- ORIGINAL ARTICLE -

Applying an Improving Strategy that embeds Functional and Non-Functional Requirements Concepts

Aplicando una Estrategia de Mejora que incluye Conceptos de Requisitos Funcionales y No Funcionales

Pablo Becker, Guido Tebes, Denis Peppino and Luis Olsina

GIDIS_Web, School of Engineering, UNLPam, General Pico, La Pampa, Argentina
beckerp@ing.unlpam.edu.ar; guido_tebes@hotmail.com
denispeppino92@gmail.com; olsinal@ing.unlpam.edu.ar

Abstract

Organizations should set and reach business goals for varied purposes using the suitable strategies. Basically, a strategy specifies the activities, methods and another related resources that should be considered in order to achieve a given goal purpose. Goal purposes and their associated strategies can aim at evaluating, testing, developing, or maintaining some entity. Some concrete evaluation purposes such as to understand or monitor can be achieved by strategies embracing non-functional requirements definition, measurement, evaluation and analysis activities. Other specific evaluation purposes such as to improve or control also imply changing the target entity; therefore, strategies should embrace functional requirements definition activities as well. Moreover, specific development and maintenance purposes always involve functional requirements. In this work, we relate business and information need goals with functional and non-functional requirements concepts, which are paramount for well-defined strategies. Therefore, we specify vocabularies for them, and illustrate the applicability of an improving strategy –which embeds these concepts- in the context of a running example. Having well-structured vocabularies serving as common ground for diverse strategies may promote a more effective operationalization of projects dealing with evaluation, testing, development and maintenance goal purposes.

Keywords: Functional and Non-Functional Requirements, Improving Strategy, Ontology, Purpose, Vocabulary.

Resumen

Las organizaciones deben establecer y alcanzar metas de negocio para diferentes propósitos utilizando las estrategias adecuadas. Básicamente, una estrategia especifica las actividades, los métodos y los recursos relacionados que deben considerarse para lograr un determinado propósito. Los propósitos de las metas y sus estrategias asociadas pueden apuntar a la evaluación, prueba, desarrollo o mantenimiento de alguna entidad. Algunos propósitos específicos de evaluación, como comprender o monitorear, pueden lograrse mediante estrategias que abarcan actividades de definición de requisitos no funcionales, medición, evaluación y análisis. Otros propósitos de evaluación, como mejorar o controlar, implican además cambiar la entidad o su contexto; por lo tanto, las estrategias también deben incluir actividades de definición de requisitos funcionales. En cuanto a los propósitos específicos de desarrollo y mantenimiento, estos siempre implican requisitos funcionales. Este trabajo relaciona las metas de negocio y de necesidad de información con conceptos de requisitos funcionales y no funcionales, que son fundamentales para estrategias bien definidas. Por lo tanto, especificamos sus vocabularios e ilustramos la aplicabilidad de una estrategia de mejora –la cual embebe estos conceptos- mediante un ejemplo que desarrollamos a lo largo de las secciones. Tener vocabularios bien estructurados que sirvan de base común para diversas estrategias puede promover una operacionalización más efectiva de los proyectos que tienen que ver con propósitos de metas de evaluación, prueba, desarrollo y mantenimiento.

Palabras claves: Mejorar, Ontología, Propósito, Requisitos Funcionales y No Funcionales, Vocabulario.

1. Introduction

Requirements Engineering typically differentiate two types of requirements: functional requirements (FRs), which specify the features and capabilities that an entity (e.g., a system) shall do, and non-functional requirements (NFRs), which specify the quality constraints (e.g., security, performance, usability) of its features and capabilities. In general, in order to accomplish a given organizational goal such requirements should be derived accordingly. Moreover, organizations set and reach business goals for different types of purposes using strategies. These purposes can be classified into evaluation, testing, development, and maintenance (change) categories.

On one hand, business goals are the primary goals that an organization intends to achieve setting them at different organizational levels. Commonly, software projects in conjunction with strategies operationalize specific purposes of the business goals. The purpose of a goal is the rationale for achieving it. For instance, to realize specific development and maintenance purposes such as to create, add, delete or modify concrete features and capabilities of an entity, the statement of FRs is always involved. Moreover, being aware that usually FRs may require to be satisfied by NFRs constraints, specific evaluation purposes play also a key role in this endeavor. In this sense, specific evaluation purposes are for example to understand, monitor, control, improve and select.

On the other hand, information need goals are support goals for business goals. They provide useful information in order to know the level of achievement of business goals as well as to give additional information to reach them. An information need goal may also require measurement and evaluation information need goals, which are always stated by NFRs specifications.

As aforementioned, the goals established in an organization are operationalized through projects using strategies. So, various strategies can be used for helping to achieve the different project goal purposes. Basically, a strategy specifies the activities, methods and related resources such as tools and agents that should be considered to achieve the goal. In other words, a strategy specifies what should be done and how should be performed. In [1], a family of strategies driven by measurement and evaluation –i.e., strategies for evaluation purposes- was illustrated. For instance, some specific evaluation purposes such as to understand, monitor and select the suitable alternative may be achieved by strategies embracing NFRs definition, measurement, evaluation and analysis activities. But other evaluation purposes such as to improve also imply changing the entity, so FRs definition activities are implied as well.

We have conceived a strategy as having three integrated pillars, namely: i) process specifications, ii) method specifications, and iii) well-established domain vocabularies or conceptual bases [2].

The process specifications describe a set of activities, tasks, input and output artifacts, roles, and so forth to reach a specific goal purpose. Besides, process specifications can consider process perspectives such as functional, behavioral, informational and organizational [3]. Usually, process specifications primarily state what to do rather than indicate the particular methods and tools used by specific activity descriptions. The method specifications (such as metric and indicator specification templates), represent the particular ways to perform the activity descriptions. Finally, the domain conceptual bases explicitly establish the needed minimum terms, such as project, business

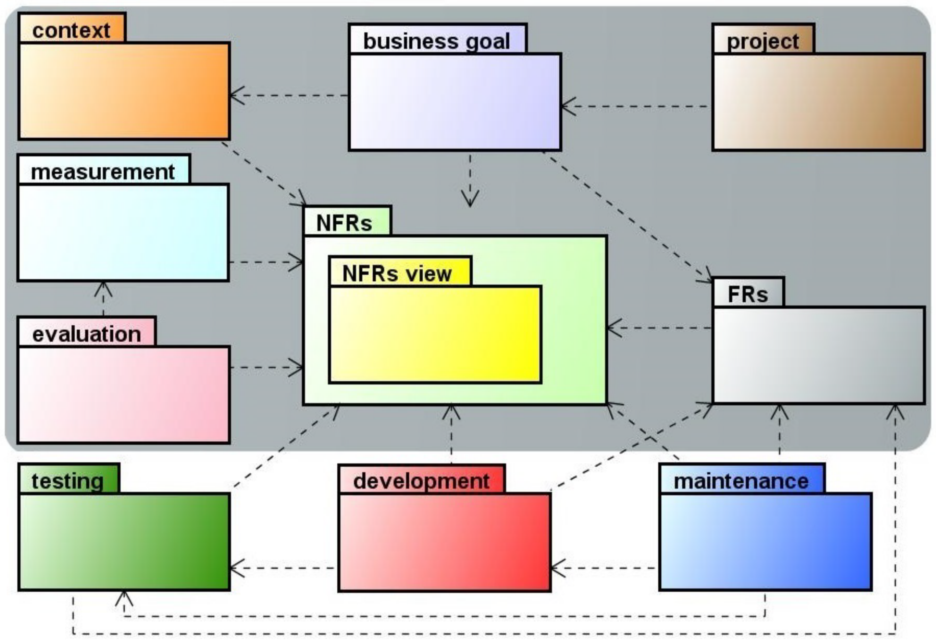


Fig. 1 Relationships between the business goal conceptual component with the project, context, functional and non-functional requirements (FRs and NFRs) components. Also, other related components or packages are shown.

goal, measurement, metric, indicator, NFR, FR, amongst others, to specify processes and methods in a consistent way for a set of families of strategies. Also, a robust conceptual base, should establish axioms in addition to main properties of terms and their relationships.

In previous works [2, 4, 5], we have developed a set of ontologies that deal with measurement, evaluation, NFRs, context, project and business goals. Recently, in [6], we have partially documented the FRs ontology. In the current work, we extend [6] by adding tables with the definitions of attributes and relationships for the FRs component.

The shaded conceptual components in Fig. 1 depicts the ontologies already developed. Note that we also show the testing conceptual component relating it with the previously built ontologies. This new component is an ongoing work, which is not documented in this paper. Consequently, the integration of all conceptual bases relates business and information need goals concepts with FRs and NFRs concepts. This integration is paramount for specifying processes and methods for our families of strategies driven by measurement and evaluation that deal with both NFRs and FRs. Ultimately, having well-structured vocabularies serving as common ground for diverse strategies may foster a more effective operationalization of projects dealing with evaluation, testing, development and maintenance goal purposes.

The contribution of this work is illustrating the applicability of these vocabularies in a particular strategy –that for space limits we could not show in [6]. That is to say, we use in a practical case a strategy for the improvement goal purpose, which embeds both FRs and NFRs concepts. Particularly, we describe with details the improving strategy's processes and methods, which are grounded in the abovementioned conceptual components. In turn, we describe how to place these components in an ontological conceptual architecture, as we see later.

The remainder sections of this paper are organized as follows. Section 2 motivates the present work by introducing a running example, which illustrates a specific evaluation business goal that requires mandatorily both FRs and NFRs concepts. Particularly, we use the GOCAMEC (*Goal-Oriented Context-Aware Measurement, Evaluation and Change*) strategy [1] for the purpose of improving the external quality of the JGUIAr[1] system. This practical case will permit the reader to gauge the applicability of the integrated conceptual framework not only for this improving strategy but also other purpose-oriented strategies. Section 3 specifies the integrated vocabulary and outlines a four-layered ontological conceptual architecture where components can be placed into. Sections 4 and 5 use the instantiated terms from the integrated vocabulary and show their applicability in the GOCAMEC's process and method specifications for the given running example. Section 6 analyzes related work. Finally, Section 7 draws conclusions and outlines future work.

2. Motivating Scenario

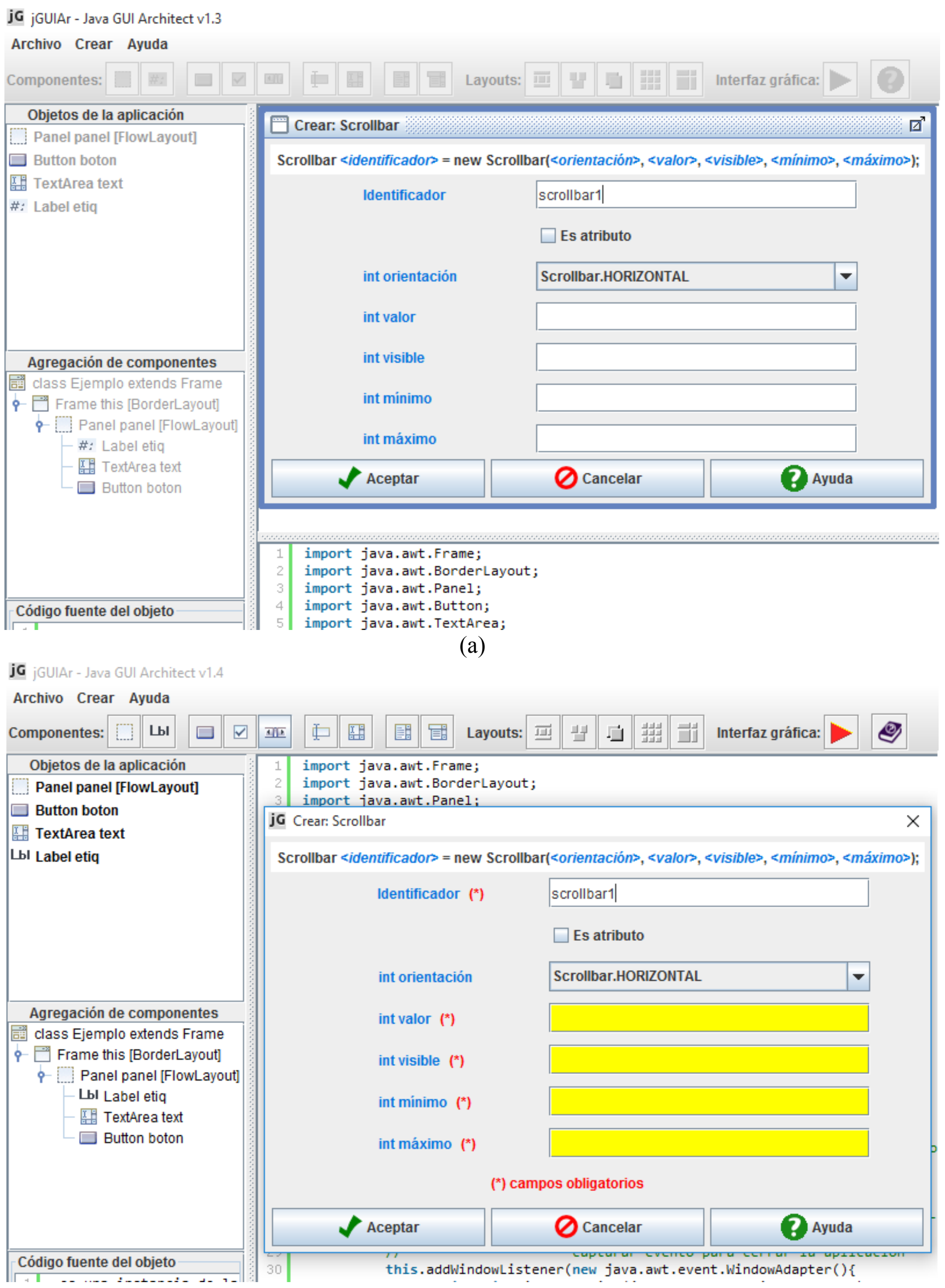
Here, we motivate the running example that will be reassumed in Sections 4 and 5. JGUIAr is a supporting tool to assist undergraduate students in the learning process of designing GUIs and looking at the generated Java code. It was introduced in our School in the Object-Oriented Programming (OOP) course since 2014. As in any developed software application, if we would decide at any moment to evaluate and analyze its current quality state, surely opportunities for improvement will raise. And this was the case, where recently we were planning and performing a new improvement cycle on JGUIAr.

Hence, the main (business) goal set at operational level was to improve at least 15 percentage points (p.p.) the JGUIAr v1.3's external quality in the period of 3 months. This endeavor should be finished before April 1st, 2018, the moment when the undergraduate students should use the tool for specific course assignments. Particularly, considering the previous experience gained observing and evaluating its usage, we decided to concentrate on potential weak attributes related to Usability, Information Quality, and Functional Quality characteristics [7] from the external quality focus standpoint.

For example, Fig. 2a shows that for the given application screen, at the moment that emerge the window to create the "Scrollbar" component, those fields related to constructor's parameters are not highlighted as mandatories. So a potential Usability attribute to be evaluated is "User error prevention for mandatory entries", observing the problem on the emerging window, where no prevention mechanism is provided. Note that an attribute represents an elementary NFR to be satisfied. Once the evaluation is carried out, if the attribute is not satisfied as required, a change action should be designed and implemented aimed at improving the concrete entity. Fig. 2b shows the new situation after changing the emerging window.

In a nutshell, the abovementioned goal embeds the improvement purpose, which implies both evaluation and change activities. In order to help

[1] **JGUIAr** (*Java Graphic User Interface Architect*) is a tool for designing Java AWT GUIs developed by Dr. Hernan Molina, in the context of an Object-Oriented Programming course in the Engineering School at UNLPam (https://sites.google.com /site/jguiarsoftware/)



Scrollbar <identificador> = new Scrollbar(<orientación>, <valor>, <visible>, <mínimo>, <máximo>);

Identificador	scrollbar1
	☐ Es atributo
int orientación	Scrollbar.HORIZONTAL
int valor	
int visible	
int mínimo	
int máximo	

```
import java.awt.Frame;
import java.awt.BorderLayout;
import java.awt.Panel;
import java.awt.Button;
import java.awt.TextArea;
```

(a)

```
import java.awt.Frame;
import java.awt.BorderLayout;
import java.awt.Panel;
```

Scrollbar <identificador> = new Scrollbar(<orientación>, <valor>, <visible>, <mínimo>, <máximo>);

Identificador (*)	scrollbar1
	☐ Es atributo
int orientación	Scrollbar.HORIZONTAL
int valor (*)	🟨
int visible (*)	🟨
int mínimo (*)	🟨
int máximo (*)	🟨

(*) campos obligatorios

```
        this.addWindowListener(new java.awt.event.WindowAdapter(){
```

(b)

Fig. 2 Two JGUIAr application' screens: a) The JGUIAr v1.3's Dialog shows that no error prevention mechanisms for mandatory entries are provided; b) The JGUIAr v1.4's Dialog depicts that error prevention mechanisms for mandatory entries are provided.

achieving this goal purpose, which belongs to the evaluation category, the suitable strategy should be used. The selected strategy should support at least the following three aspects: i) to understand the current quality state for the evaluable entity and recommend changes, mainly for the weakly benchmarked attributes/indicators; ii) to make changes on the developable (concrete) entity or its subentities; and iii) to understand the ulterior quality state (the improvement) after performed changes.

Therefore, the evaluation and change's activities and methods specified for the improving strategy require to instantiate not only NFRs concepts but also FRs concepts. Consequently, by having explicit vocabularies serving as common ground for diverse strategies may benefit a more effective operationalization of goal purposes.

3. Linking Business and Information Need Goal Terms with Functional and Non-functional Concepts in an Integrated Conceptual Framework

There exist different ways to structure a conceptual base such as glossaries, taxonomies and ontologies, among others. Briefly, a glossary is an ordered set of terms and their definitions; a taxonomy is a collection of terms -like a glossary-, but usually organized by kind-of semantic relations regarding a hierarchic structuring. Finally, an ontology includes terms, their definitions, properties and different types of relationships among terms in addition to axioms. According to [8], "an ontology may take a variety of forms, but necessarily it will include a vocabulary of terms, and some specification of their meaning. This includes definitions and an indication of how concepts are inter-related which collectively impose a structure on the domain and constrain the possible interpretations of terms. An ontology is virtually always the manifestation of a shared understanding of a domain that is agreed between a number of agents. Such agreement facilitates accurate and effective communication of meaning, which in turn leads to other benefits such as inter-operability, reuse, and sharing". Therefore, we can state that an ontology is a richer mechanism than other approaches for structuring a conceptual base. The main objective to build ontologies can be manifold, such as: to share a common understanding and then facilitating the communication among people; to reuse and integrate the disparate and heterogeneous representations; to formalize the representation of a domain problem or theory; and, as the basis to support semantic reasoning to full-fledged knowledge-based applications, among other aims.

In previous works [2, 4, 5, 6, 9, 10], we have built a set of ontologies that deal with project, business goals, NFRs, measurement, evaluation, context and FRs (recall shaded components in Fig. 1). In this Section, we extend [6] by adding attribute and relationship definitions for the FR ontology and integrating all the previously built ontologies to produce a harmonized conceptual framework. The integrated vocabulary relates business goals with FRs and NFRs concepts, among others, which is paramount for specifying processes and methods for our families of strategies driven by measurement and evaluation that deal with both NFRs and FRs. This

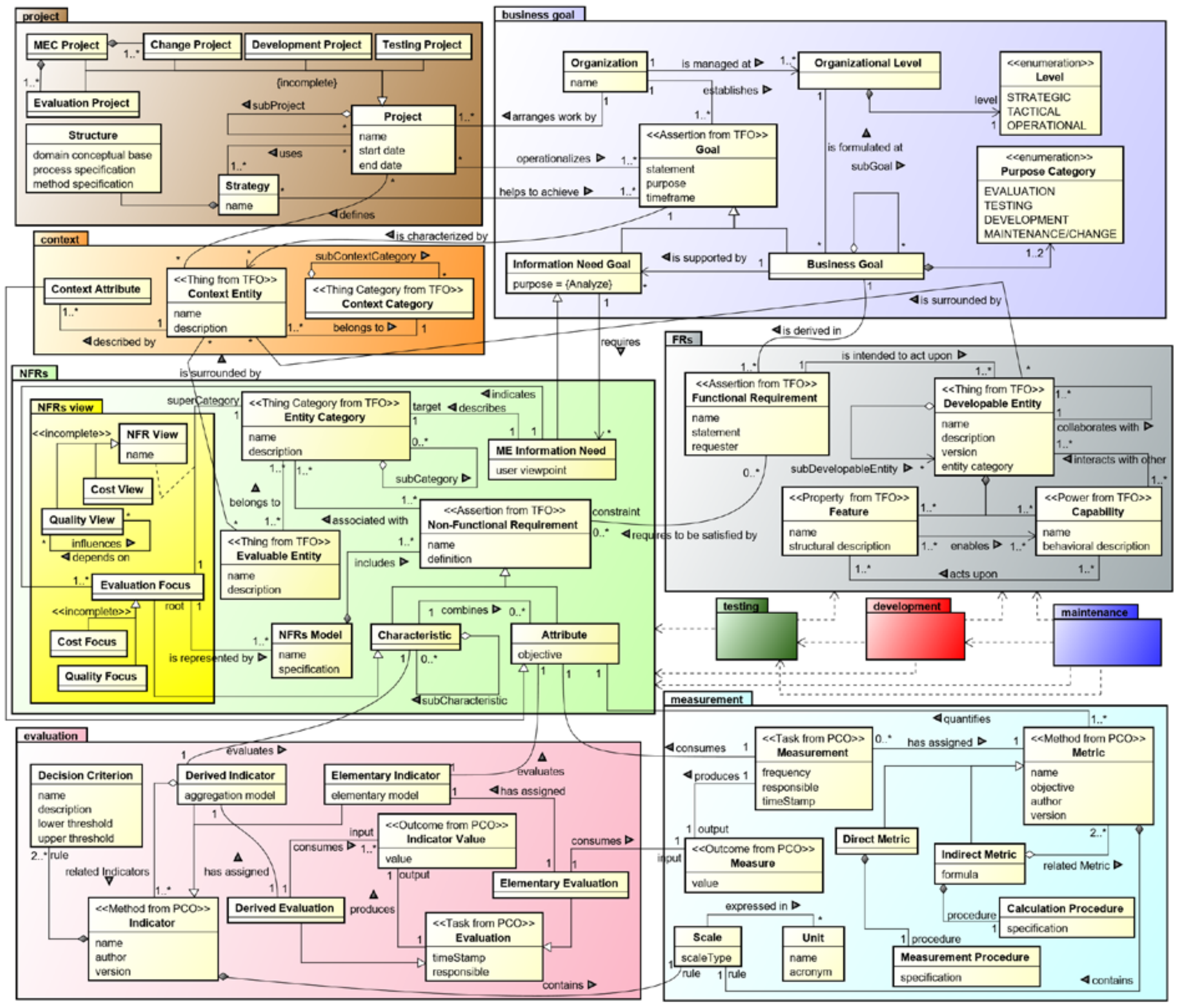


Fig. 3 Main terms, attributes and relationships for the project, business goal, context, NFRs, measurement, evaluation and FRs subontologies. Note that ME stands for Measurement and Evaluation, MEC for Measurement, Evaluation and Change, PCO for Process Core Ontology and TFO for Thing Foundational Ontology.

integration allows a shared understanding for all strategies, and promote more consistent specifications for processes and methods. In Section 5, we illustrate the GOCAMEC strategy and show how the integrated vocabulary is used to specify NFRs, FRs, metrics and indicators.

Additionally, in the end of the present Section, we introduce a four-layered ontological conceptual architecture in which these conceptual components (process, business goal, context, NFRs, FRs, etc.) may be placed into accordingly.

Note that in this paper, we address the conceptual bases representation and their integration rather than the ontology construction process itself. Nevertheless, the stages proposed in the METHONTOLOGY [11] approach were followed such as specification, conceptualization, formalization and integration. The integration stage was done by relating the different subontologies and doing some slight changes due to the harmonization of the conceptual framework as a whole.

Using the UML class diagram [12] for representation and communication ends, Fig. 3 shows the main terms, attributes and relationships for the project, business goal, context, NFRs, measurement, evaluation and FRs conceptual components. In addition, Table 1 includes the definition of terms for these subontologies. Moreover, in Tables 2 and 3 we also include the attribute and relationship definitions considered for the FRs subontology, which was not previously published in [6]. We next describe these components, highlighting some terms, attributes and relationships in italic the first time they appear in the text.

An Organization establishes Goals, which contain an explicit declaration (statement) about the primary purpose that should be achieved in a period of time (timeframe), and is characterized by a given Context. The purpose of a goal is the rationale for achieving it (e.g., to understand, improve, modify, create, verify, and review, among many others).

Table 1 Term definitions for the business goal, project, context, FRs and NFRs sub-ontologies that are shown as UML packages in Fig. 3.

Term	**Definition**
Business Goal Sub-ontology	
Business Goal	It is a main or primary Goal that the Organization intends to achieve.
Goal (synonym: Organizational Goal, Objective)	The statement of the aim to be achieved by the Organization which considers the propositional content of a purpose in a given time frame and context.
Information Need Goal	It is a Goal intended to get insight for a given Business Goal. Note: Information Need is defined in ISO 15939 [14] as "*Insight necessary to manage objectives, goals, risks, and problems*".
Organization	A concrete entity comprising people that is structured and managed to establish and pursue organizational Goals and is affected by and affects to its environment or context.
Organizational Level	It represents a management and decision-making level in which Organization's Business Goals are formulated and Information Need Goals are taken into account. Note: Usually, long-term Business Goals are formulated at STRATEGIC Organizational Level, while short-term Business Goals are formulated at OPERATIONAL Organizational Level.
Context Sub-ontology (Note that the vocabulary depicted in Fig. 3 for this component is a subset of the terms specified in [9])	
Context Category	It is a category to which concrete Context Entities belongs to. Note: Context Category has semantic of Thing Category.
Context Entity (synonym: Concrete Object/Entity)	A concrete object surrounding a target entity that represents part of the situation, which is relevant for a particular organizational Goal. Note 1: Context Entity has semantic of Thing [15]. Note 2: In the above definition the target entity can refer to both the Evaluable Entity (in the NFRs component) and the Developable Entity (in the FRs component).
Context Attribute	It is an Attribute that describe the Context Entity.
Functional Requirements (FRs) Sub-ontology	
Capability	It is a power of a Developable Entity which refers to what it is supposed to do and behave.
Developable Entity (synonym: Concrete Object/Entity)	A concrete object to be developed or modified considering its Features and Capabilities. Note 1: Developable Entity has semantic of Thing [15]. Note 2: Things (Developable Entities), properties (Features) and powers (Capabilities) do not exist in spatiotemporal isolation from one another, so they all emerge simultaneously to form a unity [15]. Note 3: Examples of Developable Entities are work products (model, source code, document), systems, among others.
Feature	It is a property of an Developable Entity which refers to the intrinsic constitution,

	structure, or parts of it.
Functional Requirement (FR)	It states what the new or existing Developable Entity (or sub-Developable Entity) does, or shall do by referring to its Features and/or Capabilities considering a given requester need. Note: To be operationalized, it can state the action to be performed upon the Developable Entity.
Non-Functional Requirements (NFRs) Sub-ontology	
Attribute	It is a measurable physical or abstract property associated with an Entity Category. Note: An Attribute represents an elementary NFR.
Characteristic (synonym: Dimension, Factor, Calculable Concept)	It represents a non-elementary NFR, which combines measurable Attributes. Note 1: Characteristics can be evaluated but cannot be measured as an Attribute -at least in a non-very trivial way such as "good" or "bad". Note 2: A Characteristic can have sub-characteristics.
Entity Category (synonym: Object Category)	Object category to be evaluated by associating its Characteristics and Attributes. Note: Entity Category has semantic of Thing Category.
Evaluable Entity (synonym: Concrete Object/Entity)	A concrete object to be evaluated. Note 1: Evaluable Entity has semantic of Thing [15]. Note 2: Examples of Evaluable Entities are concrete products (model, source code, document), systems, resources, work processes, among many others.
ME Information Need	It is an Information Need Goal that are achieved by conducting measurement and evaluation activities.
NFRs Model (synonym: Calculable- Concept Model)	A set of Characteristics, and eventually sub-characteristics, and the relationships between them, which provide the basis for specifying the NFRs and their further evaluation. Note: A possible instance of a NFRs Model is the ISO 25010 quality-in-use model [13].
Non-Functional Requirement (NFR)	It represents a constraint by means of a Characteristic or Attribute to be evaluated on how or how well an Evaluable Entity performs or shall perform. Note: FRs should require to be satisfied by constraints or 'ilities', which are represented by NFRs.
NFRs View Sub-ontology (which is included the NFRs package)	
Cost Focus	It is an Evaluation Focus for cost.
Cost View	It is a NFR view for cost.
Evaluation Focus (synonym: Calculable-Concept Focus)	It is a Characteristic which represents the root of a NFRs Model.
NFR View (synonym: Calculable- Concept View)	Association relationship between one Evaluation Focus and one Entity Category. Note 1: The Entity Category must be the super-category, i.e., the highest abstraction level of an Entity Category of value to be evaluated in a given Organization. Note 2: Names of entity super-categories are *Resource*, *Process*, *Software Product*, *System*, *System in use*, among others. Note 3: Names of NFR Views are: Quality View, Cost View, etc. [10].
Quality Focus	It is an Evaluation Focus for quality.
Quality View	It is a NFR view for quality.
Project Sub-ontology (Note that the vocabulary depicted in Fig. 3 for this component is a subset of the terms specified in [2])	
Change Project (synonym: Maintenance Project)	It is a Project for operationalizing a Business Goal with the purpose of changing the current state of an Entity. Note: Different kinds of changes (e.g., adaptive, perfective, corrective changes) can be made in the maintenance phase.
Development Project	It is a Project for operationalizing a Business Goal with the purpose of developing a new Entity.
ME Project	It is a Project for operationalizing a Business Goal with the purpose of measuring and evaluating an Evaluable Entity that requires a ME Information Need.
MEC Project	It is a Project for operationalizing a Business Goal with the purpose of evaluating and changing a concrete object by performing ME-driven changes. Note: A MEC Project comprises both Change and ME Projects.
Project	An concrete entity representing a temporary and goal-oriented endeavor with definite start and finish dates, which considers a managed set of interrelated activities, tasks and resources aimed at producing and modifying unique work products (i.e., artifacts, services or results) for satisfying a given requester need.
Strategy	Principles, patterns, and particular domain concepts and framework that can be specified by a set of core processes, in addition to a set of appropriated methods and tools, as core resources, for helping to achieve the Project's Goal purpose.
Testing Project	It is a Project for operationalizing a Business Goal with the purpose of testing a testable entity.
Measurement Sub-ontology (Note that the vocabulary depicted in Fig. 3 for this component is a subset of the terms specified in [2])	
Base Measure	A Measure that does not depend upon other Measure.
Calculation Procedure	Set of established and ordered instructions of an Indirect Metric or Indicator that indicates how the described steps in an Indirect Measurement or Evaluation task should

	be carried out.
Derived Measure	A Measure that is derived from other Measures.
Direct Measurement	Measurement that produces a Base Measure.
Direct Metric	A Metric of an Attribute that does not depend upon a Metric of any other Attribute
Indirect Measurement	Measurement that produces a Derived Measure.
Indirect Metric	A Metric of an Attribute that depends of Metrics of other Attributes.
Measure	The number or category assigned to an Attribute of an Evaluable Entity by making a Measurement. Note: It is the Measurement output that represents an outcome as work product.
Measurement	A task that uses a Metric in order to produce a Measure's value. Note: This task quantifies an Attribute by producing a Measure as outcome.
Measurement Procedure	Set of established and ordered instructions of a Direct Metric that indicates how the described steps in a Direct Measurement task should be carried out.
Metric	The defined Measurement or Calculation Procedure and the Scale. Note: A Metric is a method which is applicable to the description of a Measurement task.
Evaluation Sub-ontology (Note that the vocabulary depicted in Fig. 3 for this component is a subset of the terms specified in [2])	
Derived Evaluation	Evaluation that produces an Indicator's value by assessing a Characteristic.
Derived Indicator	An Indicator that is derived from other Indicators to evaluate a Characteristic.
Elementary Evaluation	Evaluation that produces an Indicator's value by assessing an Attribute. Note: An Attribute is a non-functional elementary requirement from the Evaluation standpoint.
Elementary Indicator	An Indicator that does not depend upon other Indicators to evaluate an Attribute.
Evaluation	A task that uses an Indicator in order to produce an Indicator's value.
Indicator	The defined Calculation Procedure and Scale in addition to the Indicator Model and Decision Criteria in order to provide an evaluation of a Characteristic or Attribute with respect to a defined Information Need. Note: An Indicator is a method which is applicable to the description of an Evaluation task.
Indicator Value	The number or category assigned to a Characteristic or Attribute by making an Evaluation. Note: It is the Evaluation output that represents an outcome as work product.

Table 2 Attribute definitions for the terms included in the FRs sub-ontology.

Term	**Attribute**	**Definition**
Capability	**name**	Label or name of a Developable Entity's Capability to be identified.
	behavioral description	An unambiguous textual statement describing the behavior or action that is enabled by a Developable Entity's Feature.
Developable Entity	**name**	Label or name of a Concrete Object to be identified.
	description	An unambiguous textual statement describing the Concrete Object.
	version	Unique identifier, which indicates the level of evolution of the Concrete Object.
	entity category	Object category or class to which the concrete Developable Entity belongs to.
Feature	**name**	Label or name of a Developable Entity's Feature to be identified.
	structural description	An unambiguous textual statement describing the Developable Entity's Feature.
Functional Requirement (FR)	**name**	Label or name of a FR to be identified.
	statement	An explicit declaration of what the new or existing Developable Entity does, or shall do.
	requester	An agent that requires or establishes the FR.

Table 3 Relationship definitions included in the FRs sub-ontology.

Relationship	**Definition**
acts upon	The Capability of a concrete object acts upon one or more of its Features
collaborates with	A Developable Entity can cooperate with one or more concrete entities.
enables	One or more Features of a concrete object empower one or more of its Capabilities.
is composed of	A Developable Entity is composed of one or more Features and Capabilities.
is derived in	A Business Goal is derived in one or more FRs.
is intended to act upon	A FR is associated with (aimed at acting on) one or more Developable Entities.
interacts with other	The Capability of a concrete object interacts with other concrete entities.
is surrounded by	A Developable Entity (as target entity) may be surrounded by Context Entities.
requires to be satisfied by	A FR can be satisfied by none or more constraints (i.e., NFRs).
subDevelopableEntity	A Developable Entity may be composed of none or several sub-developable entities, which are in turn Developable Entities.

Also, goals can be classified into Business and Information Need Goals. Business goals are the major or primary goals that an organization intends to achieve by setting them at different Organizational Levels. In turn, a business goal can have business subgoals. On the other hand, information need goals are intended to support business goals. Commonly, they provide useful information in order to know the degree of achievement of business goals. An information need goal can also require ME Information Need goals (ME stands for measurement and evaluation). The latter is a more specific type of information need goal, which is driven by NFRs definition and ME activities.

Note that a ME information need describes the target object type to be evaluated (Entity Category) indicating also an Evaluation Focus, which can be for instance the Quality Focus, Cost Focus, among others. A key concept is the Non-Functional Requirement term, which is specialized by means of Characteristics and Attributes to be evaluated on how or how well an Evaluable Entity performs or shall perform. From other perspective, a Functional Requirement should require to be satisfied by constraints or 'ilities', which can be specified by a non-functional requirement. Particularly, NFRs can be represented by a NFRs Model.

The association between an Entity Category –for example service, product, or system– and a Quality Focus –e.g., service quality, internal quality and external quality respectively- produces a Quality View (i.e. a specialization of NFRs View). Moreover, a Quality View can be influenced by another Quality View. For example, the product quality view (i.e., the independent quality view) influences the system quality view (i.e. the dependent quality view) [10, 13].

In a ME Project, in order to evaluate NFRs, metrics and indicators should be designed. A Metric provides a Measurement specification of how to quantify a particular attribute of an Evaluable Entity, using a particular procedure, and how to represent its values, using a particular Scale. Two types of metrics are distinguished. Direct Metrics are those for which values are obtained directly from measuring the corresponding entity's attribute, by using a Measurement Procedure. On the other hand, Indirect Metrics' values are calculated from others metrics' values following a function specification and a particular Calculation Procedure. A Measurement specifies the task by using a particular metric description in order to produce a Measure value. Other associated metadata is the responsible name and the timestamp in which the measurement was performed.

An Indicator allows specifying how to calculate and interpret the attributes and characteristics of a NFRs model. Two types of indicators are distinguished. First, Elementary Indicators that evaluate lower-level requirements, namely, attributes. Each elementary indicator has an Elementary Model that provides a mapping function from the metric's measures (the domain) to the indicator's scale (the range). The new Scale is interpreted using a set of Decision Criterion, which help analyze the level of satisfaction reached by each attribute. Second, Derived Indicators, which evaluate mid-level and higher-level requirements, i.e. sub-characteristics and characteristics in a NFRs model. Different aggregation models can be used to perform evaluations. The global derived indicator's value ultimately represents the global degree of satisfaction in meeting the stated information need for a given purpose and user viewpoint. An Evaluation represents a task which involves a single calculation, following a particular indicator specification –either elementary or global-, producing an Indicator Value.

Note that some terms in the measurement and evaluation components in Fig. 3 are enriched with stereotyped terms from the Process Ontology [2]. For example, the measurement and evaluation terms have both the semantic of Task. While the metric and indicator terms have both the semantic of Method, and the measure and indicator value terms have the meaning of Outcome. The reasons of using stereotypes to enrich semantically terms are exposed in [2].

Coming back to the Business Goal term, we have recently added to it the purpose category attribute. We have identified four categories of purposes, namely: "Evaluation", "Testing", "Development", and "Maintenance/Change". For the Evaluation Category, we have discussed in [1] three specific subcategories which include specific purposes and a family of evaluation Strategies. A business goal dealing with some specific evaluation purpose such as to 'understand', or 'monitor' may be achieved by strategies embracing NFRs definition, measurement, evaluation and analysis activities. But other specific evaluation purposes such as to 'improve' or 'monitor and control' usually imply changing the Developable Entity and therefore embrace FRs definition activities as well. Likewise, specific Development and Maintenance/Change purposes such as 'create', 'modify', 'delete' or 'add' always involve FRs. As a consequence, a given business goal is derived in one or more FRs.

Looking at the FRs component in Fig. 3, we can say that a particular Functional Requirement is intended to act upon a Developable Entity. It is the concrete object to be developed, tested or modified considering its Features and Capabilities. A

Developable Entity has both Features and Capabilities. The former has semantic of property of a Developable Entity, which refers to the intrinsic constitution, structure, or parts of the Developable Entity. While the latter has semantic of power of an Entity, which refers to what it is supposed to do and behave. Analyzing the discussion raised in the article named "*The Ontology of Things, Properties and Powers*" [15], we adopt its three terms. So, Developable Entity in our proposed ontology has semantic of Thing, Feature has semantic of Property, and Capability has semantic of Power. The author in [15] argues that things (concrete Entities), properties (Features) and powers (Capabilities) do not exist in spatiotemporal isolation from one another, so they all emerge simultaneously to form a unity.

Note that in Fig. 3, for instance, the «Thing» stereotype enriches semantically to three terms, namely: Developable Entity (in the FRs component), Context Entity (in the Context component), and Evaluable Entity (in the NFRs component). On the other hand, it is also worth mentioning that the terms, attributes and relationships included in the FRs component are the minimum and necessary ones for describing functional requirements. Specific conceptual domain components for requirements development, architectural design, among others will be included in the development package. Tables 2 and 3 include the attribute and relationship definitions considered for the FRs subontology,

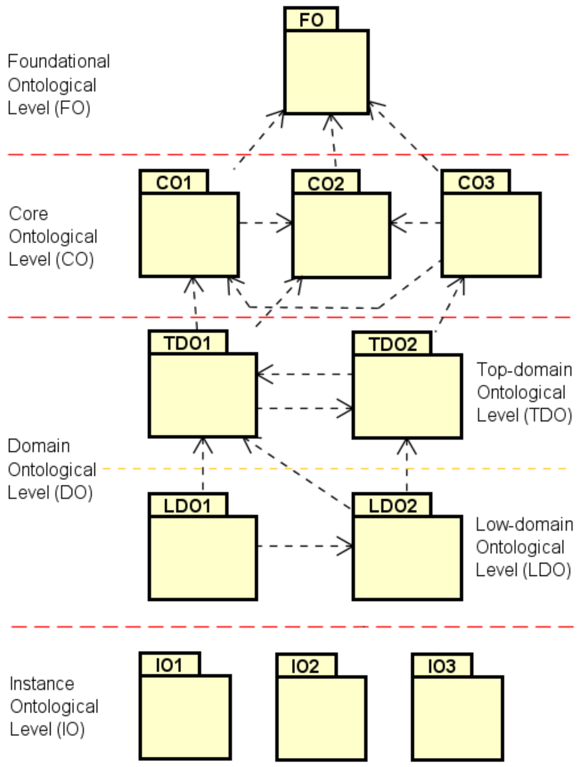

FO
Foundational Ontological Level (FO)
CO1
CO2
CO3
Core Ontological Level (CO)
TDO1
TDO2
Top-domain Ontological Level (TDO)
Domain Ontological Level (DO)
LDO1
LDO2
Low-domain Ontological Level (LDO)
IO1
IO2
IO3
Instance Ontological Level (IO)

Fig. 4 Four-layered architecture which considers Foundational, Core, Domain and Instance ontological levels.

respectively.

Lastly, aimed at having the general description of components, we can say that an Organization arranges work by means of Projects, which allow operationalizing the established organizational Goals. There exist different types of projects such as development, maintenance, testing, among others. For example, a Development Project operationalizes a business goal that has a development purpose category, such as for example to create a new software product or system. A ME Project operationalizes a business goal that has an evaluation purpose category, which is always supported by ME information need goals. Additionally, a MEC Project (MEC stands for measurement, evaluation and change) operationalizes both a business goal and its related ME information need goals with the purpose of improvement. Fig. 3 shows that a MEC Project is composed by a ME subproject and by a change subproject in which changes are driven by measurement and evaluation. A Change Project operationalizes a business goal with the purpose of changing or improving the current state of a concrete entity.

In the end, a project uses a Strategy as a resource, which helps to achieve a business goal for any purpose category. Recall that a well-specified strategy should contain: Process specification (i.e. what should be done), Method specification (how tasks should be performed), and a Domain Conceptual Base that formally states the domain concepts, properties and relationships to strengthen the process and method specifications.

It is worth mentioning that in follow-on manuscripts we will thoroughly discuss the previous components in the framework of an ontological conceptual architecture. This layered architecture considers Foundational, Core, Domain and Instance ontological levels as depicted in Fig. 4. In turn, the domain level is split down in two sub-levels: Top-domain and Low-domain levels.

For example, the Thing and Thing Category terms are concepts of the thing ontology placed at the foundational level. On the other side, context, business goal, and project components (shown in Fig. 1, among others such as process, resource, work product, situation, etc. that are not shown in the figure) are placed at the core level. Furthermore, FRs, NFRs, measurement, evaluation, testing, development and maintenance components are placed at the domain level. At instance level ontologies for quality characteristics, units, scales, among others can be allotted.

Note that Fig. 4 also conveys that ontologies at the same architectural level can be related each other. Moreover, ontologies at lower levels can be semantically enriched by ontologies at upper levels.

For example, the reader can see in Fig. 3 that concepts of measurement and evaluation components at domain level are enriched by concepts of the process ontology at core level.

4. Instantiating NFRs and FRs concepts for the JGUIAR case

In this Section, we reassume the example introduced in Section 2. So, we exemplify the improvement purpose which implies both evaluation and change activities. Also, this requires instantiating both NFRs and FRs concepts in addition to business and information need goals concepts.

In our example, "Informatics Department, Engineering School at UNLPam" is the organization that establishes the following business goal at operational level: "Improve 15 p.p. the current JGUIAr app's external quality in 3 months". In order to give supporting information to this evaluation business goal an information need goal should be established such as "Analyze if external quality has improved 15 p.p. after changes" across the 3-month time frame.

That information need goal will allow to understand the extent to which the established business goal has been achieved after making measurement, evaluation and change activities. To our case, the business goal is operationalized by a MEC project using the GOCAMEC strategy.

Fig. 5 shows the generic A1-A6 activities that GOCAMEC specifies for any quality view. We will instantiate and illustrate GOCAMEC in Section 5 for the System Quality View. As indicated in Section 2, it supports the following three aspects: i) to understand the current quality state for the evaluable entity and recommend changes, mainly for the weakly benchmarked attributes; ii) to make changes on the developable entity or its subentities; and iii) to understand the ulterior quality state (the improvement) after performed changes.

Fig. 6 (at the bottom side) shows that for the established business/information need goals, two ME information need goals are instantiated in conjunction with one business subgoal, labelled accordingly such as:

(1) "Understand the JGUIAr external quality weaknesses";

(2) "Apply changes on the current JGUIAr version (v1.3) in 1 month";

(3) "Understand the JGUIAr external quality after changes (i.e., the new v1.4)".

So if we analyze these aspects in Fig. 6, we see that (1) and (3) are ME information need goals, which are operationalized by ME projects. Therefore, the GOCAMEC's activities and methods require to instantiate NFRs concepts. Conversely, (2) is a business subgoal, which is operationalized by a change project. Thus, the GOCAMEC's activities and methods require to instantiate FRs concepts as well.

Firstly, to illustrate (1), Fig. 7a instantiates terms

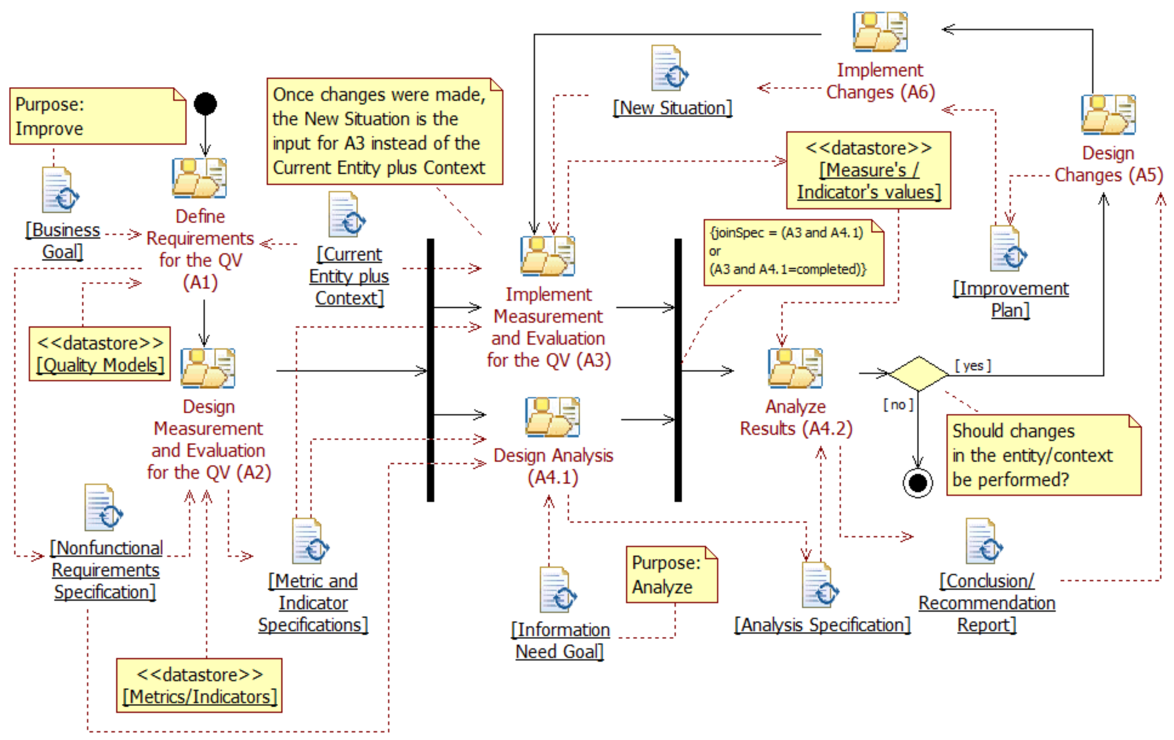
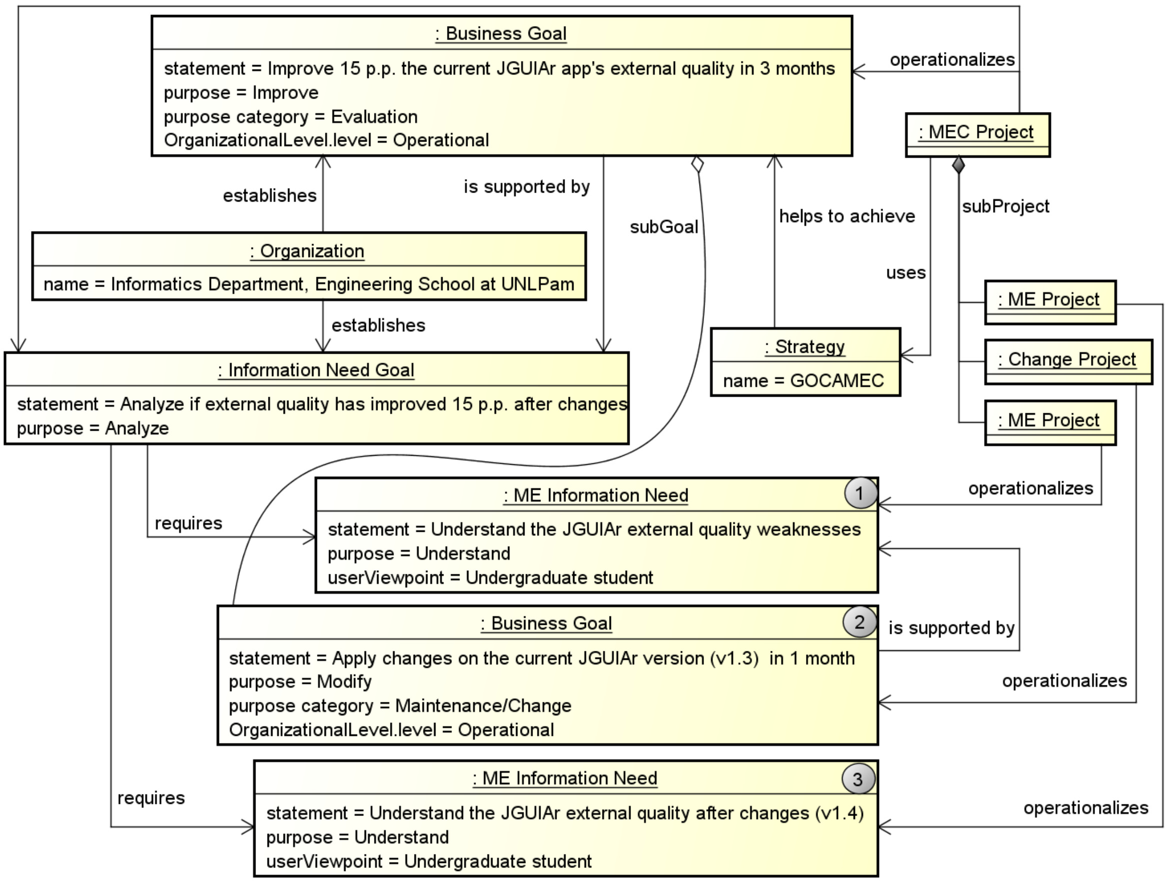

Purpose: Improve
[Business Goal]
Define Requirements for the QV (A1)
<<datastore>> [Quality Models]
Design Measurement and Evaluation for the QV (A2)
[Nonfunctional Requirements Specification]
<<datastore>> [Metrics/Indicators]
[Metric and Indicator Specifications]
Once changes were made, the New Situation is the input for A3 instead of the Current Entity plus Context
[Current Entity plus Context]
[New Situation]
Implement Changes (A6)
<<datastore>> [Measure's / Indicator's values]
Implement Measurement and Evaluation for the QV (A3)
Design Analysis (A4.1)
[Information Need Goal]
Purpose: Analyze
{joinSpec = (A3 and A4.1) or (A3 and A4.1=completed)}
Analyze Results (A4.2)
[Analysis Specification]
[yes]
[no]
Should changes in the entity/context be performed?
Design Changes (A5)
[Improvement Plan]
[Conclusion/ Recommendation Report]

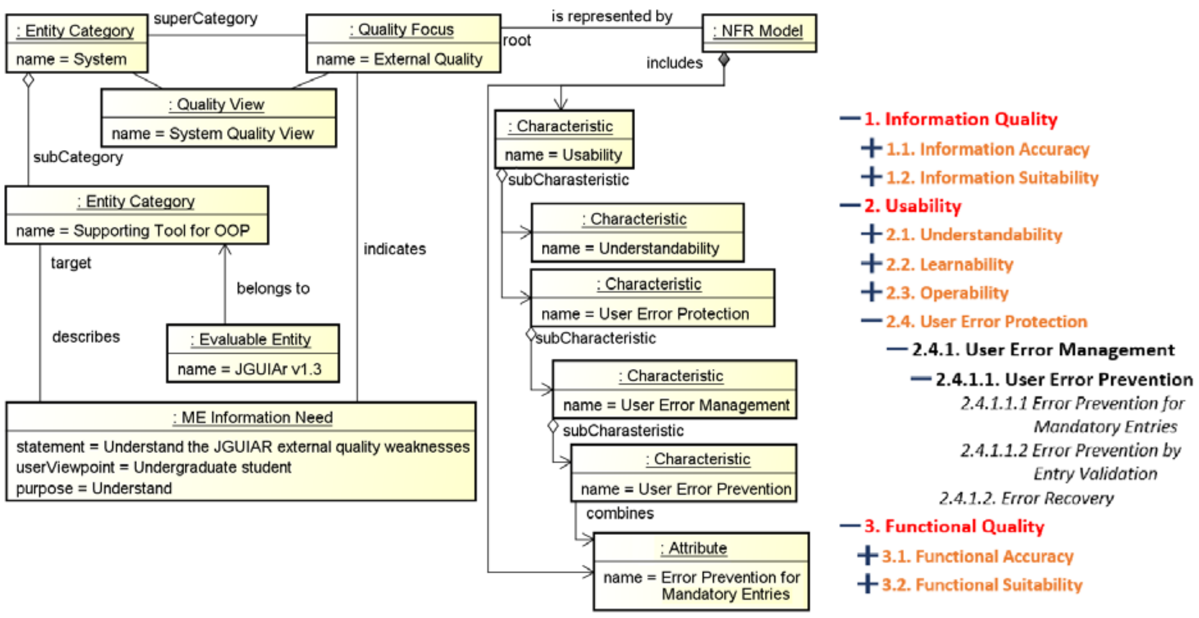
Fig. 5 Generic process for the GOAMEC strategy: Functional and behavioral perspectives. Note: QV stands for Quality View.

of the NFRs component (recall Fig. 3), which includes also the NFRs view component. For the "Understand the JGUIAr external quality weaknesses" goal, the quality focus is "External Quality". This focus is represented by a NFRs (quality) model. To build this model we have



: Business Goal
statement = Improve 15 p.p. the current JGUIAr app's external quality in 3 months
purpose = Improve
purpose category = Evaluation
OrganizationalLevel.level = Operational
operationalizes
: MEC Project
establishes
is supported by
subGoal
helps to achieve
subProject
: Organization
name = Informatics Department, Engineering School at UNLPam
uses
: ME Project
establishes
: Strategy
name = GOCAMEC
: Change Project
: Information Need Goal
statement = Analyze if external quality has improved 15 p.p. after changes
purpose = Analyze
: ME Project
operationalizes
: ME Information Need
1
requires
statement = Understand the JGUIAr external quality weaknesses
purpose = Understand
userViewpoint = Undergraduate student
: Business Goal
2
is supported by
statement = Apply changes on the current JGUIAr version (v1.3) in 1 month
purpose = Modify
purpose category = Maintenance/Change
OrganizationalLevel.level = Operational
operationalizes
: ME Information Need
3
requires
statement = Understand the JGUIAr external quality after changes (v1.4)
purpose = Understand
userViewpoint = Undergraduate student
operationalizes

Fig. 6 Scenario instantiation where a Business Goal from the operational level is supported by an Information Need Goal. Taking into account the GOCAMEC strategy, these goals are decomposed in two ME Information Need Goals and in one Business subGoal. Note that ME stands for Measurement and Evaluation; MEC for Measurement, Evaluation and Change; and p.p. for percentage points.



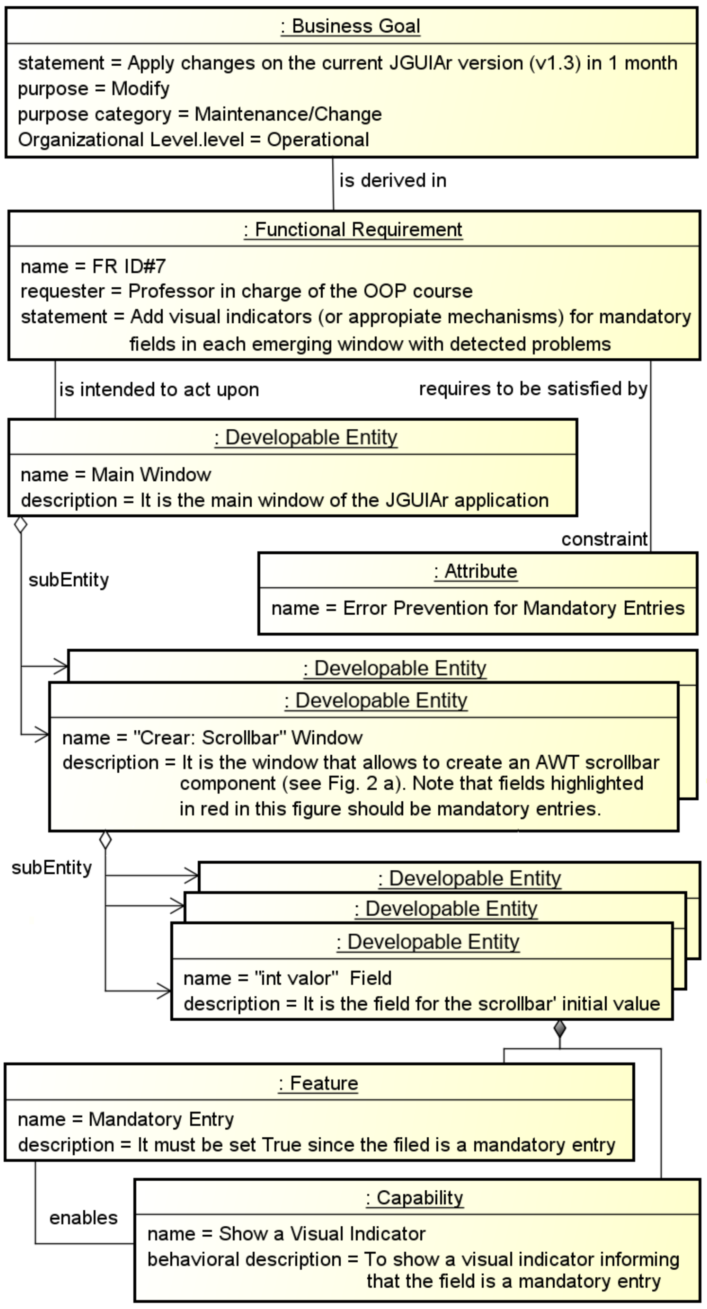
(a) (b)

Fig. 7 a) Instantiation of terms for the *NFRs* and *NFRs View* conceptual components considering the *ME Information Need Goal* labeled (1) in Fig. 6. b) Excerpt of the NFRs tree used to evaluate the JGUIAr application's External Quality.

considered the 2Q2U (internal/external Quality, Quality in use, actual Usability, and User experience) model [16]. The resulting NFRs model includes the "Information Quality", "Usability" and "Functional Quality" characteristics. Note that for the illustration aim just Usability is represented in Fig. 7a. In turn, "Usability" has four sub-characteristics (see Fig. 7b). For instance, the "User Error Prevention" sub-characteristic combines attributes such as "Error Prevention for Mandatory Entries". (Note that Fig. 7 just includes some sub-characteristics and attributes in order not to clutter the diagram).

The definition of "Error Prevention for Mandatory Entries" attribute is "degree to which mechanisms to prevent mistakes or incidents in mandatory entries are provided". The quantification of each attribute is made by means of metrics and their interpretation by means of indicators (terms from the measurement and evaluation components respectively).

Additionally, the (1) ME information need specified in Fig. 7a describes also the "JGUIAr v1.3" evaluable entity which belongs to the "Supporting Tool for OOP" entity category. In turn, its super category pertains to "System". It is important to remark that the association between the quality focus and the entity super category determines the quality view, which in our case is the "System Quality View".



: Business Goal
statement = Apply changes on the current JGUIAr version (v1.3) in 1 month
purpose = Modify
purpose category = Maintenance/Change
Organizational Level.level = Operational
is derived in
: Functional Requirement
name = FR ID#7
requester = Professor in charge of the OOP course
statement = Add visual indicators (or appropiate mechanisms) for mandatory fields in each emerging window with detected problems
is intended to act upon
requires to be satisfied by
: Developable Entity
name = Main Window
description = It is the main window of the JGUIAr application
constraint
: Attribute
name = Error Prevention for Mandatory Entries
subEntity
: Developable Entity
: Developable Entity
name = "Crear: Scrollbar" Window
description = It is the window that allows to create an AWT scrollbar component (see Fig. 2 a). Note that fields highlighted in red in this figure should be mandatory entries.
subEntity
: Developable Entity
: Developable Entity
: Developable Entity
name = "int valor" Field
description = It is the field for the scrollbar' initial value
: Feature
name = Mandatory Entry
description = It must be set True since the filed is a mandatory entry
enables
: Capability
name = Show a Visual Indicator
behavioral description = To show a visual indicator informing that the field is a mandatory entry

Fig. 8 Instantiation of terms for the FRs conceptual component considering the Business subGoal labeled (2) in Fig. 6.

Going a step forward, in order to illustrate the above (2) business subgoal for the change purpose category, Fig. 8 instantiates the terms of the *FRs* component. For the statement "*Apply changes on the current JGUIAr version (v1.3) in 1 month*", one or many specific FRs are derived. In principle, it may be derived as many FRs as recommendations for change actions were issued in the (A4.2) analysis activity.

Note that Fig. 8 instantiates just one FR, namely: "*Add visual indicators (or appropriate mechanisms) for mandatory fields in each emerging window with detected problems"*. This FR is intended to act upon the "*Main Window*" entity and its sub-entities, as for example "*'Crear: Scrollbar' Window*", which in turn has several mandatory fields, such as "*'int valor' Field*". Recall that Fig. 2a shows the "*'Crear: Scrollbar' Window*" with the detected problem. That is, this dialog window has several mandatory fields without error prevention mechanisms. Therefore, each mandatory field in the dialog must have its "*Mandatory Entry*" feature set to true, which enable (as capability) "*to show a visual indicator informing that the field is a mandatory entry*".

5. Use of the GOCAMEC Strategy for Improving the JGUIAr Application

In this Section, we illustrate the practical case introduced in Sections 2 and 4, in which the GOCAMEC strategy is applied for achieving the aforementioned business goal: "Improve 15 p.p. the current JGUIAr app's external quality in 3 months". This goal contains the purpose of improving, so for its realization the GOCAMEC process includes the following three aspects:

(1) "Understand the JGUIAr external quality weaknesses";

(2) "Apply changes on the current JGUIAr version (v1.3)"; and

(3) "Understand the JGUIAr external quality after changes (i.e., the new v1.4)".

For achieving (1), the GOCAMEC activities – shown in Fig. 5- to be carried out are: (A1) Define NFRs; (A2) Design Measurement and Evaluation; (A3) Implement Measurement and Evaluation; (A4.1) Design Analysis; and (A4.2) Analyze

<u>INDIRECT METRIC</u>

Name: Proportion of Error Prevention for Mandatory Entries (%PEPME)

Objective: Calculate the percentage of error prevention for mandatory entries, for all given screens, windows or dialogs.

Author: Peppino – Tebes **Version:** 1.0

Calculation Procedure:

Formula:

$$\%PEPME = \frac{100}{m}\left(\sum_{j=1}^{m}\left(\frac{\sum_{i=1}^{n_j}\mathrm{SLME}_i}{2n_j}\right)\right)$$

Where m is the number of inspected screens, windows or dialogs (with m>0); n_j is the number of mandatory fields (with $n_j>0$)

Numerical Scale:

Representation: Continuous **Value Type:** Real **Scale Type**: Ratio

Unit Name: Percentage **Acronym**: %

<u>Related Direct Metric</u>

Name: Support Level for Mandatory Entries (SLME)

Objective: Determine the support level (user errors prevention) for mandatory entries in a given field.

Author: Peppino – Tebes **Version:** 1.0

Measurement Procedure:

Type: Objective

Specification: The expert inspects the screen (or window, or dialog) where is the given field to determine the rating (0, 1 or 2). Where 0 means that the mandatory data field has not any kind of visual indicator w.r.t. its mandatory nature; 1 means that the mandatory data field has some kind of indicator w.r.t. its mandatory nature, but it is not explicitly informed; and 2 means that the mandatory data field has a clear visual indicator of its mandatory nature and it is explicitly informed.

Numerical Scale:

Representation: Discrete **Value Type:** Integer **Scale Type**: Absolute

Unit Name: Prevention Level **Acronym**: PL

Fig. 9 Metric specification for the "Error Prevention for Mandatory Entries" (EPME) attribute.

Results. In order to help achieving the (2) subgoal, the GOCAMEC process includes two activities, which are Design and Implement Changes (A5 and A6 in Fig. 5, respectively). Finally, to reach (3), the (A3) Implement Measurement and Evaluation, and (A4) Analyze Results activities are performed again.

Since the process in Fig. 5 is generic for any quality view, we instantiate those activities considering the specific quality view. For example, A1 is now renamed "Define non-Functional Requirements for the System Quality View", A2 is renamed as "Design Measurement and Evaluation for the System Quality View", and so forth for the remainder activities. In the next paragraphs, we describe the instantiated activities and methods of GOCAMEC for improving the JGUIAr application. The reader can be aware of the added value of reusing the terms, attributes and relationships described in Section 3 throughout the instantiation of GOCAMEC activities. Therefore, we can affirm that well-defined vocabularies promote robust process and method specifications as we highlight later on.

(A1) Define non-Functional Requirements for the System Quality View: Looking at the "Improve 15 p.p. the current JGUIAr app's external quality in 3 months" business goal statement, we can determine that the evaluation focus is the External Quality. For this focus, we used the 2Q2U model as a NFRs model, which extends the quality model of the ISO 25010 standard [13]. The first column of Table 4 contains the defined NFRs tree, where the root characteristic (or evaluation focus) is the External Quality, and the associated sub-characteristics to be evaluated are the Usability, Information Quality and Functional Quality. In addition, we established 22 attributes, which are combined to these sub-characteristics. In Section 4 we defined the "Error prevention for mandatory entries" (EPME) attribute, which is used to exemplify the remaining activities. It is important to remark that the A1 activity is grounded on the subontology defined for the NFRs component in Fig. 3. Also, in the A1 activity, concepts related with the context component are considered as well. For instance, as context properties and their values, we can mention that the users of JGUIAr are OOP undergraduate students, who have few or none experience in GUI development with Java.

(A2) Design Measurement and Evaluation for the System Quality View: Once the NFRs are defined, metrics and indicators from a repository (see Metrics/Indicators <<datastore>> in Fig. 5) should be selected. Specifically, one metric and one elementary indicator per each attribute, and one derived indicator per each characteristic of the NFRs tree should be chosen. Fig. 9 represents the indirect metric and its related direct metric specifications for the EPME attribute. Likewise, Fig. 10 represents the elementary indicator specification for the EPME attribute.

On the other hand, in this study all the indicators have a ratio scale type. Also they use three acceptability levels, namely: "Unsatisfactory" (●) whose values ranges between [0-80); "Marginal"

ELEMENTARY INDICATOR

Name: Performance Level of the Support for Mandatory Entries (PL_SME)

Author: Peppino – Tebes **Version:** 1.0

Elementary Model:

Specification: the mapping is: PL_SME = %PEPME.

Decision criteria [three Acceptability levels]**:**

Decision criterion 1: if $0 \leq$ PL_SME < 80

Name Unsatisfactory (●)

Description: Indicates change actions must be taken.

lower threshold: 0 **upper threshold**: 80

Decision criterion 2: if $80 \leq$ PL_SME < 90

Name: Marginal (▲)

Description: Indicates a need for improvement actions.

lower threshold: 80 **upper threshold**: 90

Decision criterion 3: if $90 \leq$ PL_SME ≤ 100

Name: Satisfactory (■)

Description: Indicates no need for change actions.

lower threshold: 90 **upper threshold**: 100

Numerical Scale:

Representation: Continuous **Value Type:** Real **Scale Type**: Ratio

Unit Name: Percentage **Acronym**: %

Fig. 10 Indicator specification for the "Error Prevention for Mandatory Entries" (EPME) attribute.

(▲) between [80-90); and "Satisfactory" (■) between [90-100]. For instance, Unsatisfactory level means that change actions must be taken usually with high priority. While Marginal level indicates a need for improvement actions.Note that metric and indicator templates represent method specifications, which follow the terminology defined in the measurement and evaluation components depicted in Fig. 3. For example, the Elementary Indicator template used in Fig. 10 defines the name, author

Table 4 NFRs tree (1st column). Indicator's Values (in [%]) of the JGUIAr's evaluation before changes (2nd column) and after changes (3rd column). Gain (in [p.p.]) after changes (5th column). The symbol "●" means unsatisfactory; "▲" marginal; "■" satisfactory; and "♦" in the 4th column that we performed changes to the attribute.

Characteristics (in bold) and *Attributes (in italic)*	**Indicator´s Value**		**Changed**	**Gain**
	JGUIAr v.1.3	**JGUIAr v.1.4**		
- 1. External Quality	**68.87** ●	**94.14** ■		**25.27**
- 1.1. Information Quality	**81.90** ▲	**100** ■		**18.10**
- 1.1.1. Information Accuracy	**100** ■	**100** ■		**0**
- 1.1.1.1. Credibility	**100** ■	**100** ■		**0**
1.1.1.1.1. Objectivity	*100* ■	*100* ■		*0*
1.1.1.1.2. Verifiability	*100* ■	*100* ■		*0*
- 1.1.2. Information Suitability	**77.60** ●	**100** ■		**22.40**
- 1.1.2.1. Added Value	**100** ■	**100** ■		**0**
1.1.2.1.1. Beneficialness	*100* ■	*100* ■		
- 1.1.2.2. Coverage	**98.20** ■	**100** ■		**1.8**
1.1.2.2.1. Completeness	*96.43* ■	*100* ■	♦	*3.57*
1.1.2.2.2. Conciseness	*100* ■	*100* ■		*0*
- 1.1.2.3. Consistency	**50** ●	**100** ■		**50**
1.1.2.3.1. Consistency of established language use	*50* ●	*100* ■	♦	*50*
- 1.2. Usability	**71.87** ●	**94.64** ■		**22.77**
- 1.2.1. Understandability	**70.55** ●	**80** ▲		**9.45**
- 1.2.1.1. Familiarity	**70.55** ●	**80** ▲		**9.45**
- 1.2.1.1.1. Button Icon Ease to be Recognized	**88.18** ▲	**100** ■		**11.82**
1.2.1.1.1.1. Main button icon ease to be recognized	*87.50* ▲	*100* ■	♦	*12.50*
1.2.1.1.1.2. Contextual button icon ease to be recognized	*88.88* ▲	*100* ■	♦	*11.12*
1.2.1.1.2. Foreign language support	*0* ●	*0* ●		*0*
- 1.2.2. Learnability	**78.54** ●	**100** ■		**21.46**
- 1.2.2.1. Feedback Suitability	**66.67** ●	**100** ■		**33.33**
1.2.2.1.1. Dialog message appropriateness	*66.67* ●	*100* ■	♦	*33.33*
- 1.2.2.2. Helpfulness	**83.85** ▲	**100** ■		**16.15**
1.2.2.2.1. Context-sensitive help appropriateness	*95.83* ■	*100* ■	♦	*4.17*
1.2.2.2.2. Main Help appropriateness	*100* ■	*100* ■		*0*
1.2.2.2.3. Controls emerging message appropriateness	*6.25* ●	*100* ■	♦	*93.75*
- 1.2.3. Operability	**77.16** ●	**100** ■		**22.84**
- 1.2.3.1. Data Entry Ease	**90.47** ■	**100** ■		**9.53**
1.2.3.1.1. Defaults	*90.47* ■	*100* ■	♦	*9.53*
- 1.2.3.2 Consistency	**68.75** ●	**100** ■		**31.25**
1.2.3.2.1. Icon-action uniqueness	*68.75* ●	*100* ■	♦	*31.25*
- 1.2.4. User Error Protection	**62.24** ●	**100** ■		**37.76**
- 1.2.4.1. User Error Management	**62.24** ●	**100** ■		**37.76**
- 1.2.4.1.1. User Error Prevention	**45.86** ●	**100** ■		**54.14**
1.2.4.1.1.1. Error prevention for mandatory entries	*0* ●	*100* ■	♦	*100*
1.2.4.1.1.2. Error prevention by entry validation	*91.72* ■	*100* ■	♦	*8.28*
1.2.4.1.2. Error recovery	*85.71* ▲	*100* ■	♦	*14.29*
- 1.3. Functional Quality	**58.25** ●	**89.78** ▲		**31.53**
- 1.3.1. Functional Accuracy	**39.24** ●	**100** ■		**60.76**
- 1.3.1.1. Correctness	**39.24** ●	**100** ■		**60.76**
- 1.3.1.1.1. Consistency	**39.24** ●	**100** ■		**60.76**
1.3.1.1.1.1. Consistency in master-slave panels output	*96.87* ■	*100* ■	♦	*3.13*
1.3.1.1.1.2. Functional consistency due to emerging windows	*12.82* ●	*100* ■	♦	*87.18*
- 1.3.2. Functional Suitability	**80** ▲	**80** ▲		**0**
-1.3.2.1. Added Value	**100** ■	**100** ■		**0**
1.3.2.1.1. Synchronization (Integratedness) of Panes	*100* ■	*100* ■		*0*
-1.3.2.2. Coverage	**50** ●	**50** ●		**0**
1.3.2.2.1. Completeness	*50* ●	*50* ●		*0*

and version, which are shown as attributes in the Indicator term of Fig. 3. In addition, in Fig. 10, an Elementary Model and three Decision Criteria are specified, which adhere to the evaluation component´s terminology. For instance, the Elementary Indicator term has an attribute named elementary model and also associates two or more Decision Criteria as rules. Finally, the elementary indicator specification defines a Numerical Scale and the percentage Unit. This adheres to Fig. 3's terminology where the Indicator term is associated with the Scale term. Scale in turn is associated with the Unit term. Consequently, the template is fully compliant to the terminology shown in the evaluation component.

Once A2 is finished, all metrics and indicators to be used in the MEC Project were registered in the "Metrics and Indicators Specifications" artifact (as shown in Fig. 5).

(A3) Implement Measurement and Evaluation for the System Quality View: Following the specified measurement or calculation procedure for each metric, the measurements of all attributes were performed. After measuring, we carry out the elementary evaluation following the specification of the elementary model associated with each indicator. For example, the elementary indicator value for the EPME attribute gave 0%, representing thus a weakness (recall Fig. 2a, which shows a dialog window where no error prevention mechanism for mandatory entries is provided). Finally, the derived indicators values for all characteristics were yielded. Particularly, we used the LSP (*Logic Scoring of Preference*) [17] aggregation model, which calculates the indicator value of each characteristic according to the assigned logic operator and weights (importance) of its associated sub-characteristics and/or attributes. The JGUIAr v. 1.3 evaluation results (measure's/indicator's values) and metadata were stored in a repository (see Measure's/Indicator's values <<datastore>> in Fig. 5). The second column of Table 4 shows the indicator's values for the JGUIAr v. 1.3.

(A4.1) Design Analysis: Concurrently with A3 activity, the analysis design can be performed. This activity produces the "*Analysis Specification*" work product (see Fig. 5) in which scales and scale types for datasets must be taken into account. Additionally, this document contains the suitable mathematical and statistical methods and techniques to be used in A4.2, regarding the properties of data, in addition to how the data and information will be displayed. Ultimately, the analysis should be designed considering the specific evaluation purpose to be achieved in a given context.

For example, for the improvement purpose, the analysis should concentrate on weaknesses, i.e., on attributes that benchmarked with low performance, while for the compare and adopt purpose, the analysis should focus on strengths, i.e., on attributes that benchmarked with high performance.

The *Importance-Performance Analysis* (IPA) is a useful analysis technique that may help analysts to gauge whether to focus on strengths or weaknesses taking into account the evaluation purpose. The IPA was proposed in [18] as a means to measure customer satisfaction regarding a product or service. This technique considers both the importance of a product or service to a user and the performance of a business providing that product or service. Therefore, by using IPA we examine not only the performance of the attributes of a target entity, but also the importance of these attributes as a determining factor in user satisfaction. For this end, four quadrants are considered as shown in Fig. 11:

- (A) *Concentrate here* (high importance, low performance): require immediate attention for improvement and are major weaknesses.
- (B) *Keep up with the good work* (high importance, high performance): indicate opportunities for achieving or maintaining competitive advantage and are major strengths.
- (C) *Low priority* (low importance, low performance): are minor weaknesses and do not require additional effort.
- (D) *Possible overkill* (low importance, high performance): indicate that business resources committed to these attributes would be overkill and should be deployed elsewhere.

For the GOCAMEC strategy, analysts should concentrate mainly on attributes that fall in the (A) quadrant, and to a lesser extent in the (C) quadrant. Note that in Fig. 11 the performance axis is represented in our case by elementary indicators as performance variables. And the importance is represented by relative weights.

We assume for this study that indicators values falling within the unsatisfactory (●) and marginal (▲) ranges are in the (A) and (C) quadrants. Consequently, these attributes should be recommended to be improved. On the other hand, for attributes falling within the satisfactory (■) range -(B) and (D) quadrants-, and with their indicators

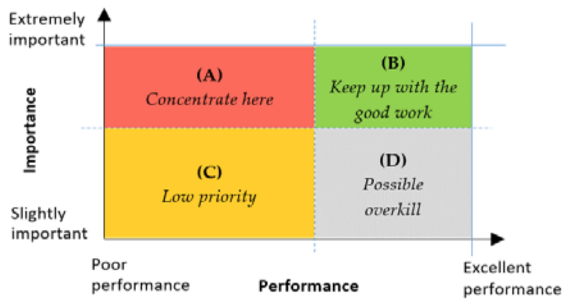


Fig. 11 The four quadrants for the Importance-Performance Analysis.

achieving less than 100%, they could also be considered for improvement, but obviously with very low priority.

(A4.2) Analyze Results: Following the guidelines that we established in the "*Analysis Specification*" document, we looked at the indicators values of JGUIAr v. 1.3 that were obtained in the A3 activity. Hence, we recommended improvements for most of the attributes. In the fourth column of Table 4 we highlighted with the "◆" symbol those attributes on which we recommended to apply changes in order to improve their performance level. For the EPME attribute, the indicator value was 0% so it represents a weakness. Therefore, the recommendation was "*to ensure that in the set of selected dialog windows, the mandatory entries have some mechanism for error prevention*" (see R11 recommendation in Fig. 12). Note that we did not consider the R5 recommendation for the "*Foreign language support*" attribute (which ranked 0%) since at this moment the JGUIAr app is devoted to undergraduate students in our School where courses are only in Spanish. However, we are planning to extend it to English as well.

(A5) Design Changes: Taking as input the "*Recommendation Report*" generated in the A4.2 activity, the changes to be performed were designed. At this point, it is important to remark that GOCAMEC is a strategy in which changes are driven by measurement and evaluation. This means that, for instance, a good metric specification can help taking better decisions in planning and performing change actions.

For example, the "Proportion of Error Prevention for Mandatory Entries" metric specification (recall Fig. 9) allows to record, at measurement time, the availability (or not) of error prevention mechanisms for all mandatory fields per each evaluated window. Therefore, at change time, developers can know each field ID# that has not error prevention mechanism per each window ID#. For improving the acceptability level of the EPME attribute (0%), we proposed adding the commonly used "(*)" symbol to visualize users those fields that are mandatories. As a result, all proposed changes were registered in the "*Improvement Plan*" document. Fig. 13 shows an excerpt of the Improvement Plan for the JGUIAr v.1.3 application.

At this point, it is worth mentioning that for each elementary NFR (attribute) to be improved, we derived and defined a related FR. For example, for the CA10 change action (Fig. 13) related to the R11 recommendation, we defined the ID#7 FR (as depicted in Fig. 8) that states: "*Add visual indicators (or appropriate mechanisms) for mandatory fields in each emerging window with detected problems*". This FR requires to be satisfied by the EPME attribute (as NFR) and is intended to act upon the

ID	Recommendation (R#)	Implied Attribute	Priority
	⋮		
R2	*To ensure that all information given by the application is in Spanish.*	*Consistency of established language use* (1.1.2.3.1.)	L
	⋮		
R5	*Add English as a foreign language.*	*Foreign language support* (1.2.1.1.2.)	VL
	⋮		
R9	*To ensure that in the set of selected dialog windows, the mandatory entries have default values.*	*Defaults* (1.2.3.1.1.)	H
	⋮		
R11	*To ensure that in the set of selected dialog windows, the mandatory entries have some mechanism for error prevention.*	*Error prevention for mandatory entries* (1.2.4.1.1.1.)	VH
	⋮		

Fig. 12 Excerpt of the Recommendation Report. Note that priorities go from very high (VH) to very low (VL) priority.

ID	Change Action (CA#)	Source of the CA	Method
	⋮		
CA2	*Translate information blocks that are not in Spanish.*	R2	Manual translation
	⋮		
CA7	*Add default values in mandatory fields*	R9	Reprograming
	⋮		
CA10	*Add visual indicators (or appropriate mechanism) for mandatory fields*	R11	Refactoring
	⋮		

Fig. 13 Excerpt of the Improvement Plan.

Main Window entity. In turn, the "*Crear: Scrollbar*" Window is a sub-entity of the Main Window. In addition, the "*int valor*" field is a sub-entity of the "*Crear: Scrollbar*" Window. The last entity (or sub-entity) has the "*Mandatory Entry"* feature, which enables the "*Show a Visual Indicator*" capability.

(A6) Implement Changes: In this activity, the changes proposed in the *"Improvement Plan*" document were performed. For instance, considering the CA10 change action (Fig. 13), in the "*int valor*" field we added the "(*)" visual indicator (due to it is a mandatory field) as well as a legend in the bottom of the "*Crear: Scrollbar*" Window indicating the meaning of this symbol. The result of these changes is illustrated in Fig. 2, which depicts the "'*Crear: Scrollbar*" Window before and after changes. Note that this change was applied on each mandatory field in the corresponding windows.

Once all the raised problems were fixed on the application, the "*New Situation*" (i.e., the JGUIAr v. 1.4) was produced.

(A3) Implement Measurement and Evaluation for the System Quality View: In the second iteration of the A3 activity, we used the same metrics and indicators that were used in the first measurement and evaluation cycle. But now, it was carried out on the new version of the evaluable entity i.e., the JGUIAr v 1.4 app. The yielded results are shown in Table 4, third column.

(A4.2) Analyze Results: In this activity, we analyzed the obtained gain, i.e., the improvement for those attributes that we took into account by performing the changes. For example, after adding accordingly all missing visual indicators to each mandatory field, the "*Error Prevention for Mandatory Entries*" attribute increased from 0% (in JGUIAr v. 1.3) to 100% (in v. 1.4). This and other changes allowed that "*Usability*" went from 71.87% to 94.64%. Regarding the "*Information Quality*" and "*Functional Quality*" sub-characteristics, the changes allowed to increase them from 81.9% to 100% and from 58.25% to 89.78%, respectively. Consequently, the "*External Quality*" evaluation focus raised from 68.87% to 94.14%.

Ultimately, at this moment, we can "*Analyze if external quality has improved 15 p.p. after changes*" across the 3-month timeframe (as stated in the information need goal in Fig. 6). Specifically, we can assess if the related business goal "*Improve 15 p.p. the current JGUIAr app's external quality in 3 months*" has been achieved. For our practical case, the JGUIAr new version has reached an increase of 25 percentage points, so the business goal was achieved successfully and the process was thus finalized. If the case were that it was not achieved and there would be time within the 3-month timeframe, a new change and evaluation cycle can be performed.

6. Related Work

As quoted in the Introduction Section, we have documented in [1] a family of strategies for various evaluation purposes. These strategies are made up of three integrated pillars such as process specifications, method specifications, and well-established domain vocabularies. Regarding the latter aspect, in the present work, we have integrated the FRs conceptual base to the previously developed NFRs and business goal vocabularies. Hence, strategies have now a wider, shared conceptual ground for supporting not only evaluation goal purposes but also testing, development and maintenance goal purposes.

Considering that a strategy is an important resource in helping to achieve project goal purposes, it should be noted that there exist few strategies that integrate these three pillars at once, as discussed in [2]. An approach related to integrated strategies is *GQM$^+$Strategies* [19], which includes a goal-oriented framework for the design and implementation of measurement software projects at different organizational levels. In this approach, business goals are linked to measurement goals using the *Goal Question Metrics* method. *GQM$^+$Strategies* has a vocabulary structured as a glossary. But it neither makes explicit the FR and NFR terms and their relationships with business goals, nor has the semantic richness as an ontology provides.

Regarding conceptual bases for NFRs and FRs linked to organizational goals we can consider [20], in which the goals are classified into business, enterprise architecture, or IT goals. The scope of the proposed vocabulary aims at designing decision-making strategies. But the approach does not consider integrated strategies for evaluation goal purposes, as documented in [1].

Another related work worth mentioning is [21], where authors state that NFRs are not stand-alone goals, as their existence is always dependent on other concepts in the project context. Also, considering ISO 9126 standard (the former one to [13]), they state that NFRs, namely quality requirements, are defined with an existential restriction to have at least one association point with FRs as they represent a set of attributes that bear on the existence of a set of functions and their specified properties. Moreover, they relate NFRs with ME concepts. However, the [21] terminology is not exploited as one pillar by ME integrated strategies. In [22] authors distinguish between functional goals that need to be achieved by functions performed by the system or an external actor, and quality goals that capture qualities of the system. They propose a modeling language for NFRs but do not develop any strategy that helps to achieve goals as we do.

In [23], authors present a method for deriving event-based specifications. It is written in the SCR

(*Software Cost Reduction*) tabular language, from operational specifications built according to the KAOS goal-oriented method. Thanks to this method, SCR specifiers may follow upstream goal-based processes to incrementally elaborate, structure and document their tabular specification in a guided fashion, and to perform goal-level analysis for earlier detection and resolution of obstacles and conflicts. Conversely, KAOS modelers may obtain downstream tabular specifications in a systematic way for later specification analysis through exhaustiveness checking, simulation, model checking and test data generation. As another benefit, domain and requirements models can be captured in terms of a rich ontology –goals, agents, requirements, expectations, objects, etc. However, in this approach, strategies that help to achieve goals are not considered.

Additionally, some research considers only NFRs and FRs terminologies without explicitly linking them with business and information need goal concepts. For example, [24] presents the Softgoal UML profile for seamlessly representing NFRs and FRs using the goal-oriented NFR Framework [25]. This profile defines the visual syntax and semantics of the modeling concepts, relationships, and constraints. On the other hand, authors in [26] present an evolution of a Software Requirements Ontology (SRO) that was reengineered by mapping the concepts of its previous version to the Unified Foundational Ontology (UFO). They say that "*SRO also covers requirements quality evaluation that was not discussed in this paper. However, there are other aspects of the requirements domain that are not addressed by the SRO and that should be incorporated to it in future works, such as traceability of requirements to business goals*". However, to our knowledge the referred future work has not been issued so far.

In [27], authors claim that the definitions of NFR/FR and hardgoal/softgoal are, in fact, orthogonal, allowing them to identify NFRs that are in fact hardgoals as well as FRs that are softgoals. Moreover, they state that categories of NFRs and FRs are not disjoint. We can observe, however, that the NFR and FR terms are not related between them as we have discussed in Section 3.

Lastly, [28] presents a framework for integrating NFRs (modelled using an AND/OR tree) into the Entity Relationship and Object Oriented models. A set of heuristics is used in order to guide software engineers in the early integration of NFRs to the requirements model. Authors believe that the systematic integration of NFRs into conceptual models may be helpful in identifying conflicts between NFRs and FRs. They use a taxonomy to classify NFRs into primary and specific NFRs. Specific NFRs are those that decompose a primary NFR. They also classify NFRs as dynamic and static NFRs. However, diverse strategies for different goal purposes are not considered.

7. Final Remarks

In this paper, we have addressed the linking of business and information need goal terms with FRs and NFRs terms, which are paramount for envisioning and developing strategies for a greater variety of purposes. We hypothesize that having robust vocabularies serving as a common ground for different strategies may foster a more effective operationalization of evaluation, testing, development and maintenance/change project goal purposes.

Specifically, we have integrated a set of vocabularies structured as ontologies. Consequently, the original conceptual components –in particular, the NFRs and context subontologies- underwent some slight updates in order to harmonize the conceptual framework as a whole. It is important to remark that the terms, attributes and relationships included in the ontologies are the minimum and necessary for the scope of our current work, i.e., to build evaluation, testing and development/maintenance strategies, to promote a shared understanding for all the strategies, and to produce more consistent specifications for processes and methods.

In addition, we have employed an academic software application currently in use in an OOP course as a running example to demonstrate the linking of business and information need goal terms with FRs and NFRs terms, attributes and relationships. In Section 4, we have used this example aimed at verifying that the proposed conceptual bases have practical potential. Specifically, we have focused on an evaluation business goal at operational level of an organization for the 'improvement' purpose. This purpose implies activities and methods for measurement, evaluation and change. Therefore, it requires instantiating the terminology specified in the NFRs and FRs conceptual components. Moreover, in Section 5, we have shown the applicability of the GOCAMEC strategy highlighting how its activities and methods are based on the proposed conceptual bases.

As future lines of research, we will augment the family of strategies with new integrated strategies for other goal purposes. For example, strategies that help to achieve business goals for testing purposes. As the reader can figure out, the new strategies will need particular specifications of processes and methods for the testing domain. Therefore, a testing component should be built and related with the already developed FRs and NFRs components. Moreover, the former evaluation strategies could be integrated with the new testing strategies, since testing goal purposes could require supporting ME information needs, e.g., to evaluate the level of efficiency and effectiveness of the testing endeavor, or to analyze the amount of types of errors and faults by severity detected in the testing process.

Note that we just have finished a Systematic Literature Review on Software Testing Ontologies [29]. So as an ongoing work, from this secondary study and other sources such as software testing standards, we are developing a top-domain software testing component that is linked with the FRs y NFRs ontologies, as suggested by the relationships between these conceptual components in Fig. 1. Additionally, we are currently placing and harmonizing the previous depicted conceptual components (and others new components to be built) into the framework of the ontological conceptual architecture shown in Fig. 4.

Acknowledgements

This work and line of research are supported by Science and Technology Agency of Argentina, in the PICT 2014-1224 project at Universidad Nacional de La Pampa.

Competing interests

The authors have declared that no competing interests exist.

References

[1] Olsina L., Becker, P.: Family of Strategies for different Evaluation Purposes. In: XX Conferencia Iberoamericana en Software Engineering (CIbSE'17) held in the framework of ICSE, CABA, Argentina, Published by Curran Associates 2017, pp. 221-234, ISBN 978-99967-839-2-0, (2017)

[2] Becker P., Papa F., Olsina L.: Process Ontology Specification for Enhancing the Process Compliance of a Measurement and Evaluation Strategy. In: CLEIej.18:(1), pp. 1-26, (2015). DOI: http://dx.doi.org/10.19153/cleiej.18.1.2

[3] Curtis B., Kellner M., Over J.: Process Modelling. Communications of ACM, 35:(9), pp.75-90, (1992)

[4] Rivera M.B., Becker, P., Papa, M.F., Olsina L.: A Holistic Quality Evaluation, Selection and Improvement Approach driven by Multilevel Goals and Strategies. In: CLEIej.19:(3), Paper 3, pp. 1-28. ISSN 0717-5000, (2016)

[5] Olsina L., Papa F., Molina H.: How to Measure and Evaluate Web Applications in a Consistent Way. Web Engineering: Modelling and Implementing Web Applications, Rossi G., Pastor O., Schwabe D., Olsina L. (Eds.), Springer HCIS, Chapter13, pp. 385-420, (2008)

[6] Olsina L., Becker P.: Linking Business and Information Need Goals with Functional and Non-functional Requirements. In: Proceed. of the XXI Conferencia Iberoamericana en Software Engineering (CIbSE'18), Bogotá, Colombia, Published by Curran Associates, pp. 381-394, (2018)

[7] Olsina L., Lew P., Dieser A., Rivera B.: Updating Quality Models for Evaluating New Generation Web Applications. Journal of Web Engineering, Special issue: Quality in new generation Web applications. Rinton Press. USA. 11:(3), 209-246, (2012)

[8] Uschold, M.: Knowledge Level Modelling: Concepts and Terminology. The Knowledge Engineering Review, 13 (1): pp. 5-29, (1998)

[9] Molina H., Olsina L.: Assessing Web Applications Consistently: A Context Information Approach, In: Proceed. of IEEE Computer Society, 8th Int'l Congress on Web Engineering (ICWE08), NY, USA, pp. 224-230, ISBN 978-0-7695-3261-5, (2008)

[10] Rivera M.B., Becker P., Olsina L.: Quality Views and Strategy Patterns for Evaluating and Improving Quality: Usability and User Experience Case Studies, In: Journal of Web Engineering, Rinton Press, US, 15:(5&6), pp. 433-464, ISSN 1540-9589, (2016)

[11] Fernández-López, M., Gómez-Pérez, A., and Juristo, N.: Methontology: From Ontological Art Towards Ontological Engineering. In: Proceed. of the Ontological Engineering American Association for Artificial Intelligence, pp. 33-40, (1997)

[12] OMG: Unified Modeling Language, v2.5.1, (2017)

[13] ISO/IEC 25010: Systems and software engineering - Systems and software Quality Requirements and Evaluation (SQuaRE) - System and software quality models, (2011)

[14] ISO/IEC/IEEE 15939: Software Engineering - Software Measurement Process, 1st Ed., (2017)

[15] Fleetwood S.: The ontology of things, properties and powers. Journal of Critical Realism, 8:(3), pp. 343-366, Available from: http://eprints.uwe.ac.uk/15967, (2009)

[16] Lew P., Olsina L., Zhang L.: Quality, quality in use, actual usability and user experience as key drivers for web application evaluation. In: Proceed. of the LNCS 6189, Springer, 10th Int'l Congress on Web Engineering (ICWE2010), Vienne, Austria, pp 218–232, (2010)

[17] Dujmovic J.: A Method for Evaluation and Selection of Complex Hardware and Software Systems. 22nd Int'l Conference for the Resource

Management and Performance Evaluation of Enterprise CS. CMG 96 Proceedings, pp. 368-378, (1996)

[18] Martilla J., James J.: Importance-Performance Analysis. Journal of Marketing, 41:(1), pp. 77-79, (1977)

[19] Basili V., Lindvall M., Regardie M., Seaman C., Heidrich J., Jurgen M., Rombach D., Trendowicz A.: Linking Software Development and Business Strategy through Measurement. IEEE Computer, 43:(4), pp. 57–65, (2010)

[20] Plataniotis G., Kinderen S., Qin M., Proper E.: A Conceptual Model for Compliance Checking Support of Enterprise Architecture Decisions. In: Proc. of IEEE, 17th Conference on Business Informatics, Lisbon, Portugal, pp. 191-198, (2015)

[21] Kassab M., Ormandieva, O., Daneva, M.: An Ontology Based Approach to Non-Functional Requirements Conceptualization. In: 4th Int'l Conf. on Software Engineering Advances (ICSEA), Porto, Portugal, pp. 299- 308, (2009)

[22] Li F-L., Horkoff J., Mylopoulos J., Guizzardi R., Guizzardi G., Borgida A., Liu L.: Non-functional Requirements as Qualities, with a Spice of Ontology. In: IEEE 22nd Int'l Requirements Engineering Conference (RE), Karlskrona, Sweden, pp. 293-302, (2014)

[23] De Landtsheer R., Letier E. and van Lamsweerde A.: Deriving tabular event-based specifications from goal-oriented requirements models. In: Proceed. of RE'03, 11th IEEE Joint Int'l Requirements Engineering Conference, Monterey (CA), pp. 200-210, (2003). DOI: 10.1109/ICRE.2003.1232751

[24] Supakkul S., Chung L.: A UML profile for goal-oriented and use case-driven representation of NFRs and FRs. In: Proceed. 3rd ACIS Int'l Conf. on Software Engineering Research, Management and Applications (SERA), Mount Pleasant, USA, pp. 112- 119, (2005)

[25] Mylopoulos J., Chung L., and Nixon B. A.: Representing and using nonfunctional requirements: A process-oriented approach. In: IEEE Transactions on Software Engineering, 18:(6), pp. 483-497, (1992)

[26] Falbo R., Nardi J.: Evolving a Software Requirements Ontology. In: Proceed. of the XXXIV Conferencia Latinoamericana de Informática (CLEI), Santa Fe, Argentina, 10 pags., (2008)

[27] Guizzardi R., Li F-L., Borgida A., Guizzardi G., Horkoff J., Mylopoulos J.: An Ontological Interpretation of Non-Functional Requirements. In: Proc. 8th Int'l Conf. on Formal Ontology in Information Systems (FOIS), Rio de Janeiro, Brazil, Vol 267, pp. 344-357, (2014).

[28] Cysneiros L.M., Leite J.C.S.P. and Neto, J.S.M.: A Framework for Integrating Non-Functional Requirements into Conceptual Models. In: Requirements Engineering Journal, 6:(2), pp: 97-115, (2001). DOI: https://doi.org/10.1007/s007660170008

[29] Tebes G., Peppino D., Becker P., Matturro G., Solari M., Olsina L.: A Systematic Review on Software Testing Ontologies. To appear in Springer proceedings of QUATIC, Ciudad Real, Spain, pp. 1-14, (2019)

Citation: P. Becker, G. Tebes, D. Peppino and L. Olsina. *Applying an Improving Strategy that embeds Functional and Non-Functional Requirements Concepts.* Journal of Computer Science & Technology, vol. 19, no. 2, pp. 153-174, 2019.
DOI: 10.24215/16666038.19.e15
Received: May 2, 2019 **Accepted:** August 26, 2019.