Procedure to Validate Non-Functional Requirements in Information Management Systems

Niurka Martínez Durán, Ing.[1], Alexander Delgado Gutierrez, Ing.[2], Jorge Emilio Escala Maceo, Lic.[1]
[1]University of Informatics Sciences, Cuba, nduran@uci.cu, jeem@uci.cu
[2]University of Informatics Sciences, Cuba, adgutierrez@uci.cu

Abstract– ***For a proper development of Information Management Systems it is required a validation of non-functional requirements. This activity is performed by the Quality Center for Technological Solutions (CALISOFT, Spanish acronyms), not from the initial stages, but once the system is implemented. Provoking mistakes made in specification of requirements to be dragged until the end, with an impact at every stage of the software development. The present work aims at designing a procedure to validate non-functional requirements of Information Management Systems. For that, it was made an analysis of the requirements engineering, specifically for the validation stage. Three phases of the procedure design were identified, each one of them containing: activities, responsible, participants, input and output artefacts and technique to be used. It was also proposed a set of metrics to assess the quality of the non-functional requirements following the ISO/IEC 9126 standard. The proposal in applied to the Integral Solution for Project Management and Centralized Actions (SIGEPAC by Spanish acronyms) Finally, the result was a procedure to validate non-functional requirements, allowing detecting and correcting mistakes since the beginning.***

Keywords-- ***Requirement engineering, metrics, non-functional requirements, validation.***

Digital Object Identifier (DOI): http://dx.doi.org/10.18687/LACCEI2015.1.1.031
ISBN: 13 978-0-9822896-8-6
ISSN: 2414-6668

Procedure to validate nonfunctional requirements in Information Management Systems

Ing. Niurka Martínez Durán[1], Ing. Alexander Delgado Gutierrez [2], Lic. Jorge Emilio Escala Maceo[3]
[1]University of Informatics Sciences, Cuba, nduran@uci.cu, jeem@uci.cu
[2]University of Informatics Sciences, Cuba, adgutierrez@uci.cu

Abstract– For a proper development of Information Management Systems it is required a validation of nonfunctional requirements. This activity is performed by the Quality Center for Technological Solutions (CALISOFT, Spanish acronyms), not from the initial stages, but once the system is implemented. Provoking mistakes made in specification of requirements to be dragged until the end, with an impact at every stage of the software development. The present work aims at designing a procedure to validate nonfunctional requirements of Information Management Systems. For that, it was made an analysis of the requirements engineering, specifically for the validation stage. Three phases of the procedure design were identified, each one of them containing: activities, responsible, participants, input and output artifacts and technique to be used. It was also proposed a set of metrics to assess the quality of the nonfunctional requirements following the ISO/IEC 9126 standard. The proposal in applied to the Integral Solution for Project Management and Centralized Actions (SIGEPAC by Spanish acronyms) Finally, the result was a procedure to validate nonfunctional requirements, allowing detecting and correcting mistakes since the beginning.

Keywords-- Requirement engineering, metrics, nonfunctional requirements, validation.

I. INTRODUCTION

The development of the Information and Communication Technologies (ICTs) has reached an unparalleled worldwide success, provoking in return a fruitful advance in the software industry. One of the significant problems in the field of informatics is the Quality Management, due to technological innovations that have led to dramatically increase the size and complexity of informatics systems. Since its appearance, this topic has been a concern to specialists, engineers, researchers and traders of this branch, which have conducted studies around two main objectives: first to obtain a software with quality and second to assess the software quality.

In the middle of this competitive view, the Information Management Systems emerge, computer applications specially designed for the management and continuous improvement of policies, procedures and processes of the organization. One of the key aspects for the proper functioning of this variant of software development, is to identify and validate nonfunctional requirements.

"Non-functional requirements, as the name suggests, are requirements that are not directly concerned with the specific functions delivered by the system. They may related to emerging system properties such as reliability, response time and storage occupancy."[1] They arise from the needs of the user, due to budget constraints, the policies of the organization, the need for interoperability of software or hardware with other systems, or external factors such as safety regulations or laws about privacy.

In this scenario a proper validation systematically increases the expectations of end users. Its implementation is generally difficult because typical defects usually appear as contradictions in the specification, small differences between functional and nonfunctional requirements, little understandable and redundant specifications, getting in conflict and not indicating all the necessary hardware resources. The validation cost is very high and the clients paying for the system sometimes think that these are not justified.

At the University of Informatics Sciences validation of software products is carried out by the Quality Center for Technological Solutions (CALISOFT) and quality groups belonging to each development center. The strategy followed is to perform this activity at the end of the implementation of the system, by checking that the initial nonfunctional requirements are implemented correctly, as described in the document Software Requirements Specification (SRS). Many of the errors in these requirements are given by inadequate specification that can only be demonstrated when the product is already in its final stage, causing delays, rising costs and customer dissatisfaction. Validation at the end allows detecting deficiencies with the execution of system implementation, but not making it initially causes to drag errors and omissions of nonfunctional requirements, which increase their impact with the gradual development of the software.

In an Information Management System in which a proper validation of nonfunctional requirements is not carried out in its development, from initial stages, it is likely to have problems in efficiency, portability, deployment, ease of use, robustness, reuse and compatibility with other systems. It also causes in the organization using it, difficulties to constantly renew its objectives, strategies, operations and service levels. For all things mentioned above it is intended as general objective: To develop a procedure for the validation of nonfunctional requirements of Information Management Systems.

II. MATERIALS AND METHODS

According to ISO / IEC 9126 standard, the quality of the software product should be detailed hierarchically into a model composed of features and subfeatures [2]. This model relates functional and nonfunctional requirements through the six quality attributes proposed (functionality, reliability, usability, efficiency, maintainability and portability) and some sub-categorizes as shown in the following picture:

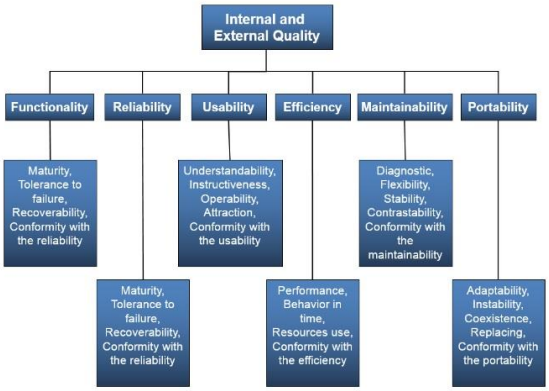

Fig. 1 Features and sub-features model of internal and external quality.

In this research were used those features and sub-features that match the classification of nonfunctional requirements (NFR) listed below:

Functionality

- Security: Capability of the software product to protect systems or data so that unauthorized individuals or systems cannot read or modify them, and authorized persons or systems have access to them.
- Interoperability: The capability of the software product to interact reciprocally with one or more specified systems.

Reliability

- Maturity: Capability of the software product to avoid total failure as a result of a software failure occurred.
- Recoverability: Capability of the software product to restore a specified level of performance and recover the data directly affected in case of total failure.

Usability

- Understandability: Capacity of the software product to enable the user to understand whether the software is suitable, and how it can be used for particular tasks and conditions of use.

Validation of Requirements

According to ISO / IEC 9126 standard, the quality of the software product should be detailed hierarchically into a model composed of features and subfeatures. This model relates functional and nonfunctional requirements through the six quality attributes proposed (functionality, reliability, usability, efficiency, maintainability and portability) and some sub-categorizes as shown in the following picture:

Validation is one of the stages of the requirements engineering (RI). Its purpose is to verify that all requirements listed in the specified document represent a description, at least acceptable, of the system to be implemented. This involves verifying that the requirements are consistent and that are complete. It also allows demonstrating that the requirements defined in the system are those the client really wants, it checks that it has not omitted any, and that they are not ambiguous, inconsistent or redundant. Its mission is to demonstrate that the requirements definition specifies the system the user wants. Discover problems in the requirements document before risking resources in the implementation. Among its activities is the assessment of the requirements, to which this investigation was focused.

Validation of a system is not only performed at the beginning, as part of the requirements engineering, but also at the end of development to determine if the initial conditions are satisfied. It is performed by means of several test cases for each specified requirement or use case. According to the IEEE 2004 Standard Verification and Validation, validation is a process that provides evidence of whether the software, associated products and processes [3]:

- Satisfy the system requirements assigned to the software at the end of each life cycle activity.
- Solve the problem correctly (for example, use the appropriate model and implement business rules correctly).
- Satisfy the use and needs of the users. (Chun, 1999)
- Software requirements validation methods.

Methods for validation can be classified into:

- Static Methods: Focused on the analysis and corroboration of the representation of the system, including documents, diagrams and code.
- Dynamic Methods: Involve running some kind of system implementation. It might seem that only static methods are enough, but this makes no sense, since static methods are more likely oriented towards verification and cannot demonstrate that the system meets user expectations, which is confirmed through validation.

For the present investigation it was intended to use both methods in order to detect and correct errors in nonfunctional requirements.

Validation Techniques

Validation techniques of requirements are made in order to examine them, to ensure that the appropriate system is defined. Permitting to detect errors early in order not to lead to unexpected results, avoid overspending and great loss of time. Among the validation techniques that were used in this research are: reviews [4], audits [5] and validation testing [6] for international use and effectiveness in terms of supporting the validation processes.

Software Quality Models.

Different literatures agree that a quality model is the set of characteristics and the relationships between them, which provide the base or rule to specify quality requirements, assessing the quality or compare any aspect of the software.

[7]. Promote the proper use of methods and tools, and enable communication among developers. The software quality models that are used in this research are mentioned at this point to serve as a reference and guide in achieving the objectives proposed.

- ISO / IEC 9126 -1991. Defines six quality characteristics desirable in any software [2].
- IEEE 830-1998: Recommended Practice for Software Requirements Specification [8].

Validation of Information Management Systems

An Information Management System is an application containing an integrated set of processes, mainly formal, that the organization knows and knows how to use (informal are not excluded) and are recorded in data through a database. Developed in a user-computer environment and operating on a set of structured data (database) using computer hardware and software, telecommunications networks, management techniques or other forms of information technology [9].

It is characterized by the availability of information when necessary and with appropriate means; by providing information selectively (quantity versus quality) variety in the form of information presentation (graphical, numerical, etc..) degree of "intelligence" of the system (preset relations); response time of the system: from a request to completion; accuracy: conformity between the data supplied and the real ones; generality: availability to meet different needs; flexibility: ability to adapt to new needs; reliability: probability of correct operation for a certain period of use; security: protection against loss and / or unauthorized use of resources; storage: repetition level of information to protect losses; and friendliness: the need for learning to its management. Examples of these include: Green SQA, Software Quality Assurance, SQA SA, Indudata Ltda, and TSOFT. These companies have high international recognition for the quality of services provided. All information related to the practiced methodologies or procedures performed, as well as results obtained can only be accessed by staff involved in the contract.

At the University of Informatics Sciences NFR validation of these systems is performed by CALISOFT and quality groups associated with each development center. It runs once the implementation is finished through a Deployment Testing Plan. It defines the types of tests to be performed, which are established according to the six quality characteristics defined in ISO / IEC 9126 standard.

After an analysis of existing alternatives for the validation of nonfunctional requirements, it was concluded that they are not feasible for the present investigation. The main disadvantage in the international arena is the inability to obtain information on how to validate the Information Management Systems. Particularly in this University validation is performed only in the final stage of software development causing failed implementations, delays in delivery time, unexpected costs and customer dissatisfaction.

Modeling tool and prospective method

Visual Paradigm was selected as a tool for the procedure design for validating nonfunctional requirements in Information Management Systems, for the advantage of having a user interface easy to use and allows diagrams necessary for the procedure development, it also generates documentation in HTML and PDF formats, without using external tools; and availability on multiple platforms [10].

Prospective Methods study the future, in regards to the evolution of the factors of techno-socio-economic environment and their interactions. The Experts method (Delphi method) is based on consultation with people who have great knowledge about the environment in which the organization carries out its work. These people express their ideas and finally a report is made indicating which are, in their opinion, the possible alternatives for the future [11]. For the presence in the University of staff with expertise in the subject from the Quality Center for Technological Solutions (CALISOFT), the importance of an assessment of engineers performing software testing to improve the quality of products and the features mentioned before, it was decided to use the Delphi method to validate the proposal of this research.

III. RESULTS AND DISCUSSION

The proposal for the procedure to validate nonfunctional requirements in Information Management Systems is presented as follows.

Main features of the procedure

The procedure consists of a series of activities organized in a logical and sequential way according to the element being analyzed. The implementation of the procedure is an iterative and incremental process by its own nature, that is, several iterations are performed while validating requirements do not meet the characteristics specified; allowing developers to run multiple sequences gradually, meaning that, as time goes by on the calendar each one of these produces an increase in the software quality.

It fits any software development methodology. The artifacts and proposed roles in it are declared by the authors according to their research. It is proactive, for the preventive nature of its activities, since it indicates how to develop correctly, in a coherent and organized way, each activity and task. It has a wide range of applicability, given the feasibility of its use for various types of projects during the validation activity as part of the Requirements Engineering. It also maintains an approach on feedback from its participants, fostered by the implementation of its activities.

Objective: To validate the nonfunctional requirements for the detection and correction of errors during the development life cycle of Information Management Systems.

Scope: Information Management Systems.

Structure: The procedure is structured in three phases to facilitate the organization of the activities carried out and a satisfactory range of the objective proposed. The phases are

named: Phase I. Validation of Nonfunctional Requirements List, Phase II. Validation of NFR Specifications and Phase III. Validation for Quality Metrics.

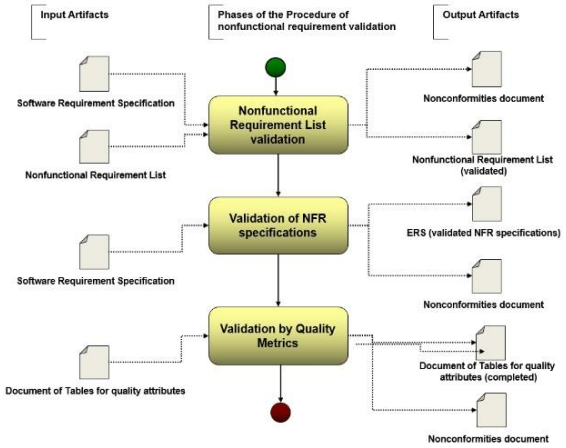

Fig. 2 Phases of the Procedure to validate nonfunctional requirements.

The activities of the three phases of the procedure include the following elements:

Description: Consists of explaining in detail what to make those involved in each activity and how defined techniques are used to fulfill the stated goal. Describe the treatment that will be given to the inputs of the activities to produce outputs.

Objective: This is the primary goal of the activity; defines the purpose of it, towards which the people involved in its implementation work.

Responsible: Plays a leading role in the development of activities. Main responsible of input and output devices, and the work flow done.

Participants: People in charge of materializing correction activities of detected errors. Their main task is to work together with the responsible of the procedure.

Activities: Set of actions that are based on the stage of the procedure, aiming to achieve its goal. Each activity consists of a sequence of tasks or steps logically ordered.

Input Artifacts: Composed of all the information and documentation necessary for the implementation of activities. Participants make use of them, processing them to obtain the outputs for each particular activity.

Output Artifacts: Documents, models, tables and general information obtained as a result of the implementation of activities. Some of them are the entries of other activities that in turn are used to generate other outputs.

Techniques: Used to enable participants in each of the activities, gathering and obtaining information necessary for its implementation. Provides an exchange between actors promoting their good understanding and feedback.

Phase I. Validation of Nonfunctional Requirements List

Phase I is focused on analyzing each nonfunctional requirement, contained in the List of Nonfunctional Requirements. This artifact is generated by the quality administrator, which makes a request to the SRS analyst and puts the statements on the list, separated from its specifications and considering the already defined classifications of all existing NFR, as follows:

<Classification>

NFR<Number>:<Statement>

The statement of the NFR is verified regarding ambiguity, consistency; possibility to be proved and that describes properties of the system. Permitting to obtain nonconformities from the errors found, which are stated in the DNC to be analyzed and corrected by the analyst.

Many iterations are performed as necessary until it is accomplished, ranging from 95% to 100%, the goals of each task. The percentage is defined with these limits letting exist from one to three errors in relation to the amount of NFR because its correctness depends on the opinion of the analyst responsible for this activity. The percent of achievement is specified in Table 1: Deficiencies of NFR, found in the document Nonfunctional Requirements List (see Annex 1). It also shows those concepts that may be difficult to interpret in the implementation of these activities.

TABLE I
PHASE 1. VALIDATION OF NONFUNCTIONAL REQUIREMENTS LIST

Phase 1. Validation of Nonfunctional Requirements List	
Objective	Validate the statements of nonfunctional requirements.
Responsible	Quality Administrator.
Participants	Analyst, Principal of the project, Client.
Input Artifacts	Nonfunctional Requirements List. Software Requirement Specification Document.
Output Artifacts	Nonfunctional Requirements List. (validated). Nonconformities Document.
Activities	Detail in the List of Nonfunctional Requirements all the statements of NFR of the application. Verify the statements of the NFR are: • Unambiguous and correct. • Possible to prove • Described as constraints or properties of the system. • Concise and abstract. Specify errors found in the DNC.
Technique	Reviews.

Phase II. Validation of NFR specifications

Phase II is directed to review the specifications of nonfunctional requirements for software, checking first that meet with the parameters set by the IEEE 830 standard. Activities are ruled by what this standard indicates, as the desirable features for correct specification of software requirements.

Once corrected the stated errors of NFR (Phase I), it becomes necessary to check the specification found in the ERS as follows:

<Classification>

NFR<Number> : <Statement>

<Specification>

Several iterations of this step are performed as often as necessary, that is, until the target for each activity is accomplished within the range of 95% to 100%. The percentage is defined by these limits due to there may be one to three errors in relation to the number of existing NFR, that cannot be eradicated. This problem is evident because the correction of non conformities depends on the opinion of the analyst responsible for this activity. The technique used is audit, which will be implemented through a checklist. It not only evaluates the SRS artifact in general, but also establishes a series of questions for each activity to ensure proper development and calculate the percentage of achievement.

TABLE II
PHASE II. VALIDATION OF NFR SPECIFICATIONS

Phase II. Validation of NFR specifications	
Objective	Validate the nonfunctional requirements specifications.
Responsible	Quality Administrator.
Participants	Analyst, Principal of the Project.
Input Artifacts	Software Requirement Specification Document..
Output Artifacts	Software Requirement Specification Document (with specifications of NFR validated). Nonconformities Document.
Activities	• Verify that the specification of NFR is correct. • Verify that the specification of NFR is unambiguous. • Verify that the specification of NFR is complete. • Verify that the specification of NFR is possible to prove. • Verify that the specification of NFR is consistent. • Verify that the specification of NFR is modifiable.
Technique	Audit.

Phase III. Validation for Quality Metrics

Phase III is focused on evaluating the six quality attributes contained in the ISO / IEC 9126 standard that match the selected nonfunctional requirements. The metrics set, using the same standard as a guide, allow regulating the features and sub-features associated with non-functional requirements.

Validations take into account the metrics that best meet the current needs of the NFR. An analysis of the results obtained is performed after the application to adapt them from quantitative to qualitative, which subsequently allows defining the percent of achievement of the characteristic considered. All these calculations and conversions are listed in the Document of Tables for quality attributes. Several iterations are performed to achieve accomplishment of each quality attribute from a range of 90% to 100%, because there may be NFR that are not evaluated with the metrics proposed.

Structure metrics

For a better understanding of the metrics it has been defined a common structure for its approach:

- Metric Name: Name of the metric.
- The metric is proposed to measure: Question to be answered with the application of the metric.
- Application Method: Provides a sequence of steps for the application of the metric.
- Measurement (formula): Provides measurement formula and meaning of the data used.
- Interpretation of the value obtained: Provides the range for limiting the value obtained and its conversion to a qualitative value.
- Unit of measure: Standardization of the measurement being performed.

TABLE III
PHASE 3. VALIDATION FOR QUALITY METRICS

Phase 3. Validation for Quality Metrics	
Objective	Validate the quality of NFR through the metrics defined.
Responsible	Quality Administrator.
Participants	Architect, Principal of the Project.
Input Artifacts	Document of Tables for quality attributes.
Output Artifacts	Document of Tables for quality attributes. (completed). Nonconformities Document.
Activities	• Validate the NFR associated to functionality. • Validate the NFR associated to reliability. • Validate the NFR associated to usability. • Validate the NFR associated to efficiency. • Validate the NFR associated to maintainability. • Validate the NFR associated to portability.
Technique	Validation Tests.

Representation of the results of the nonfunctional requirements

With the application of metrics to NFR, according to the results, three cases are established: Appropriate Case, Worst Case and Recognized Case, with the percentage of achievement it means. The sum of the percent of cases defined represents the state of the attribute sub-feature being evaluated.

If any of the metrics obtain a result which puts it in Appropriate Case: it means that although the NFR is implemented, there are few poor elements that do not allow proper operation; or Worst Case: means that there are NFR with implementation to be done or poor; it is registered nonconformity. Permitting to go directly to the affected area which includes the requirement, not to generalize its condition, avoiding ambiguities.

The Document of Tables for quality attributes contains a summary table at the end that specifies the percentage of

achievement of the attributes (NFR) that are measured by the metrics, and the final amount of non conformities found. It will decide whether a new iteration of the phase is done, that is, if the percentage of achievement of each quality attribute is not between 90% and 100%, another iteration will run.

Implementation of the procedure

The procedure is applied to SIGEPAC, a computer tool to register, follow up, and assess financial and physical performance of projects, as well as the impact of its results. The software solution is composed essentially by two subsystems: software management and dataware house. Herein are the results obtained in the implementation of the proposal designed.

Implementation of phase 1

Generated by the quality manager, the List of Nonfunctional Requirements is the first document which according to the proposed procedure is applied to the validation. After capturing the title of all the non-functional requirements separated from their specification and respecting the classification proposed by the SRS, it is made a detailed analysis by the quality manager to ensure fulfilling the objectives of this phase.

It was detected in a first iteration that only 85.41 % of the NFR were unambiguous, and in the second iteration 96% had that feature. Example of refined requirements where ambiguity has been removed:

Initial requirement: NFR 4. Allows keyboard use to perform operations on the system (Allow quick access to the system using the keyboard)

Corrected requirement: NFR 4. Allows keyboard use for system access operations.

Some other errors detected during this phase of the procedure are that in the first iteration only 93.75 % of the requirements were concise and abstract, in the second iteration 98% met this feature, that is, they were able to provide greater amount of information with less amount of words, allowing to abstract as much as possible of what can be the future system. As an example of an abstract and concise requirement we have:

NFR 37. Define communication interface: nonfunctional requirements that allowed a detailed abstraction of how the future of the system would be regarding the communication interface.

Continuing the implementation of the procedure it was found that 96% of the NFR were described as a restriction or properties of the system, making possible in the second iteration that 100% of the requirements meet this feature. NFR possible to prove in the first iteration were at 93, 75%, and in the second iteration 100% of them had this characteristic.

According to the method proposed in this paper, all errors detected are in a Non-conformities Document, which is sent to analysts, which in turn check it and applying the correction technique again meet with clients and the principal of the project to correct them. It was necessary to make a total of two iterations of this phase to validate the list of non-functional requirements.

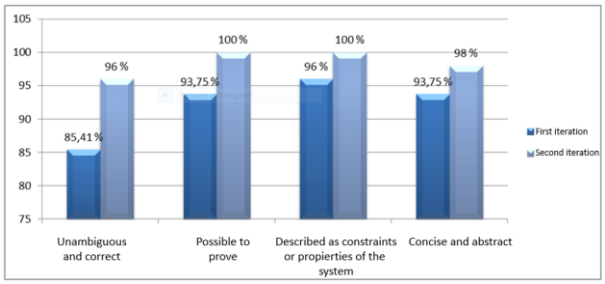

Fig. 3 Results of Phase I

Implementation of Phase II

After being analyzed the NFR separated from their specifications and non conformities detected, which are solved by analysts, the refinement of the Software Requirements Specification Document (SRS) is performed; answering after that the questions in the checklist of this phase.

Example of the implementation of checklists to the SRS document.

Elements defined by activities on phase II.					
Activity 1: Ckech the SRS to be correct.					
Importance	Parameters to assess.	Eval	(NP)	Amount of elements affected.	# Non conformity
	1. Are all non-functional requirements requested by the customer present?	0			
critical	2. All specified NFR contribute to satisfy a real need of the software?	0			
	3. Are there NFR with lack of information by the client?	1		2	10, 11
	4. Are there NFR with added information by the analyst?	1		3	12, 13, 14
	5. Is the source of NFR identified? (for example: a person, a regulation)	0			

In the SRS document they were detected that 10.41 % of a total of 48 non-functional requirements, were incorrect. 2.08% have problems of ambiguity, they are neither complete not modifiable. 4.16% are not consistent, but there were no critical questions graded as wrong, that is to say, defined with 1.

To make the SRS document meets all the features required in the process it was clearly evident the need for two iterations, although most of non-functional requirements met

all the characteristics established for this phase, the correction characteristic was not found in the defined range (95% to 100 %). The following data were obtained as results.

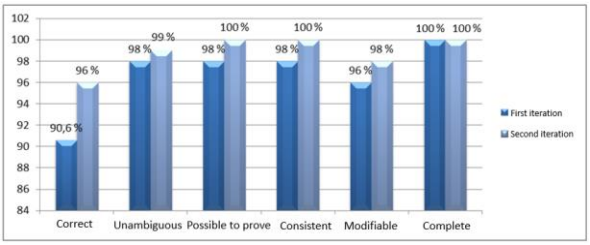

Fig. 3 Results of Phase II

Implementation of Phase III

For the implementation of Phase III it was necessary to run the metric established by quality attributes, which allow us a percentage assessment of the six characteristics of quality desirable for the software and set by the ISO / IEC 9126 standard.

Functionality

In the Functionality were used metrics as access control and user accounts to measure the security sub characteristic. All test cases were carried out in the SIGEPAC security module, from which satisfactory results were obtained since there were no violations to the system access. In the metric of user accounts the results were similar, as there is only one shared account (administrator), this is responsible for managing user accounts by giving them the necessary permissions for the use of the system functionalities.

Some other metrics that evaluate the functionality, in terms of interoperability, are data interchangeability, format based, data interchangeability, based on successful attempt, where tests cases were performed to the four Pentaho components, Pentaho BI Suite Community Edition in its 3.5 version, which are specified in the NFR 30: Business intelligence layer. Important to point out that although there were few poor data exchanges, there were actually problems with the component Pentaho Desing Studio 3.5.

After adding the percentage of the two sub-characteristics it was determined that the functionality is 95% complete.

Reliability

Measurements made on the reliability grant it 90%, since the percentage was fulfilled in terms of failure eradication, there was a qualitative assessment of appropriate case for mean time recovery metrics and mean time between total failures, as there were total failures in the interaction with the Pentaho Desing Studio 3.5 component that did not meet the NFR 14 that sets a range of recovery going from 10 minutes to 72 hours for the solution of the problem, validation and testing.

Usability

Usability is 95%. Although in the SRS are not specified the amount of tutorials the application must have, a necessary aspect to implement the accessibility metric to tutorials, if it is detailed that you must implement a help. Out of the 12 requirements related to usability, one is not fully implemented (NFR 12: Change component without the need to authenticate again).

Efficiency

The characteristic of efficiency, which is evaluated by the metrics response time and user waiting time when using the equipments E/S is 100% according to the NFR measured. For the first metric used it is applied NFR 18. System response time, which sets five minutes as the maximum time for answers, and for the second NFR 4. Allow the use of the keyboard for system quick access operations. There are requirements as NFR 21. Number of users connected simultaneously, its implementation is not measured by the metrics established and it has not been possible to prove it.

Maintainability

Maintainability, which is measured by the metric implementation degree of diagnostic functions, is 95% since not all the registered failures were diagnosed with such diagnostic functions.

Portability

It important to say, that this feature in the project management module, where the procedure is applied, is 100% implemented, because it is easy to install by the user as this module is supported on a web application and moreover all NFR concerning portability have been successfully developed.

With the implementation of the Phase III to the project Integral Solution for Project Management and Centralized Actions SIGEPAC concludes the implementation of the procedure. The results were analyzed making it necessary to perform two iterations of Phase I and Phase II. After this example we can say that the procedure is easy to understand and apply, with necessary practical contribution to software specialists dedicated to validate non-functional requirements.

IV. CONCLUSIONS

With the completion of this research a procedure for validating nonfunctional requirements of Information Management Systems was designed, coming to the following conclusions:

- Standards and quality rules were analyzed, as part of the implementation of the theoretical basis of the research, including also the conceptual context, analyzing subjects as the study of requirements engineering, which proved as a result that validation of requirements and software metrics contributes to the control, monitoring and improvement of the quality of the software development process.
- A procedure capable of improving the quality of nonfunctional requirements of Information Management Systems was designed; being necessary for the proposal to define a set of activities, responsible, participants, input and

output artifacts and techniques that led to its best understanding.

- The Integral Solution for Project Management and Centralized Actions was applied, determining the feasibility of the procedure.

REFERENCES

[1] SOMMERVILLE,I. Software Engineering. 8th ed., 2004,p. 121
[2] ISO / IEC 9126 -1991.
[3] IEEE (2004). SWEBOK. Guide to the Software Engeneering Body of Knowledge.
[4] Duran Toro, Amador. Metodología para la elicitación de requisitos de software. Sevilla, Universidad de Sevilla. . 2002.
[5] Escalona, M. Koch, N. Ingeniería de requisitos en aplicaciones para la web. Un estudio comparative..Universidad de Sevilla.: s.n., 2002
[6] L. C. Briand, J. W. Daly y J. K. Wust. A Unified framerwork for Coupling Measurement in Object-Oriented Systems. 1999. pp. 91-121. Vol. IEEE transactions on software engineering.
[7] Robles, M.C..E.A. Métricas para la Gestión de Proyectos de Software. p. 65.
[8] IEEE Std 830 (1998), IEEE Guide for Developing System Requirements Specifications, http://www.standards.ieee.org
[9] Ingeniería de Software. Calidad del Producto (ISO 9126). ISO, 2001. 2001, Norma Cubana, Vols. Part 1: Modelo de Calidad(ISO/IEC 9126-1:2001).
[10] Visual Paradigm. Visual Paradigm. [Online] [Cited: 12 06, 2012.] http://www.visual-paradigm.com/.