad permite unificar los esfuerzos enfocados en el logro de la interoperabilidad de los sistemas de software.

Palabras clave: buenas prácticas; interoperabilidad; sistemas de software heterogéneos; *Semat*.

*   M.Sc. Universidad Nacional de Colombia (Medellín-Antioquía, Colombia). dimtorresri@unal.edu.co.
**   Ph.D. Universidad Nacional de Colombia (Medellín-Antioquía, Colombia). cmzpata@unal.edu.co.

Resumo

A interoperabilidade entre sistemas de software heterogêneos é una subcaracterística da qualidade do software; existem vários métodos que a abordam, os quais se diferenciam em aspectos como a generalidade, a metodologia de desenvolvimento e os produtos de trabalho, entre outros; porém, em alguns destes métodos a interoperabilidade administra-se como um requisito não funcional que se identifica e se especifica mediante práticas gerais. Em outras propostas a interoperabilidade avalia-se e alcança-se mediante a elaboração de produtos de trabalho que não se enquadram em práticas definidas. Neste artigo propõem-se quatro boas práticas para abordar a interoperabilidade de sistemas de software heterogêneos, as quais são representadas mediante os elementos do núcleo do Semat (Teoria e Método da Engenharia de Software), que proporciona uma linguagem com elementos claros e simples. A definição de boas práticas de interoperabilidade permite unificar os esforços focalizados na conquista da interoperabilidade dos sistemas de software.

Palavras chave: Interoperabilidade, Sistemas de software heterogêneos, Boas práticas, Semat.

158

Revista Facultad de Ingeniería (Rev. Fac. Ing.) Vol. 26 (44), pp. 157-166. Enero-Abril, 2017. Tunja-Boyacá, Colombia.

I. Introduction

In the ISO/IEC 25000 standards [1], interoperability is a software sub-characteristic defined as the degree at which a software product can cooperatively operate with others. Interoperability relates to the system interconnection needs in heterogeneous environments. A set of activities is proposed in different methods for achieving interoperability in the development process [2-7].

The interoperability methods we assess here show differences is aspects like development methodology (agile vs. plan-based), generality (non-functional vs. interoperability requirements), or work product definition. Neither the NFR-Framework [3], the NFRE [4], the method of functional interoperability analysis (traditional methodologies) [6], the NORMAP (agile methodologies) [5], nor other models [7] are used to define specific practices with enough detail on the activities and work products, which are necessary for supporting the interoperability processes.

Additionally, an interoperability measure for the software system is obtained and assessed in LISI [2], and work products are obtained in order to support decisions regarding interoperability of the target system; however, the practices are not explicitly defined. For this reason, activities are not grouped in repeatable processes for allowing good results for interoperability among software systems.

In this paper, we propose four best practices for interoperability among heterogeneous software systems with the level of detail necessary for work products in interoperability. Such practices result from the synthesis of common practices identified in existing methods. For each practice, we describe the essential elements of the Semat (Software Engineering Method and Theory) kernel, since it is based on a simple language and a graphic representation [8].

In this paper, we define the best interoperability practices as a way to consolidate a set of activities with proven results proposed in the methods. Processes repeated across different methods —which commonly provide good results for interoperability management— are grouped under best practices by using the consolidated information. Likewise, the representation in the Semat kernel enables the incorporation of the best practices defined in any method expressed by using the same notation.

This paper is structured as follows: in Section 1 we define interoperability and the Semat kernel; in Section 2 we describe the methods assessed as background for our proposal; in Section 3 we propose the methods representation, and their practices by using on the Semat kernel; in Section 4 we describe the proposed best practices; and finally, in Section 5 we discuss conclusions and future work.

II. Theoretical framework

A. Interoperability among heterogeneous software systems

In the quality model of the ISO/IEC 25000 standards [1], interoperability is a compatibility sub-characteristic associated with information exchange in shared environments. These environments have heterogeneous characteristics from diverse sources. Several methods of interoperability among heterogeneous systems are proposed [2-7]. NFR-Framework [3] is used to design and implement functional requirements of the software system; and LISI [2] is used to evaluate and achieve interoperability.

B. Semat (Software Engineering Method and Theory)

Semat [8] is an initiative intended to establish software engineering as a rigorous discipline. The first result is a kernel with elements that are essential and universal to all development endeavor, and are expressed in a simple language. *Essence* [9] is the language for expressing the Semat kernel elements in a simple and natural language, and in a graphic way. Table 1 shows the *Essence* graphic elements with their names, symbols, and descriptions.

Revista Facultad de Ingeniería (Rev. Fac. Ing.) Vol. 26 (44), pp. 157-166. Enero-Abril, 2017. Tunja-Boyacá, Colombia.

159

TABLE 1
GRAPHIC ELEMENTS OF THE SEMAT KERNEL [9]

Name	Symbol	Description
Alpha		Essential element for determining the progress and health of any software engineering endeavor.
Activity Space		Activities always carried out in any software engineering endeavor: customer, solution, and endeavor.
Activity		Defines one or more types of work products and tasks, and guidelines on how use them in a practice context.
Practice		Description on how to handle a specific aspect of a development software endeavor.
Activity Association		Connects a work product with an alpha or a role. Connects an activity space with an activity or a phase.
Role / Phase		Set of responsibilities. Development process stage.
Work product		Device of value and relevance for a software endeavor.

III. INTEROPERABILITY METHODS

In this paper, we evaluate methods of elicitation, refinement, specification, and design of non-functional requirements because they are established proposals in software engineering, and because interoperability is considered as a non-functional requirement. Furthermore, the maturity assessment method LISI is included because it is robust, and its representation is thought to encompass other similar methods. Table 2 summarizes the reviewed methods with the evaluation criteria.

A. NFR-Framework

The NFR-Framework [3] is a method to elicit, refine, and specify non-functional requirements (NFR), and is based on producing and refining a work product known as SIG (Softgoal Interdependency Graph). In this method NFRs are decomposed in small objectives called *softgoals*, which are related to functional requirements throughout the software development life cycle.

B. NFRE (*Non-functional Requirements Engineering*)

NFRE [4] is a method based on NFR-Framework for performing similar tasks regarding the NFRs. In NFRE, NFRs identification is based on a method oriented toward the TORE (Task-oriented Requirements Engineering) tasks. Therefore, NFRE is considered as a process based on IT (Information Technology) for supporting different tasks concerning the NFRs.

C. NORMAP (*Non-functional Requirements Modeling for Agile Process*)

NORMAP [5] is proposed as the version for the agile process of the NFR-Framework. This method encompasses seven stages to develop NFRs: 1) selected NFRs and initial data collection, 2) initial data pre-processing, 3) automatic parsing of requirement statements, 4) modeling agile loose cases, 5) modeling agile use cases, 6) modeling agile choose cases, and 7) requirements implementation sequence planning.

D. Requirements model of quality attributes

This process model is proposed for identifying and specifying cross-cutting quality attributes to the

development process [7]. Attributes are integrated in the functional requirements description, and identified at early stages of the development process. The resultant work product is a list of quality attributes that identifies the decomposition of the attributes into simpler subattributes, the dependencies, the priority, and the UML (Unified Modeling Language) model where the attribute is included.

E. Non-functional interoperability analysis method

This method introduces the non-functional required interphase (NRI), and the non-functional provided interface (NPI) concepts [6]. NRI specifies the NFRs the customer component expects to be accomplished. NPI specifies the NFRs the supplier component must

or expect to support. Once the NRI and NPI are compared, the degree of interoperability is assessed, and the interoperability is solved by using different strategies.

F. Interoperability maturity models

LISI (Levels of Information Systems Interoperability) [2] is a method for defining levels of interoperability: isolated, connected, functional, domain, and enterprise. The levels are measured according to the system capacity in four attributes: procedure, applications, infrastructure, and data. Furthermore, different work products for supporting the decision-making process regarding the desired interoperability are proposed.

TABLE 2
EVALUATION OF METHODS USED TO STUDY SOFTWARE INTEROPERABILITY

Evaluated method	Application stage	Evaluation criteria								
		Non-functional requirements	Software interoperability	Work product	Defined practices	FR relationships	Work artifacts	Component application	Identification of roles	Interoperability elements
NFR-Framework [3]	Software life cycle	√		√		√				
NFRE [4]	Software life cycle	√							√	
NORMAP [5]	Specification-Design	√		√		√				
Requirements model of quality attributes [7]	Software life cycle			√		√	√			
Non-functional interoperability analysis method [6]	Design	√					√	√		
LISI [2]	Software life cycle		√	√						√

We used the following criteria to evaluate the methods: (i) application stage: life cycle stage in which the method is applied, and its reach; (ii) non-functional requirements: NFRs specification, analysis, design, or implementation; (iii) software interoperability: focused on the software system interoperability; (iv) work products: proposed work products for modeling, interdependence, design, etc. of NFRs attributes; (v) defined practices: identification and collection of activities that can form the NFRs practices; (vi)

FR relationships: generation of interdependencies between functional requirements and NFRs; (vii) work artifacts: modification or setting up of UML artifacts in NFRs modeling; (viii) component applications: focused on software components; (ix) identification of roles: identification of roles associated with interoperability or NFRs; and (x) interoperability elements: identification of interoperability elements at different stages of the development process.

Revista Facultad de Ingeniería (Rev. Fac. Ing.) Vol. 26 (44), pp. 157-166. Enero-Abril, 2017. Tunja-Boyacá, Colombia.

161

IV. REPRESENTATION OF INTEROPERABILITY METHODS BY USING SEMAT

In the assessed methods, the best practices are not explicitly described. Therefore, we should characterize each method by means of the following steps: procedure reading, definition of the set of activities proposed in the method, identification of the Semat elements associated with each activity, activity grouping according to their objectives, and finally, characterization and labeling of the repeatable activities with proven good results. Practices identified for each method are represented by using *Essence* [9].

Representations in the Semat kernel of the NFR-Framework, the NFRE, and the requirements model of quality attributes are shown in Figure 1. NORMAP, LISI, and the non-functional interoperability analysis method are represented in Figure 2. Additionally, in Figure 3, we propose the representation of the LISI requirements management best practice in terms of activity spaces and activities, and the RUP (Rational Unified Process) phase in which activities are executed.
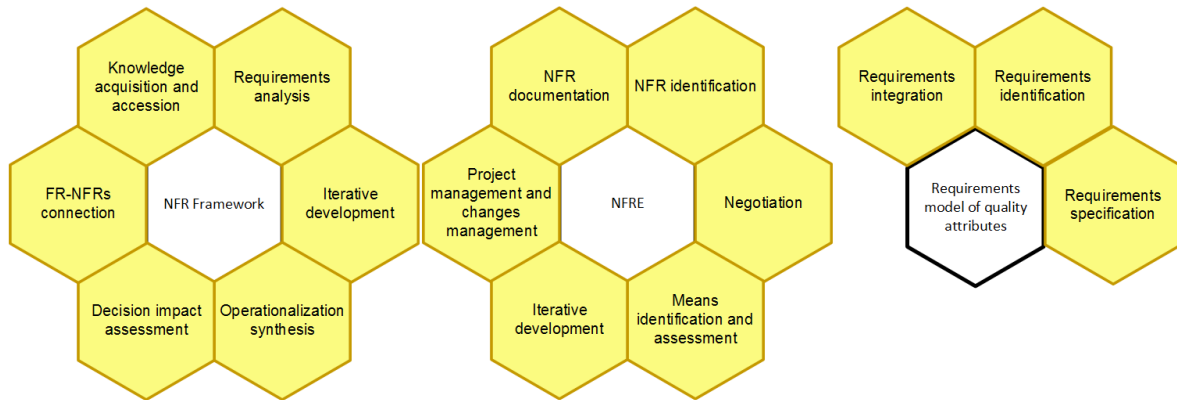

FIG. 1. Practices of the NFR-Framework, NFRE, and Requirements model for quality attributes.
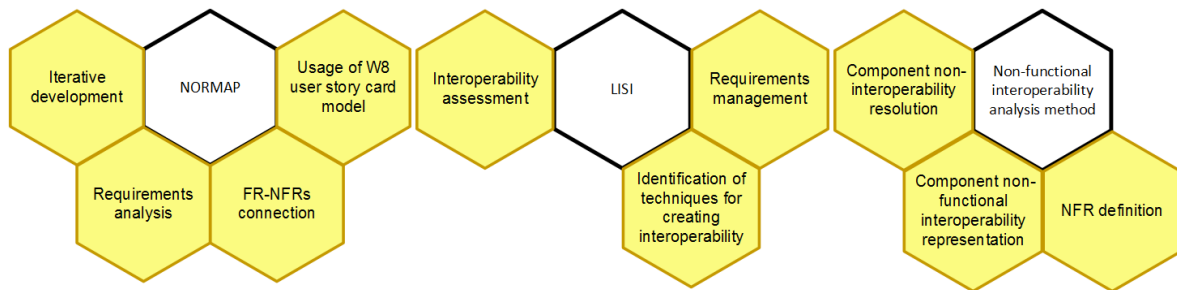

FIG. 2. Practices of the NORMAP, LISI, and Non-functional interoperability analysis method.

V. INTEROPERABILITY BEST PRACTICES

Once we identified and characterized the best practices of the evaluated methods, we proposed four interoperability best practices (Figure 4): "Interoperability Requirements Management," "Interoperability Technique Identification," "Interoperability Assessment," and "Iterative Development." For each practice, we identified activity spaces, activities, roles, and work products.
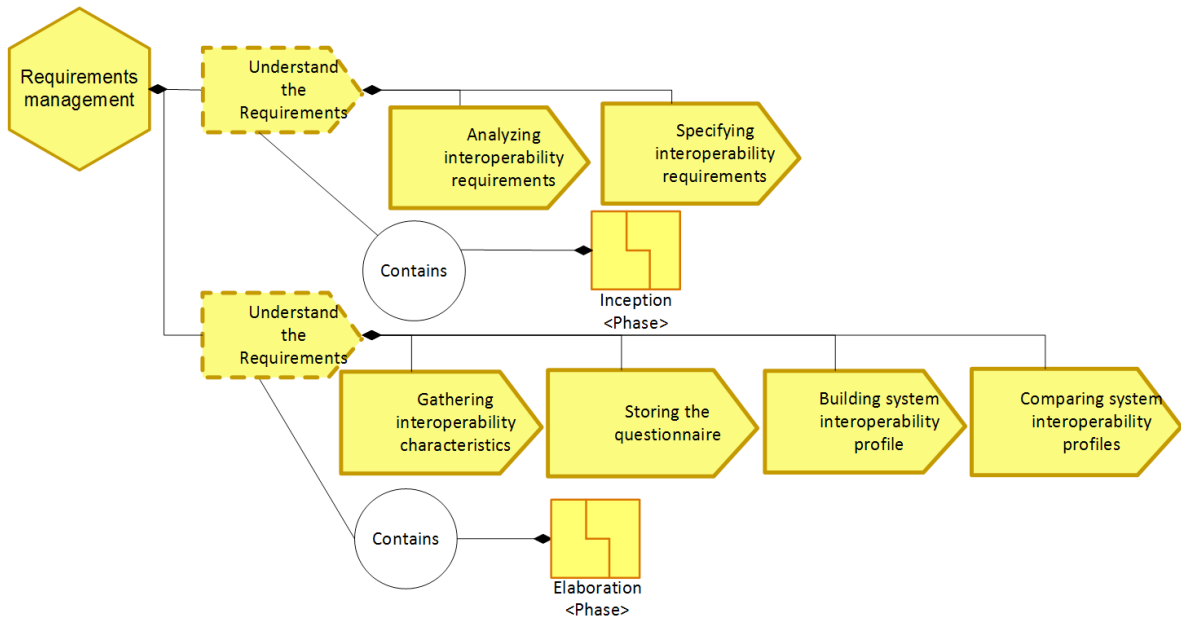
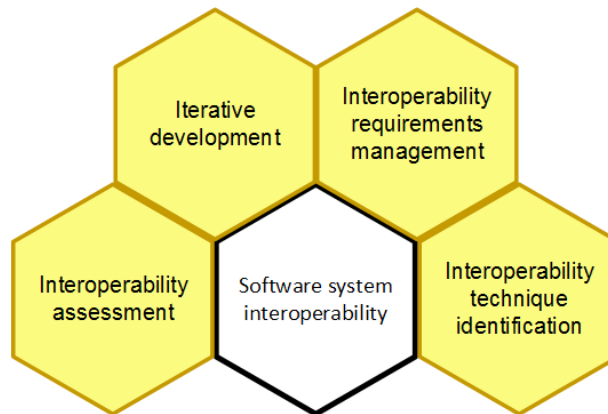**Fɪɢ. 3.** Requirements management best practice of the LISI method.

Fɪɢ. 4. Practices of interoperability among software system.

Such practices are proposed by recognizing common activities in the assessed methods, some of them named in different ways. In addition, activities used to solve the system interoperability are unified. The proposed roles are the most common to interoperability practices and they contain competencies implicit in the methods. The work products are a compilation of those we found in the methods, and are selected because they provide information regarding the current system interoperability state, and information for supporting decision making for accomplishing the target interoperability.

In Figure 5a, we represent the "Iterative Development" best practice, which groups the necessary activities for revising and executing modifications to the functional and non-functional requirements without generating inconsistencies, and assuring the design can be adapted. In Figure 5b, we represent the "Interoperability Assessment" best practice, which characterizes the system interoperability in terms of the necessary features for interacting with other systems. This characterization is used to identify the current level of interoperability, and to propose adjustments to achieve the expected level. In Figure 5c, we represent the "Interoperability Technique Identification" best practice, which has to do with the technical decisions made by developers regarding the software system design and implementation.
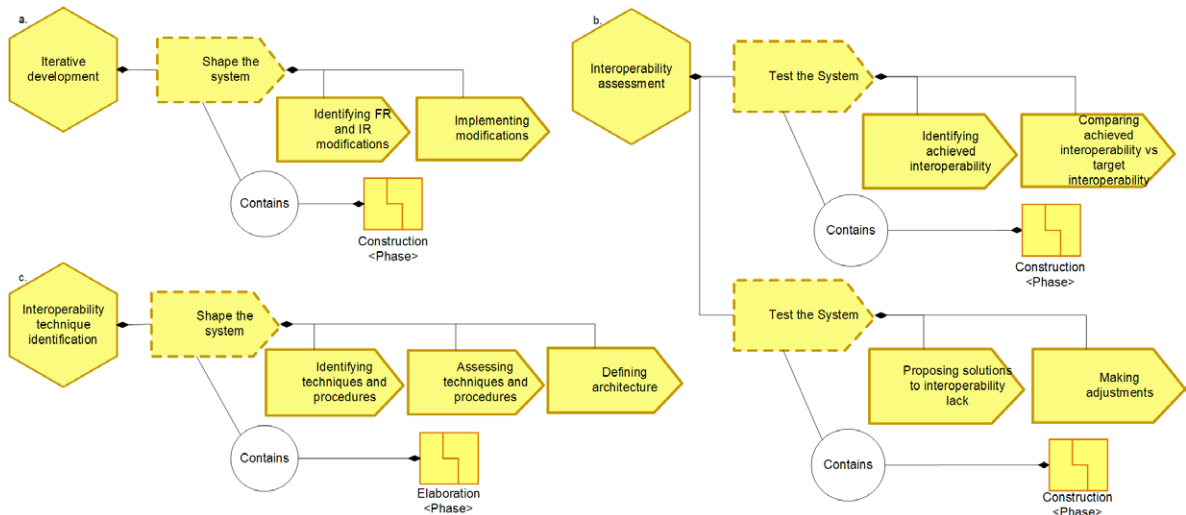
Revista Facultad de Ingeniería (Rev. Fac. Ing.) Vol. 26 (44), pp. 157-166. Enero-Abril, 2017. Tunja-Boyacá, Colombia.

163

Fig. 5. Semat representation of a) Iterative Development practice; b) Interoperability Assessment practice; c) Interoperability Technique Identification practice.

In Figure 6, we represent the "Interoperability Requirements Management" practice, the alphas associated with work products and the roles related to interoperability.
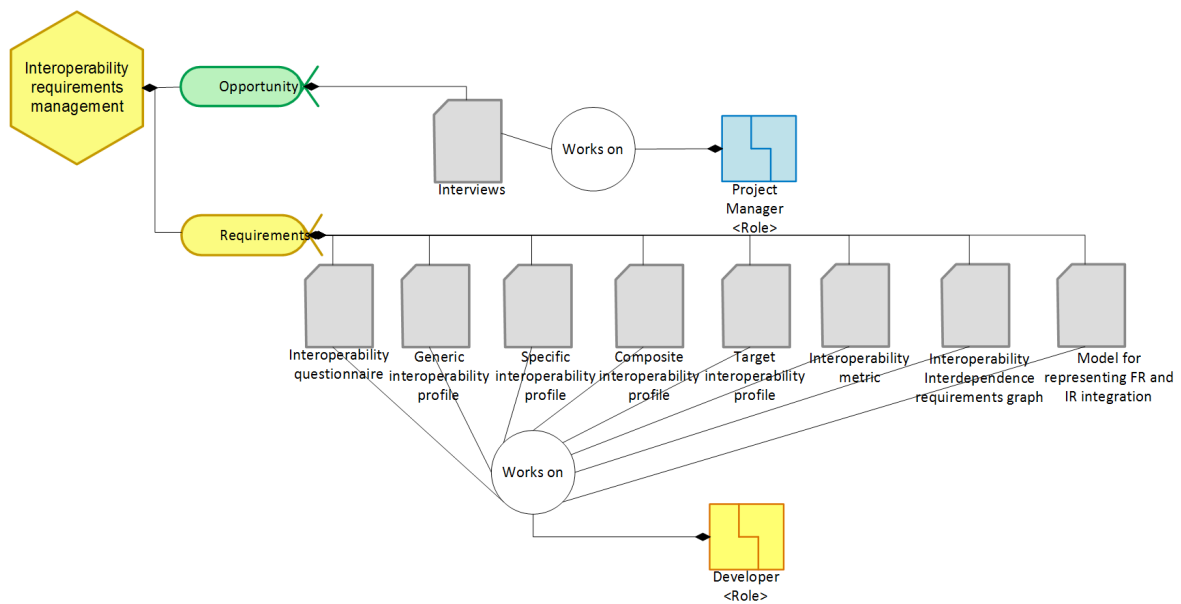

Fig. 6. Semat representation of the "Interoperability Requirements Management" practice.

Finally, in Figure 7, we represent the "Interoperability Requirements Management" best practice. In this representation, we associate activity spaces with phases and activities contributing to achieve the interoperability objectives.
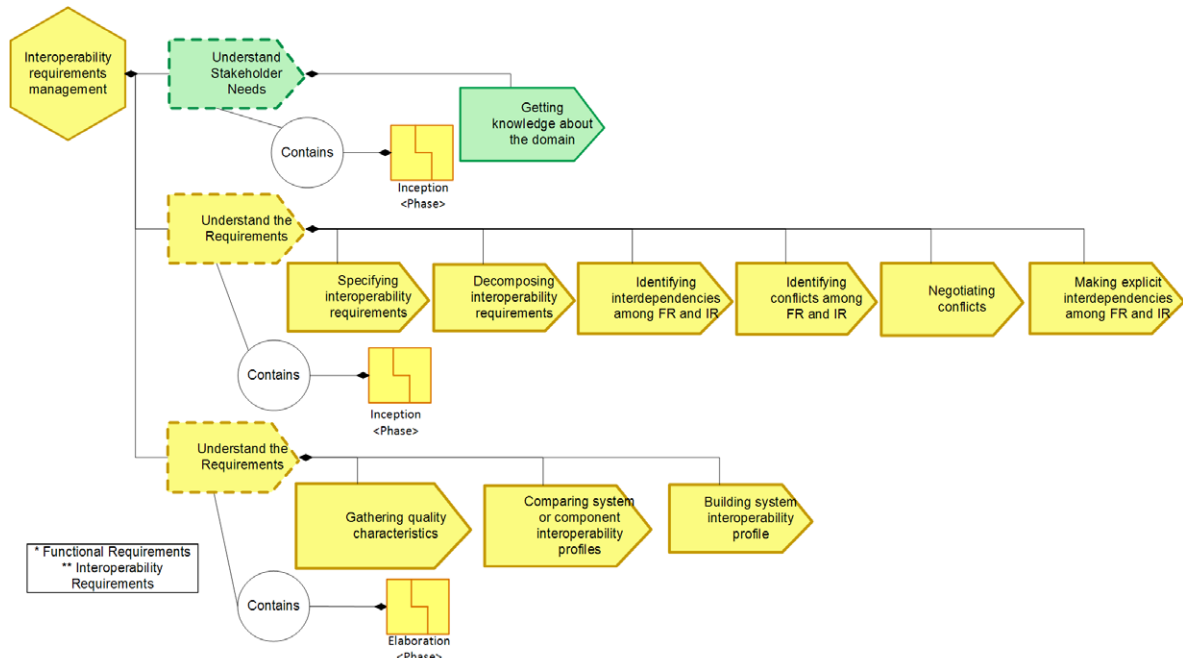
Fig. 7. Semat representation of the "Interoperability Requirements Management" practice.

VI. Conclusions and future work

Three main conclusions arise from the representation made of the methods for managing non-functional requirements:

1) Software systems interoperability is treated as a non-functional requirement in four out of the six evaluated methods; therefore, information about inoperability features and elements was not compiled.

2) Iterative development, connection between functional and non-functional requirements, and decision assessment were repeated in some of the methods. Iterative development is a good practice used to represent the need for refining design and implementation in order to accomplish the proposed goals. With connection between functional and non-functional requirements we can demonstrate the close relationship between system functionalities and properties the stakeholder is willing to accept in the solution. Finally, with decision assessment we show the need for qualitative and quantitative measurements in order to propose actions aimed at rectifying and achieving the objectives.

3) Best practices are grouped under a particular name in order to approach in a general way the activities proposed in the methods.

We demonstrate with our Semat-based representation of the best practices that interoperability requires specific procedures, since both the activities and work products require particular information about the software system for guaranteeing the expected results. Moreover, we can suggest with the representation that interoperability is a non-functional requirement crosscutting the development process, since interoperability activities are executed during all development phases. Consequently, once interoperability is considered one of the software system objectives, the requirements specification and software system architecture are aligned with this objective. Finally, we proposed work products for gathering information about the target interoperability, the current interoperability profiles, and the accomplished interoperability.

We suggest —as future work— the assessment of the proposed work products in order to validate their utility, and the efficiency of the obtained information.

Revista Facultad de Ingeniería (Rev. Fac. Ing.) Vol. 26 (44), pp. 157-166. Enero-Abril, 2017. Tunja-Boyacá, Colombia.

165

AUTHOR CONTRIBUTIONS

Both authors contributed to all the stages of this study.

REFERENCES

[1]   Systems and software Engineering-Systems and software Quality Requirements and Evaluation, ISO/IEC 25000: 2014.

[2]   C4ISR Interoperability Working Group, *Levels of information systems interoperability (LISI)*. Washington, DC, 1998.

[3]   L. Chung, B. A. Nixon, E. Yu, and J. Mylopoulos, *Non-functional requirements in software engineering*. Boston, MA: Springer US, 2000. DOI: http://doi.org/10.1007/978-1-4615-5269-7.

[4]   B. Paech and D. Kerkow, "Non-functional requirements engineering-quality is essential," in *Proc. 10th International Workshop on Requirements Engineering Foundation for Software Quality,* Riga, Latvia, 2004, pp. 237-250.

[5]   W. M. Farid, "The NORMAP Methodology: Lightweight Engineering of Non-functional Requirements for Agile Processes," in *Proc. 2012 19th Asia-Pacific Software Engineering Conference*, Hong Kong, China, Dec. 2012, pp. 322-325. DOI: http://doi.org/10.1109/APSEC.2012.23.

[6]   S. Supakkul, E. Oladimeji, and L. Chung, "Toward component non-functional interoperability analysis: A uml-based and goal-oriented approach," in *Proc. IEEE International Conference on Information Reuse and Integration,* Waikoloa, USA, 2006, pp. 351-358.

[7]   I. Brito, A. Moreira, and J. Araújo, "A requirements model for quality attributes," in *Proc. Aspect-Oriented Requirements Engineering and Architecture Design,* Enschede, Netherlands, 2002, pp. 1-7.

[8]   C. M. Zapata (Traductor), I. Jacobson, P. P. W. Ng, P. E. McMahon, I. Spence, and S. Lidman, "La Esencia de la Ingeniería de Software: El Núcleo de Semat*," Revista Latinoamericana de Ingeniería de Software*, vol. 1(3), pp. 71-78, Jun. 2013.

[9]   I. Jacobson, P. Ng, P. McMahon, I. Spence, and S. Lidman, *Kernel and Language for Software Engineering Methods (Essence)*, Needham, MA: OMG, 2014.