Specifying Process Views for a Measurement, Evaluation and Improvement Strategy

Pablo Becker¹, Philip Lew² and Luis Olsina¹

¹GIDIS/Web, Engineering School at Universidad Nacional de La Pampa, Argentina
²School of Software, Beihang University, China

beckerp@ing.unlpam.edu.ar, philiplew@gmail.com, olsinal@ing.unlpam.edu.ar

Any organization that develops software strives to improve the quality of its products. To do this first requires an understanding of the quality of the current product version. Then by iteratively making changes, the software can be improved with subsequent versions. But this must be done in a systematic and methodical way and for this purpose, we have developed a specific strategy called SIQinU (Strategy for understanding and Improving Quality in Use). SIQinU recognizes problems of quality in use through evaluation of a real system-in-use situation, and proposes product improvements by understanding and making changes to the product’s attributes. Then, re-evaluating quality in use of the new version, improvement gains can be gauged along with the changes that led to those improvements. SIQinU aligns with GOCAME (Goal-Oriented Context-Aware Measurement and Evaluation), a multi-purpose generic strategy previously developed for measurement and evaluation, which utilizes a conceptual framework (with ontological base), a process, and methods and tools. Since defining SIQinU relies on numerous phase and activity definitions, in this paper we model different process views, e.g. taking into account activities, interdependencies, artifacts and roles, while illustrating them with excerpts from a real case study.

Keywords: Process model, quality in use, quality improvement, SIQinU strategy, GOCAME strategy.

1. INTRODUCTION

Even though software product launches now may consist of ‘continuous beta’, users expect more and better functionality, combined with increased quality from the user’s perception. Methodically improving the perceived quality (i.e. its quality in use) particularly for web applications (WebApps) –a kind of software application- is not an easy job. WebApps are no longer simple websites conveying information. Rather, they have become fully functional software applications often with complex business logic and sometimes critical to operating the business. Users, in addition, are becoming more demanding and diverse in their requirements. Consequently, WebApp quality and especially the quality in use –namely the perceived quality by the end user (QinU) has taken on increased significance as web and now cloud deployment have become mainstream delivery methods. Systematic means for evaluating QinU is important because it enables understanding the quality satisfaction level achieved by the application, and provides useful information for recommendation and improvement processes in a consistent manner over time. Coincident with consistent and systematic evaluation of WebApp quality, the main goal is to ultimately improve its QinU.

This leads to our strategy with the objectives of understanding and improving the QinU of WebApps. QinU is currently redefined in the ISO 25010 standard [16], which was reused and enlarged in [18]. QinU from the actual usability standpoint (that embraces performance or ‘do’ goals in contrast to hedonic or ‘be’ goals [6]), is defined as the degree to which specified users can achieve specified goals with effectiveness in use, efficiency in use, learnability in use, and accessibility in use in a specified context of use [18]. Utilizing this model, we developed SIQinU as an integrated means to evaluate and find possible problems in QinU which are then related to external quality (EQ) characteristics and attributes (by doing a mapping between QinU problems and EQ). This is followed by evaluating the application from the EQ standpoint and then making recommendations for improvements if necessary. The new version, based on recommended improvements is re-evaluated to gauge the improvement gain from both the EQ and QinU point of views. One aspect of SIQinU’s uniqueness is that it collects user usage data from Web Apps in a
real context of use whereby code snippets are inserted (or using similar techniques) to gather data related to the task being executed by users at the sub-task level enabling non-intrusive evaluations.

It is worth mentioning that SIQinU aligns with the GOCAME strategy [22]. GOCAME, a multi-purpose goal-oriented strategy, was previously developed for supporting measurement and evaluation (M&E) programs and projects. Its rationale is based on three main pillars, namely: i) a conceptual framework utilizing an ontological base; ii) a well-defined measurement and evaluation process; and, iii) quality evaluation methods and tools instantiated from both the framework and process.

GOCAME’s first principle is that designing and implementing a robust M&E program requires a sound conceptual framework. Often times, organizations conduct start and stop measurement programs because they don’t pay enough attention to the way nonfunctional requirements, contextual properties, metrics and indicators should be designed, implemented and analyzed. Any M&E effort requires an M&E framework built on a sound conceptual base, i.e., on an ontological base, which explicitly and formally specifies the main agreed concepts, properties, relationships, and constraints for a given domain. To accomplish this, we utilize the C-INCAMI (Contextual-Information Need, Concept model, Attribute, Metric and Indicator) framework and its components [20, 22] based on our metrics and indicators ontology.

GOCAME’s second principle requires a well-established M&E process in order to guarantee repeatability in performing activities and consistency of results. A process prescribes a set of phases, activities, inputs and outputs, interdependencies, sequences and parallelisms, check points, and so forth. Frequently, process specifications state what to do but don’t mention the particular methods and tools to perform specific activity descriptions. Thus, to provide repeatability and replicability in performing activities, a process model for GOCAME was proposed in [4], which is also compliant with both the C-INCAMI conceptual base and components. Finally, methods and tools –the third pillar in the GOCAME strategy- can be instantiated from both the conceptual framework and process; e.g. the WebQEM methodology [23] and its tool called C-INCAMI_tool [22].

SIQinU utilizes the above three GOCAME principles while also re-using the C-INCAMI conceptual base and process. However, since SIQinU is a specific-purpose goal-oriented strategy, it has specific processes, methods and procedures that are not specified in GOCAME. Since the process aspect is critical in specifying SIQinU –given of its numerous interrelated phases and activities, this work defines its process model in detail through illustration with excerpts of a real case study. Note that processes can be modeled taking into account different views [9] such as: i) functional that includes the activities’ structure, inputs and outputs, etc.; ii) informational that includes the structure and interrelationships among artifacts produced or consumed by the activities; iii) behavioral that models the dynamic view of processes; and, iv) organizational that deals with agents, roles and responsibilities. Additionally, a methodological view is described in [21], which is used to represent the process constructors (e.g. specific methods) that can be applied to different descriptions of activities. In order to specify all these views different modeling languages can be used. However, no modeling language fits all needs and preferences. Each has its own strengths and weaknesses, which can make it more suitable for modeling certain views than others [1].

This paper using UML 2.0 activity diagrams [27] and the SPEM profile [26], stresses the functional, informational, organizational and behavioral views for the SIQinU process. Modeling its process helps to: i) ease the repeatability and understandability among practitioners, ii) integrate and formalize different activities that are interrelated in different phases, and iii) promote the learnability and interoperability by reusing the same ontological base coming from the C-INCAMI framework. This manuscript is an extension of the work presented in [5] elaborating on new aspects and views (e.g., informational and organizational) for both GOCAME and SIQinU process, as we remark later on. Summarizing, the main contributions of this paper are:

- A six-phased strategy (SIQinU) useful for understanding and improving the QinU for WebApps, which is specified and illustrated from the process viewpoint regarding activities (i.e. the functional view), interdependencies (behavioral view), artifacts (informational view) and roles (organizational view).
Foundations for reusing a multi-purpose goal-oriented strategy (i.e. GOCAME) to derive and integrate more specific-purpose strategies (e.g. SIQinU) regarding its conceptual M&E framework, methods and process views.

The remainder of this manuscript is organized as follows. Section 2 gives an overview of the six-phased SIQinU strategy. Section 3 provides the GOCAME rationale considering its three pillars, which are to a great extent reused by SIQinU; particularly, in sub-section 3.4 we discuss why SIQinU is in alignment with GOCAME regarding its conceptual M&E framework (sub-section 3.1), its process views (sub-section 3.2) and its methods (sub-section 3.3). Section 4 models and illustrates the six-phased SIQinU process from the above mentioned process views. Section 5 analyzes related work considering the two quoted contributions, and finally, in Section 6, concluding remarks as well as future work are discussed.

2. OVERVIEW OF THE SIQINU STRATEGY

SIQinU is an evaluation-driven strategy to iteratively and incrementally improve a WebApp’s QinU by means of mapping actual system-in-use problems—that happened while real users were performing common WebApp tasks—to measurable EQ product attributes and by then improving the current WebApp and assessing the gain both at EQ and QinU levels. SIQinU can be implemented in an economic and systematic manner that alleviates most of the problems identified with typical usability testing studies which can be expensive, subjective, non-repeatable, time consuming, and unreliable due to users being observed in an intrusive way. This is accomplished through utilizing server-side capabilities to collect user usage data from log files—adding, for example, snippets of code in the application to specifically record data used to calculate measures and indicator values for QinU in a non-intrusive way.

The SIQinU strategy uses quality models such as those specified in the ISO 25010 standard [15] and its enhancement, i.e. the 2Q2U (internal/external Quality, Quality in use, actual Usability and User experience) framework [18]. Once the QinU model has been established, the data collected, and metrics and indicators calculated a preliminary analysis is made. If the agreed QinU level is not met, then EQ requirements are derived considering the application’s QinU problems and its tasks, subtasks and associated screens. In turn, taking into account the derived EQ requirements, an evaluation of the WebApp attributes is performed by inspection. Thus a product analysis regarding the EQ evaluation is performed, and changes for improvement are recommended. If the improvement actions have been implemented, then the new version is re-evaluated to gauge the improvement gain both from the EQ and the QinU standpoint. Ultimately, SIQinU is a useful strategy not only for understanding but also—and most importantly—for improvement purposes.

![Process overview of SIQinU stressing phases and interdependencies](image)

**Figure 1.** Process overview of SIQinU stressing phases and interdependencies

SIQinU uses the concepts for nonfunctional requirements specification, measurement and evaluation
design, etc., established in the C-INCAMI framework as we will see in sub-section 3.1. Also SIQinU has an integrated, well-defined and repeatable M&E process, which follows to great extent the GOCAME process as we will discuss in sub-sections 3.2 and 3.3. Specifically, the SIQinU process embraces six phases as shown in Fig. 1, which stresses the main phases and interdependencies. Additionally, table 1 provides, with Phase (Ph.) reference numbers as per Fig. 1, a brief description of each phase, the involved activities and main artifacts. Section 4 thoroughly illustrates phases, activities, interdependencies, artifacts, as well as roles taking into account aspects of the functional, behavioral, informational, and organizational views.

Table 1. SIQinU Phases, activities and artifacts.

<table>
<thead>
<tr>
<th>Phases (Ph.)</th>
<th>Phase Description and Activities involved</th>
<th>Artifacts (Work Products)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ph. II</td>
<td>Perform QinU Evaluation and Conduct Preliminary Analysis</td>
<td>- Parsed data file - Measure and indicator values for QinU - QinU preliminary analysis report</td>
</tr>
<tr>
<td>Ph. III</td>
<td>Derive/ Specify Requirements and Evaluation Criteria for EQ</td>
<td>- EQ NFR tree - EQ metrics and indicators specification</td>
</tr>
<tr>
<td>Ph. IV</td>
<td>Perform EQ Evaluation and Analysis</td>
<td>- Measure and indicator values for EQ - EQ Analysis report (and new report after re-evaluation)</td>
</tr>
<tr>
<td>Ph. V</td>
<td>Recommend, Perform Improvement Actions, and Re-evaluate EQ</td>
<td>- EQ Recommendations report - Improvement plan - New application version</td>
</tr>
<tr>
<td>Ph. VI</td>
<td>Re-evaluate QinU and Analyze Improvement Actions</td>
<td>- New measure and indicator values for QinU - QinU Improvement analysis report</td>
</tr>
</tbody>
</table>
Lastly, in the introduction we stated as contribution that GOCAME—a previously developed strategy—can be reused to derive and integrate more specific-purpose strategies (as is the case of SIQinU) regarding its conceptual M&E framework, process and methods. Therefore, in the following section the GOCAME strategy regarding these principles is outlined.

3. GOCAME STRATEGY

GOCAME is a multi-purpose M&E strategy that follows a goal-oriented and context-sensitive approach in defining projects. It allows the definition of M&E projects including well-specified context descriptions, providing therefore more robust evaluation interpretations among different project results at intra- and inter-organization levels.

GOCAME is based on the three above-mentioned pillars, namely: a conceptual framework (described in sub-section 3.1); a M&E process (sub-section 3.2); and methods and tools (sub-section 3.3). Finally, in sub-section 3.4 we discuss why SIQinU is in alignment with GOCAME regarding these capabilities.

3.1. C-INCAMI Conceptual Framework

The C-INCAMI framework provides a domain (ontological) model defining all the concepts and relationships needed to design and implement M&E processes. It is an approach in which the requirements specification, M&E design, and analysis of results are designed to satisfy a specific information need in a given context. In C-INCAMI, concepts and relationships are meant to be used along all the M&E activities. This way, a common understanding of data and metadata is shared among the organization's projects lending to more consistent analysis and results across projects.

Following the main activities of the process (shown in sub-section 3.2), the framework—i.e. the related concepts and relationships—is structured in six components or modules, namely:

i) measurement and evaluation project definition;
ii) nonfunctional requirements specification;
iii) context specification;
iv) measurement design and implementation;
v) evaluation design and implementation; and,
v) analysis and recommendation specification.

For illustration purposes, Fig. 2 shows the main concepts and relationships for four components (i.e. from ii to v), and table 2 defines the used terms—stressed in italic in the following text). The entire modeling of components can be found in [20, 22].

Briefly outlined, the GOCAME strategy follows a goal-oriented approach in which all the activities are guided by agreed Information Needs; these are intended to satisfy particular nonfunctional requirements of some Entity for a particular purpose and stakeholder's viewpoint. The nonfunctional requirements are represented by Concept Models including high-level Calculable Concepts, as in ISO 25010’s quality models [16], which in turn measurable Attributes of the entity under analysis are combined. The instantiated quality models are the backbone for measurement and evaluation. Measurement is specified and implemented by using Metrics, which define how to represent and collect attributes' values; and Evaluation is specified and implemented by using Indicators, which define how to interpret attributes' values and calculate higher-level calculable concepts of the quality model.
Figure 2. Main concepts and relationships of the C-INCAMI framework. Four out of six C-INCAMI components are depicted as packages, namely: nonfunctional requirements specification, context specification, measurement design and implementation, and evaluation design and implementation.

Table 2. Some M&E terms - see [22] for more details.

<table>
<thead>
<tr>
<th>Concept</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Project Terms</strong></td>
<td></td>
</tr>
<tr>
<td>Evaluation Project</td>
<td>A project that allows, starting from a measurement project and a concept model of a nonfunctional requirement project, assigning indicators and performing the calculation in an evaluation process.</td>
</tr>
<tr>
<td>Measurement Project</td>
<td>A project that allows, starting from a nonfunctional requirements project, assigning metrics to attributes and recording the values in a measurement process.</td>
</tr>
<tr>
<td>MEPProject (i.e. Measurement and Evaluation Project)</td>
<td>A project that integrates related nonfunctional requirements, measurement and evaluation projects, and then allows managing and keeping track of all related metadata and data.</td>
</tr>
<tr>
<td>Project</td>
<td>Planned temporal effort, which embraces the specification of activities and resources constraints performed to reach a particular goal.</td>
</tr>
<tr>
<td>Non-Functional Requirements Project</td>
<td>A project that allows specifying nonfunctional requirements for measurement and evaluation activities.</td>
</tr>
<tr>
<td><strong>Nonfunctional Requirements Terms</strong></td>
<td></td>
</tr>
<tr>
<td>Attribute (synonyms: Property, Feature)</td>
<td>A measurable physical or abstract property of an entity category.</td>
</tr>
<tr>
<td>Calculable Concept (synonym: Characteristic, Dimension)</td>
<td>Abstract relationship between attributes of entities and information needs.</td>
</tr>
<tr>
<td><strong>Concept</strong></td>
<td><strong>Definition</strong></td>
</tr>
<tr>
<td>-------------</td>
<td>---------------</td>
</tr>
<tr>
<td>Concept Model (synonyms: Factor, Feature Model)</td>
<td>The set of sub-concepts and the relationships between them, which provide the basis for specifying the concept requirement and its further evaluation or estimation.</td>
</tr>
<tr>
<td>Entity (synonym: Object)</td>
<td>A concrete object that belongs to an entity category.</td>
</tr>
<tr>
<td>Entity Category</td>
<td>Object category that is to be characterized by measuring its attributes.</td>
</tr>
<tr>
<td>Information Need</td>
<td>Insight necessary to manage objectives, goals, risks, and problems.</td>
</tr>
<tr>
<td>Requirement Tree</td>
<td>A requirement tree is a constraint to the kind of relationships among the elements of the concept model, regarding the graph theory.</td>
</tr>
<tr>
<td><strong>Context Terms</strong></td>
<td></td>
</tr>
<tr>
<td>Context</td>
<td>A special kind of entity representing the state of the situation of an entity, which is relevant for a particular information need. The situation of an entity involves the task, the purpose of that task, and the interaction of the entity with other entities as for that task and purpose.</td>
</tr>
<tr>
<td>Context Property (synonyms: Context Attribute, Feature)</td>
<td>An attribute that describes the context of a given entity; it is associated to one of the entities that participates in the described context.</td>
</tr>
<tr>
<td><strong>Measurement Terms</strong></td>
<td></td>
</tr>
<tr>
<td>Calculation Method</td>
<td>A particular logical sequence of operations specified for allowing the realization of a formula or indicator description by a calculation.</td>
</tr>
<tr>
<td>Direct Metric (synonyms: Base, Single Metric)</td>
<td>A metric of an attribute that does not depend upon a metric of any other attribute.</td>
</tr>
<tr>
<td>Indirect Metric (synonyms: Derived, Hybrid Metric)</td>
<td>A metric of an attribute that is derived from metrics of one or more other attributes.</td>
</tr>
<tr>
<td>Measure</td>
<td>The number or category assigned to an attribute of an entity by making a measurement.</td>
</tr>
<tr>
<td>Measurement</td>
<td>An activity that uses a metric definition in order to produce a measure’s value.</td>
</tr>
<tr>
<td>Measurement Method (synonyms: Counting Rule, Protocol)</td>
<td>The particular logical sequence of operations and possible heuristics specified for allowing the realization of a direct metric description by a measurement.</td>
</tr>
<tr>
<td>Metric</td>
<td>The defined measurement or calculation method and the measurement scale.</td>
</tr>
<tr>
<td>Scale</td>
<td>A set of values with defined properties. Note: The scale type depends on the nature of the relationship between values of the scale. The scale types mostly used in software engineering are classified into nominal, ordinal, interval, ratio, and absolute.</td>
</tr>
<tr>
<td>Unit</td>
<td>A particular quantity defined and adopted by convention, with which other quantities of the same kind are compared in order to express their magnitude relative to that quantity.</td>
</tr>
<tr>
<td><strong>Evaluation Terms</strong></td>
<td></td>
</tr>
<tr>
<td>Decision Criterion (synonym: Acceptability Level)</td>
<td>Thresholds, targets, or patterns used to determine the need for action or further investigation, or to describe the level of confidence in a given result.</td>
</tr>
<tr>
<td>Elementary Indicator (synonyms: Elementary Preference, Criterion)</td>
<td>An indicator that does not depend upon other indicators to evaluate or estimate a calculable concept.</td>
</tr>
<tr>
<td>Elementary Model (synonym: Elementary Criterion Function)</td>
<td>Algorithm or function with associated decision criteria that model an elementary indicator.</td>
</tr>
<tr>
<td>Evaluation (synonym: Calculation)</td>
<td>Activity that uses an indicator definition in order to produce an indicator’s value.</td>
</tr>
<tr>
<td>Concept</td>
<td>Definition</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>---------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Global Indicator (synonyms:</td>
<td>An indicator that is derived from other indicators to evaluate or estimate a</td>
</tr>
<tr>
<td>Global Preference, Criterion</td>
<td>calculable concept.</td>
</tr>
<tr>
<td>Global Model (synonyms:</td>
<td>Algorithm or function with associated decision criteria that model a global</td>
</tr>
<tr>
<td>Scoring, Aggregation Model</td>
<td>indicator.</td>
</tr>
<tr>
<td>or Function)</td>
<td></td>
</tr>
<tr>
<td>Indicator (synonym: Criterion)</td>
<td>The defined calculation method and scale in addition to the model and decision</td>
</tr>
<tr>
<td></td>
<td>criteria in order to provide an estimate or evaluation of a calculable concept with</td>
</tr>
<tr>
<td></td>
<td>respect to defined information needs.</td>
</tr>
<tr>
<td>Indicator Value (synonym:</td>
<td>The number or category assigned to a calculable concept by making an evaluation.</td>
</tr>
<tr>
<td>Preference Value)</td>
<td></td>
</tr>
</tbody>
</table>

Since each MEProject does not occur in isolation, we therefore say that measurement and evaluation should be supported by Context; thus, context specifications may be provided in order to support sounder analysis, interpretations and recommendations. A summarized description for each component is provided below.

3.1.1 M&E Project Definition Component

This component defines and relates a set of Project concepts needed to articulate M&E activities, roles and artifacts.

A clear separation of concerns among Non-functional Requirements Project, Measurement Project and Evaluation Project concepts is made for reuse purposes as well as for easing management’s role. The main concept in this component is a Measurement and Evaluation Project (MEProject), which allows defining a concrete Requirement Project with the information need and the rest of the nonfunctional requirements information. From this requirement project, one or more Measurement Projects can be defined and associated; and, in turn, for each measurement project one or more Evaluation Projects could be defined. Hence for each Measurement and Evaluation Project we can manage associated sub-projects accordingly. Each project also has information such as responsible person’s name and contact information, starting and ending date, amongst other relevant information. Ultimately this separation of concerns for each MEProject facilitates the traceability and consistency for intra- and inter-project analysis.

3.1.2 Nonfunctional Requirements Specification Component

This component includes concepts and relationships needed to define the nonfunctional requirements for measurement and evaluation. One key concept is the Information Need, which specifies (see Fig. 2):

- the purpose for performing the evaluation (which can be for instance “understand”, “predict”, “improve”, “control”, etc.);
- the focus concept (CalculableConcept) to be assessed (e.g. “operability”, “quality in use”, “actual usability”, etc.);
- the category of the entity (EntityCategory) that will be assessed, e.g. a “Web application” (which its superCategory is a “product” or “information system”), and the concrete Entities (such as “Jira”, “Mantis” WebApps, etc.). Other super categories for entities can be “resource”, “process”, “information system-in-use” (e.g. as a Web application-in-use), and “project”.
- the userViewpoint (i.e. the intended stakeholder as “developer”, “final user”, etc.) from which the focus concept (and model) will be evaluated;
- the Context that characterizes the situation defined by the previous items to a particular MEProject.

The focus concept constitutes the higher-level concept of the nonfunctional requirements; in turn, a Calculable Concept and its sub-concepts are related by means of a Concept Model. This model may be a tree-structured representation in terms of related mid-level calculable concepts and lower-level measurable Attributes, which are associated to the target entity. Predefined instances of metadata for
information needs, entities and entity categories, calculable concepts, attributes, etc., and its corresponding data can be obtained from an organizational repository to support reusability and consistency in the requirements specification along the organizational projects.

3.1.3 Context Specification Component

This component includes concepts and relationships dealing with the context information specification. The main concept is Context, which represents the relevant state of the situation of the entity to be assessed with regard to the stated information need. We consider Context as a special kind of Entity in which related relevant entities are involved. Consequently, the context can be quantified through its related entities. By relevant entities we mean those that could affect how the focus concept of the assessed entity is interpreted (examples of relevant entities of the context may include resources as a network infrastructure, a working team, lifecycle types, the organization or the project itself, among others).

In order to describe the situation, attributes of the relevant entities (involved in the context) are used. These are also Attributes called Context Properties, and can be quantified to describe the relevant context of the entity under analysis. A context property inherits the metadata from the Attribute class such as name, definition and objective, and also adds other (see Fig. 2) information. All these context properties' metadata are meant to be stored in the organizational repository, and for each MEProject the particular metadata and its values are stored as well. A detailed illustration of context and the relationship with other C-INCAMI components can be found in [20].

3.1.4 Measurement Design and Implementation Component

This module includes the concepts and relationships intended to specify the measurement design and implementation, for instance, the concrete Entities that will be measured, the selected Metric for each attribute, and so on.

Regarding measurement design, a metric provides a Measurement specification of how to quantify a particular attribute of an entity, using a particular Method, and how to represent its values, using a particular Scale. The properties of the measured values in the scale with regard to the allowed mathematical and statistical operations and analysis are given by the scaleType [10]. 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 Method. On the other hand, Indirect Metrics' values are calculated from others direct metrics' values following a function specification and a particular Calculation Method.

For measurement implementation, a Measurement specifies the activity by using a particular metric description in order to produce a Measure value. Other associated metadata is the data collector name and the timestamp in which the measurement was performed.

3.1.5 Evaluation Design and Implementation Component

This component includes the concepts and relationships intended to specify the evaluation design and implementation. Indicator is the main term, which allows specifying how to calculate and interpret the attributes and calculable concepts of nonfunctional requirement models.

Two types of indicators are distinguished. First, Elementary Indicators that evaluate lower-level requirements, namely, attributes combined in a concept model. 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 agreed Decision Criteria, which help analyze the level of satisfaction reached by each elementary nonfunctional requirement, i.e. by each attribute. Second, Partial/Global Indicators, which evaluate mid-level and higher-level requirements, i.e. sub-characteristics and characteristics in a concept model. Different aggregation models (GlobalModel), like logic scoring of preference models, neuronal networks models, and fuzzy logic models, can be used to perform evaluations. The global indicator’s value ultimately represents the global degree of satisfaction in meeting the stated information need for a given purpose and user viewpoint.

As for the implementation, an Evaluation represents the activity involving a single calculation,
following a particular indicator specification—either elementary or global—producing an Indicator Value.

It is worthy to mention that the selected metrics are useful for a measurement process as long as the selected indicators are useful for an evaluation process in order to interpret the stated information need.

3.1.6 Analysis and Recommendation Specification Component

This component includes concepts and relationships dealing with analysis design and implementation as well as conclusion and recommendation. Analysis and recommendation component uses information coming from each MEProject (which includes requirements, context, measurement and evaluation data and metadata). By storing all this information and by using different kinds of statistical techniques and visualization tools, stakeholders can analyze the assessed entities’ strengths and weaknesses with regard to established information needs, and justify recommendations in a consistent way. Note this component is not shown in Fig 2. However, it is shown in table 5 from the process specification standpoint.

3.2. GOCAME Measurement and Evaluation Process

When modeling a process, often engineers think more about what a process must do rather than how activities should be performed. In order to foster repeatability and reproducibility, a process specifies (i.e. prescribes or informs) a set of phases and activities, inputs and outputs, interdependencies, among other concerns. Also, to deal with the inherent complexity of processes, process views—also quoted in process modeling literature as perspectives—are used. A view is a particular model or approach to represent, specify and communicate regarding the process. For instance, according to [9], a process can be modeled taking into account four views, namely: functional, behavioral, informational and organizational.

Considering these process views, the functional perspective for GOCAME represents what activities and tasks should be specified, what hierarchical activities structure (also known as task breakdown structure) there exists, what conditions (pre- and post-conditions) should be accomplished, and what inputs and outputs (artifacts) will be required. Taking into account the terminology and components used in the C-INCAMI framework (sub-section 3.1), the integrated process of GOCAME embraces the following core activities: i) Define Non-Functional Requirements; ii) Design the Measurement; iii) Design the Evaluation; iv) Implement the Measurement; v) Implement the Evaluation; and vi) Analyze and Recommend as shown in Fig. 3. In addition, in table 3 we enumerate these six activities, their involved sub-activities and the main output artifacts.

---

Instead of the often-used term “task” in process modeling, which represents a fine grained or atomic activity, we will use the term “sub-activity” in the rest of the text, since in Section 4, for QinU modeling the term task has a very specific meaning.
The behavioral view represents the dynamics of the process, i.e., the sequencing and synchronization of activities, parallelisms, iterations, feedback loops, beginning and ending conditions, among other issues. The core GOCAME activities as well as sequences, parallelisms, main inputs and outputs are depicted in Fig. 3. The <<datastore>> stereotype shown in the figure represents repositories; for instance, the Metrics repository stores the metadata for the previously designed metrics. More details for the GOCAME functional and behavioral process views can be found in [4].

Table 3. GOCAME core activities and main output artifacts

<table>
<thead>
<tr>
<th>Activities (A.)</th>
<th>Sub-Activities</th>
<th>Artifacts (Work Products)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>A.1 Define Non-Functional Requirements</strong></td>
<td>Sub-activities include: i) Establish Information Need; ii) Specify Project Context; and iii) Select a Concept Model. \nNote that these (and below) sub-activities can in turn be broken down in new ones -see [4] for more details.</td>
<td>- Non-Functional Requirements Specification (this artifact is composed of: i) Information Need Specification; ii) Context Specification; and iii) Non-Functional Requirements Tree)</td>
</tr>
<tr>
<td><strong>A.2 Design the Measurement</strong></td>
<td>Sub-activities include: i) Establish Entity (optional); and ii) Assign one Metric to each Attribute.</td>
<td>- Metrics Specification</td>
</tr>
<tr>
<td><strong>A.3 Implement the Measurement</strong></td>
<td>Sub-activities include: i) Establish Entity; and ii) Measure Attributes</td>
<td>- Measure values</td>
</tr>
<tr>
<td><strong>A.4 Design the Evaluation</strong></td>
<td>Sub-activities include: i) Identify Elementary Indicators; and ii) Identify Partial and Global Indicators.</td>
<td>- Indicators Specification (this artifact is composed of: i) Elementary Indicators Specification; and ii) Partial/Global Indicators Specification)</td>
</tr>
<tr>
<td><strong>A.5 Implement the Evaluation</strong></td>
<td>Sub-activities include: i) Calculate Elementary Indicators; and ii) Calculate Partial and Global Indicators</td>
<td>- Indicator values</td>
</tr>
<tr>
<td><strong>A.6 Analyze and Recommend</strong></td>
<td>Sub-activities include: i) Design the Analysis; ii) Implement the Analysis; iii) Elaborate the Conclusion Report; and iv) Perform Recommendations.</td>
<td>- Conclusion/Recommendation Report (this artifact is composed of: i) Analysis Specification; ii) Analysis Report; iii) Conclusion Report; and iv) Recommendations Report)</td>
</tr>
</tbody>
</table>

Figure 4. Excerpt of the informational view for A.1 and A.2 in table 3, regarding artifact composition. a) Non-Functional Requirements Specification documents; b) Metrics Specification document.

On the other hand, the informational view is concerned with those artifacts produced or required (consumed) by activities, the artifact breakdown structure, strategies of configuration management and traceability models. For example, for illustration purpose, in Fig. 4 the structure for the Non-Functional Requirements Specification and Metrics Specification documents, which are outputs of A.1 and A.2 activities (see table 3) is modeled. As the reader can observe in Fig. 4a, the Non-Functional Requirements Specification artifact is composed of the Information Need Specification, the Context Specification and the Non-Functional Requirements Tree documents. Besides, the Metrics Specification artifact (Fig. 4b) is composed of a set of one or more Metric Specification, which in turn is composed of a Scale and a
Calculation or Measurement Method descriptions. Note that aimed at easing the communication among stakeholders these models can complement the textual specification made in the third column of table 3.

Finally, the organizational view deals with what agents and their associated resources participate-plan-execute-control what activities; which roles (in terms of responsibilities and skills) are assigned to agents; what groups' dynamic and communication strategies are used, among other aspects. To illustrate this, Fig. 5 depicts the different roles and their associated GOCAME activities. In table 4 each role definition and its involved activities are also listed. Note that we have used italics in the definition column (in table 4) to show the terminological correspondence between the process role definition and the C-INCAMI conceptual framework. It is important to remark that a role can be assumed by a human or an automated agent. And, a human agent can be embodied by one or more persons, i.e. a team.

![Figure 5. The organizational view: Roles assigned to the GOCAME activities](image)

<table>
<thead>
<tr>
<th>Role Name</th>
<th>Definition/Comment</th>
<th>Activities (as per Fig. 3)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Quality Assurance (QA) Manager</td>
<td>Responsible for leading a measurement and evaluation project (MEProject in Table 2) regarding the requester needs. Note this role is specified by three sub-roles as per Fig 4.</td>
<td>Note this role is responsible of the activities involved in three specific sub-roles as per Fig 4.</td>
</tr>
<tr>
<td>Nonfunctional Requirements Manager</td>
<td>Responsible for the non-functional requirements project. This role should be played by a nonfunctional requirement engineer.</td>
<td>- Define Non-functional Requirements</td>
</tr>
<tr>
<td>Measurement Manager</td>
<td>Responsible for leading a measurement project.</td>
<td>- Design the Measurement - Implement the Measurement</td>
</tr>
<tr>
<td>Evaluation Manager</td>
<td>Responsible for leading an evaluation project.</td>
<td>- Design the Evaluation - Implement the Evaluation</td>
</tr>
<tr>
<td>Evaluation Requester</td>
<td>Responsible for requesting an evaluation. Note that this role can be accomplished by a human or an organization.</td>
<td>- Define Non-functional Requirements - Design the Measurement - Design the Evaluation</td>
</tr>
<tr>
<td>Metrics Expert</td>
<td>Responsible for identifying the appropriate metrics from a catalogue for each attribute of the requirements tree, based on the established information need.</td>
<td>- Design the Measurement</td>
</tr>
<tr>
<td>Data Collector</td>
<td>Responsible for gathering measures of the attributes using the metrics specification. Note that the data collector role can be accomplished by either a human agent or an automatic agent.</td>
<td>- Implement the Measurement</td>
</tr>
<tr>
<td>Indicators Expert</td>
<td>Responsible for identifying the most appropriate indicators from a catalogue and to define decision criteria for each attribute and calculable concept of the requirements tree based on the established information need.</td>
<td>- Design the Evaluation</td>
</tr>
</tbody>
</table>
Table 5. Process template in which information and views are documented for the Analyze and Recommend activity

**Activity**: Analyze and Recommend  
**Code** (in Fig. 3): A6

**Objective**: elaborate and communicate a conclusion report and (if necessary) a recommendation report for a decision-making process.

**Description**: identify and select procedures, techniques and tools to be used in order to analyze data, metadata and information, coming from metrics and indicators, for a given information need. Based on the analysis results a conclusion report is produced, and, a recommendations report, if necessary, is yielded as well. All these reports are communicated to the evaluation requester.

**Sub-activities**:
- Design the Analysis (A6.1)
- Implement the Analysis (A6.2)
- Elaborate the Conclusion Report (A6.3)
- Perform Recommendations (A6.4)

**Involved Roles**:
- Analysis Designer  
- Data Analyzer  
- Recommender  
- Results Communicator

**Input Artifacts**:
- Non-functional Requirements Specification  
- Metrics Specification  
- Indicators Specification  
- Measures  
- Indicators values  
- Project/business commitment

**Output Artifacts**:
- Conclusion/Recommendation report.

Note that this artifact is composed of: i) Analysis Specification; ii) Analysis Report; iii) Conclusion Report; and iv) Recommendations Report.

**Pre-conditions**: the MEProject must be implemented.

**Post-conditions**: the MEProject finishes when the conclusion and/or recommendation report is communicated and agreed on between the QA manager and the requester of the evaluation.

In order to combine the above views, table 5 presents a template which specifies just the Analyze and Recommend activity. The template specifies the activity name, objective and description, the sub-activities and involved roles, input and output artifacts, pre- and post-conditions. Also a diagram
representing the *Analyze and Recommend* activity is attached as well to enable understanding and communicability.

Summarizing, the GOCAME M&E process can be described as follows. Once the *non-functional requirements project* has been created by the *non-functional requirements manager*, then, the *define non-functional requirements* activity has a specific goal or problem (agreed with the *evaluation requester*) as input and a *nonfunctional specification document* as output. Then, in the *design the measurement* activity, the *metrics expert* identifies the metrics to quantify attributes. The metrics are selected from a *metrics repository*, and the output is the *metric specification document*. Once the measurement was designed – taken into account raised issues for the evaluator requester, e.g. the precision of metrics, etc.-, the *evaluation design* and the *measurement implementation* activities can be performed in any order or in parallel as shown in Fig. 3. Therefore, the *design the evaluation* activity is performed by the *indicators expert* who allows identifying elementary and global indicators and their acceptability levels (agreed also with the *evaluation requester*). Both the *measurement design* and the *evaluation design* are led by the *measurement and evaluation managers* accordingly. To the *implement the measurement* activity, the *data collector* uses the specified metrics to obtain the measures, which are stored in the *measures repository*. Next, the *implement the evaluation* activity can be carried out by the *indicator calculator* –this role usually is enacted by a tool. Finally, *analyze and recommend* activity is performed by *analysis designer*, *data analyzer*, *recommender* and *results communicator* roles. This activity has as inputs measures and indicators values (i.e. data), the requirements specification document, and the associated metrics and indicators specifications (i.e., metadata) in order to produce a *conclusion/recommendation report*.

### 3.3. GOCAME Methods and Tools: WebQEM and C-INCAMI_Tool

While activities state ‘what’ to do; methods describe ‘how’ to perform these activities accomplished by agents and roles, which in turn can be automated by tools. In addition a methodology is a set of related methods. Since the above M&E process includes activities such as specify the requirements tree, identify metrics, and so on, we have envisioned a methodology that integrates all these aspects and tools that automate them; i.e., a set of well-defined and cooperative methods, models, techniques and tools that, applied consistently to the process activities produces the different outcomes.

Particularly, the WebQEM (*Web Quality Evaluation Method*) [23] and its associated tool so-called C-INCAMI_Tool [22] (see screenshots in Fig. 6) were instantiated from the conceptual framework and process. The methodology supports an evaluation-driven approach, relying on experts and/or end users to evaluate and analyze different views of quality such as EQ and QinU for software and system-in-use applications. Note that GOCAME strategy and its methodology can be used to evaluate not only software/WebApps but also other entity categories, such as resources and processes.

In addition to the above mentioned views, a methodological view is presented in [21]. This represents the process constructors to be applied to the different descriptions of activities in a given process. Note that for a specific activity description we can have one or more methods that give support to the same activity, and for a given method we can have one or more tools that enact it. For instance in table 3, for the A.5 activity, and particularly for the *calculate the partial/global indicators* sub-activity, many methods can accomplish this such as “linear additive scoring method”, “neural network method”, among others.
3.4. Why is SIQinU in Alignment with GOCAME?

As we have indicated in the last paragraph of Section 2, SIQinU also relies on the three GOCAME principles above outlined. In fact, SIQinU utilizes the C-INCAMI conceptual base, underlying process and methods as we discuss in Section 4. However, since SIQinU is a specific-purpose goal-oriented strategy, it has specific activities, some particular methods and procedures that are not taken into account in GOCAME. Moreover, while GOCAME is a multi-purpose strategy regarding the strategy aim, SIQinU is a specific-purpose strategy. This is so, because in GOCAME the information-need purpose can be “understand”, “predict”, “improve”, “control”—as indicated in sub-section 3.1.2,—while in SIQinU the purposes are just “understand” and ultimately “improve”. In addition, GOCAME was designed to allow assessing different calculable concepts and entities such as the EQ or QinU of any product (including WebApps), the cost of a product, the capability quality of a resource, among others. Meanwhile, SIQinU was designed to allow evaluating specifically QinU of WebApps-in-use (in a non-intrusive way) and EQ of WebApps. Even more, in SIQinU, the QinU is evaluated from the “do goals” or pragmatic view, rather than from the “be goals” (subjective view), as discussed in [18].

Considering the process, SIQinU also re-uses the GOCAME activities. For example, the GOCAME A1, A2, A4 activities, and, to some extent, the A6 activity (recall Fig. 3) are included in the SIQinU Ph. I and Ph. III (recall Fig. 1). Likewise, the A3, and A5 activities are included in Ph. II, and Ph. IV phases. However, there are particular activities in SIQinU that are not included into GOCAME. For example, in Phase V we have activities devoted to produce WebApp improvements, as well as in Phase II, there exist activities for data filtering and collection, since SIQinU proposes utilizing server-side capabilities to gather, in a non-intrusive way, user usage data.

Also measurement and evaluation methods as commented in sub-section 3.3 are reused. However, other methods and techniques that are not included in GOCAME such as those for changing the current WebApp version (in Phase V) are needed. Finally, all the roles defined in table 4 are totally reused as well, adding new ones for the activities of Ph V as for example the “Maintenance Project Manager” role (i.e. the responsible for leading a maintenance project and identifying the appropriate methods, techniques and tools to be used for change—improve- the application), and the “Developer” role (i.e. the responsible for conducting the software/web application changes).
Despite the mentioned similarities with GOCAME, the modeling of the functional and behavioral views in SIQinU is necessary given the amount of involved phases, activities, sub-activities and their workflows. These issues will be highlighted in the next section.

4. A PROCESS MODEL VIEW FOR SIQINU

Process modeling is rather a complex endeavor. Given the inherent complexity of the process domain, a process can be modeled taking into account different views as analyzed in sub-section 3.2 for GOCAME. With the aim to model the SIQinU phases and activities, their inputs and outputs, sequences, parallelism, and iterations, we specify below –using UML activity diagrams and the SPEM profile [26], the functional view taking into account behavioral concerns as well. Aspects of the organizational and informational views are to a lesser extent specified in the following diagrams, since as indicated in subsection 3.4 many of the roles and artifacts are reused from the GOCAME strategy.

Also in Section 2 (Fig. 1 and table 1), we depicted the SIQinU phases, so below we concentrate on the specifications of activities and their descriptions. In order to illustrate the SIQinU process, excerpts of a case study conducted in mid-2010 are used [19]. This case study examined JIRA (www.atlassian.com), a defect reporting WebApp in commercial use in over 24,000 organizations in 138 countries around the globe. JIRA’s most common task, Entering a new defect, was evaluated in order to provide the most benefit, since entering a new defect represents a large percentage of the total usage of the application. We studied 50 beginner users in a real work environment in their daily routine of testing software and reporting defects in a software testing department of ABC, a company (with fictitious name but real one) specializing in software quality and testing. The beginner users were testers which were the majority of users. Although there are other user categories such as test managers, QA managers, and administrators, testers is the predominant user type, so we chose beginner testers as our user viewpoint.

4.1. Phase I: Specify Requirements and Evaluation Criteria for QinU

Once the requirements project has been created, using the data of the WebApp’s usage recorded in log files, we re-engineer and establish QinU requirements, i.e. characteristics with measurable attributes, with the objective of not only understanding but also improving the system-in-use with real users. From observations of the actual WebApp, this phase embraces defining user type, designing tasks, specifying usage context and dimensions for QinU (e.g. actual usability) and their attributes. Based on these specifications, metrics (for measurement) and indicators (for evaluation) are selected. Below we describe the seven core activities (see the flow in Fig. 7) involved in Ph. I:

![Figure 7. Overview for the Specify requirements and evaluation criteria for QinU process (Ph. I).](image)
4.1.1. Establish Information Need

This activity, according to the C-INCAMI framework (recall requirements package in Fig. 2), involves Define the purpose and the user viewpoint, Establish the object and the entity under study, and Identify the focus of the evaluation (see Fig. 8). These activities are accomplished by the NFR Manager role considering the evaluation requester’s needs. In the case study, the purpose for performing the evaluation is to “understand” and “improve” the WebApp being used from the userViewpoint of a “beginner tester”. The category of the entity (i.e. the object) assessed was a “defect tracking WebApp” while the entity being studied was “JIRA” (called in our study JIRA v.1). The focus (CalculableConcept) assessed is “actual usability” and its sub-characteristics, “effectiveness in use”, “efficiency in use”, and “learnability in use” [18].

4.1.2. Specify Project Context

Once the information need specification document is yielded, we optionally can Specify Project Context as shown in Figure 7. It involves the Select relevant Context Properties sub-activity –from the organizational repository of context properties [20], and, for each selected property the Quantify Context Property activity must be performed –based on the associated metric. In the end, we get as output a context specification document for the specific project.

4.1.3. Design Tasks

In this activity, the most common and representative task or tasks should be designed. It is also important to choose a task that is performed for which sufficient data can be collected. In our case study the selected task by the evaluation requester was “Entering a new defect”, as indicated above. This JIRA task included 5 sub-tasks specified by the task designer, namely: i) Summary, steps, and results; ii) Add Detail Info; iii) Add Environment Info; iv) Add Version Info; and v) Add Attachment (see details of tasks and screens in [19]).

4.1.4. Select QinU Concept Model

It involves both Select a Model and Edit the Model sub-activities. Concept models are chosen from an organizational repository regarding the quality focus. For example, in our case study, the NFR manager based on the previously stated information need, and taking into account the concept focus to evaluate actual usability, he instantiated a concept model for the ‘do goals’ of the user [18]. Then, if the selected model is not totally suitable, e.g. some sub-characteristics or attributes are missing, it is necessary to Edit the Model, adding or removing sub-concepts and/or attributes accordingly.

Finally, a requirements tree where attributes are the leaves and the concept focus is the root is yielded. For the selected concept model and regarding the information need and task at hand, the NFR manager instantiated the model as shown in Fig. 9 (attributes are in italic). Basically, the NFR manager, in the end, needs to satisfy the objectives of the sponsoring organization, i.e. the evaluation requester.
4.1.5. Design QinU Measurement

For each attribute of the requirements tree –highlighted in italic in Figure 9-, we Identify a Metric to quantify them. The appropriate metrics are selected from a repository. In the C-INCAMI framework, two types of metrics are specified, a direct metric which applies a measurement method, i.e. our data collection procedures from log files, and an indirect metric which uses a formula (based on other direct and/or indirect metrics) and calculation method (recall measurement package in Fig. 2). If the metric is indirect it is necessary Identify related Metrics and Identify Attributes quantified by related Metrics (see Fig. 10). These two sub-activities allow identifying the extra attributes and metrics for the indirect metric so that data collector may later gather the data accordingly.

![Figure 9: Instantiated QinU NFR tree for JIRA case study.](image)

In the JIRA case study, the metrics used to measure attributes were selected by the metrics expert from a metric catalogue which contains over 30 indirect metrics and their associated direct metrics. Below we illustrate the selected indirect metric for the sub-task completeness efficiency (coded 1.2.2 in Fig. 9) attribute:

Metric: Average ratio of sub-tasks that are completed incorrectly or correctly per unit of time to do it.  
(AvgRCput)

Interpretation: \(0 \leq \text{AvgRCput}, \) more is better.

Objective: Calculate the overall average proportion of the sub-tasks that are completed, whether correct or incorrect, per time unit (usually seconds or minutes).

Calculation Method (Formula):  
\[
\text{AvgRCput} = \frac{\text{AvgRC}}{\text{AvgTC}}
\]

\(\text{AvgRC} = \) Average ratio of sub-tasks that are completed incorrectly or correctly  
\(\text{AvgTC} = \) Average time for a complete sub-task, correct or incorrect

Scale: Numeric  
Type of Scale: Ratio

Unit (type, description): sub-tasks effectiveness/time, sub-task completeness effectiveness per time unit (usually seconds or minutes).

As final output of these activities we get the QinU metrics specification document.
4.1.6. Design QinU Evaluation

Once the metric specifications have been completed, we can design an indicator for each attribute and calculable concept of the requirements tree. Taking into account the C-INCAMI framework (recall evaluation package in Fig. 2), there are two indicators types: elementary and global indicators. The Elementary Indicators evaluate attributes and map to a new scale based on the metric’s measures. The new scale is interpreted to analyze the level of satisfaction reached by each attribute. On the other hand, the Global Indicators (also called Partial Indicator if it evaluates a sub-characteristic) evaluate characteristics in a concept model and serve to analyze the level of global (or partial) satisfaction achieved.

Following the activities flow depicted in Fig. 11, for each attribute of the requirements tree the indicators expert should specify an elementary indicator by means the next iterative activities: Establish the Elementary Model, Establish the Calculation Method (optional) and Identify the Scale.

The first activity (Establish the Elementary Model) involves establishing a function to map between measure and indicator values, and define the associated decision criteria or acceptability levels (see sub-section 3.1.5). In our case, the indicators expert and the evaluation requester defined three acceptability ranges in the indicator percentage scale, namely: a value within 70-90 (a marginal –yellow- range) indicates a need for improvement actions; a value within 0-70 (an unsatisfactory –red- range) means changes must take place with high priority; a score within 90-100 indicates a satisfactory level –green- for the analyzed attribute. The acceptance levels in this case study were the same for all indicators, both elementary and partial/global, but could be different depending on the needs of the evaluation requester.

Note that the Establish the Calculation Method activity is not mandatory because usually the model used is an easily interpreted function. In other cases, the calculation method should be specified.

Regarding to the partial/global indicators, these are specified in a similar way to the elementary indicators, as we can see in Fig. 11. For example, in the JIRA case study, a global (linear additive) aggregation model to calculate the requirements tree was selected, with equal weights for their elements. This approach was used given that it was an exploratory study. Different weights would be assigned based on the requester’s objectives to reflect the different levels of importance relative to one another. For example, for effectiveness in use, some organizations may weigh mistakes or correctness more heavily than completeness depending on the domain. A pharmacy or accounting application for example may have a higher weighting for accuracy.

The final output for the QinU evaluation design is an indicators specification document for quality in use. An artifact hierarchy (i.e. the informational view) of the indicators specification document is shown in Fig. 12.
4.1.7. Design Preliminary Analysis

Taking into account the underlying SIQinU improvement objective, the specific QinU requirements for the project, the task, the metrics and indicators specifications as well as the data properties with regard to the scale, a preliminary analysis design should be drawn by the Analysis Designer role. This activity involves deciding on the allowable mathematical and statistical methods and techniques for analysis regarding the scale type, dataset properties, etc., the suitable tools for the kinds of analysis at hand, the presentation and visualization mechanisms, and so forth.

4.2. Phase II: Perform QinU Evaluation and Analysis

This phase involves the basic activities to accomplish the first purpose of the SIQinU strategy, namely: understand the current QinU satisfaction level of the actual WebApp in use. To achieve this, the next four activities (see Fig. 13) should be performed.

4.2.1. Collect Data

Taking into account the tasks specification, the log files with the user usage data are analyzed and the relevant data is filtered and organized to facilitate the measurement for each attribute in the next activity. Note that a tool can be used to process the log file for extracting the relevant data from user records.

4.2.2. Quantify Attributes

After collecting the data, we derive measurement values for each attribute in the QinU requirements.
tree. The values are obtained based on the measurement or calculation methods specified in QinU Metrics Specification according to the Design QinU Measurement activity (Fig. 10).

4.2.3. Calculate Indicators

Taking into account the measures (values) and the indicators specification, the indicators values are calculated by the indicators calculator. The global indicator value ultimately represents the degree of satisfaction in meeting the stated information need for a concrete entity, for a given purpose and user viewpoint. Within the Calculate Indicators activity, first, the Calculate Elementary Indicator activity should be performed for each attribute of the requirements tree, and then, using these indicators’ values and the specified partial or global model, the partial and global indicators are calculated by performing the Calculate Partial/Global Indicator activity for each calculable concept. Table 6, columns 2 and 3, shows each element of the QinU non-functional requirements tree evaluated at task level with Elementary, Partial and Global Indicators for the current version of JIRA (i.e., v.1).

Table 6. QinU evaluation of JIRA, both before (v.1) and after implementing improvements (v.1.1). EI stands for Elementary Indicator; P/GI stands for Partial/Global Indicator.

<table>
<thead>
<tr>
<th>Characteristics and Attributes</th>
<th>JIRA v.1</th>
<th>JIRA v.1.1</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1. Actual Usability</td>
<td>84.6%</td>
<td>84.5%</td>
</tr>
<tr>
<td>1.1.1. Effectiveness in use</td>
<td>72.2%</td>
<td>86.7%</td>
</tr>
<tr>
<td>1.1.1. Sub-Task Correctness</td>
<td>86.4%</td>
<td>91.2%</td>
</tr>
<tr>
<td>1.1.1. Sub-Task Completeness</td>
<td>87.9%</td>
<td>95.5%</td>
</tr>
<tr>
<td>1.1.1. Task Successfulness</td>
<td>41.5%</td>
<td>72.7%</td>
</tr>
<tr>
<td>1.1.2. Efficiency in use</td>
<td>49.5%</td>
<td>51.9%</td>
</tr>
<tr>
<td>1.1.2. Sub-Task Correctness Efficiency</td>
<td>37.4%</td>
<td>62.3%</td>
</tr>
<tr>
<td>1.1.2. Sub-Task Completeness Efficiency</td>
<td>77.5%</td>
<td>67.7%</td>
</tr>
<tr>
<td>1.1.2. Task Successfulness Efficiency</td>
<td>91.1%</td>
<td>56.8%</td>
</tr>
<tr>
<td>1.1.3. Learnability in use</td>
<td>37.3%</td>
<td>72.8%</td>
</tr>
<tr>
<td>1.1.3.1. Sub-Task Correctness Learnability</td>
<td>78.8%</td>
<td>75.1%</td>
</tr>
<tr>
<td>1.1.3.2. Sub-Task Completeness Learnability</td>
<td>94.4%</td>
<td>77.3%</td>
</tr>
<tr>
<td>1.1.3.3. Task Successfulness Learnability</td>
<td>66.7%</td>
<td>62.5%</td>
</tr>
</tbody>
</table>

4.2.4. Conduct Preliminary Analysis

After calculating indicators at all levels i.e., elementary, partial and global, a preliminary analysis on the current JIRA WebApp task is conducted by the Data Analyzer role. Basically, it follows the analysis design (produced in the Design Preliminary Analysis activity, in Ph. I), i.e. implementing the designed procedures and planned tools, and storing results according to established formats in order to produce the preliminary analysis report. The analysis allows us to understand how the application performs overall (globally), and also with respect to each particular attribute for each part of the task (i.e., at sub-task levels) being executed by a given user group type.

In the case study, the preliminary analysis was conducted for the above mentioned task, its five sub-tasks and their associated screens in JIRA, for the beginner user type. This allows the recommender to gauge more specifically where users had difficulty, e.g. low task successfulness, low completion rate in using the application, among others.

4.3. Phase III: Derive/Specify Requirements and Evaluation Criteria for EQ

Taking into account the preliminary analysis report yielded in Phase II for QinU and the requirements tree defined in Phase I, in Phase III a requirements tree for EQ is derived. This requirements tree is tailored considering those product features that would need improvement with potential positive impact in QinU, mainly for those problems found in Phase II. In this phase, metrics and indicators are specified in order to evaluate the WebApp through its inspection involving three main activities (see Fig. 14), namely: Select EQ Concept Model, Design EQ Measurement and Design EQ Evaluation. Note that these activities are similar to Phase I activities (recall Fig. 7), but now from the EQ viewpoint.
Given the preliminary analysis report performed in Phase II which may have reported potential problems of the actual WebApp, EQ characteristics and attributes possibly related to those QinU dimensions are identified, resulting then in a new requirements tree. The activities to be performed by the NFR manager are: Select a Model and Edit the Model.

In the case study, in this activity, the requirements tree for the EQ viewpoint was established using 2Q2U (as in Phase I), instantiating the characteristics operability and information quality to determine possible effects on effectiveness in use, efficiency in use and learnability in use. Those EQ characteristics and attributes that are possibly related to those QinU dimensions with potential problems have been instantiated resulting in the requirements tree shown on the left side of Table 7.

**4.3.2. Design EQ Measurement**

Following Fig. 14, once the EQ model was instantiated in a requirements tree, the measurement should be designed to produce the metric specifications to perform Phase IV. As can be seen in Fig. 15, this activity is similar to designing the QinU measurement (recall Fig. 10 in Phase I) and is performed by the metrics expert, but now the process is executed for EQ attributes. In addition, next to Identify a Metric for an attribute from the repository and its related metrics and attributes (if the selected metric is an indirect metric), the Select a Tool activity can be performed to choose a tool that automates the metric method.

In the case study, for each attribute from the EQ requirements tree shown in Table 7 a metric was identified by the metrics expert. For instance, for the attribute Navigability feedback Completeness (coded 1.1.1.1) the metric is:
Indirect Metric: Task Navigability Feedback Completeness (TNFC)

Objective: Calculate the average of completeness considering the navigational feedback completeness level for all sub-task screens for the given task.

Calculation Method (Formula):

\[
\text{TNFC} = \frac{1}{n} \left( \frac{1}{m} \sum_{j=1}^{n} \frac{\text{NFC}_j}{m} \right)
\]

for \( j = 1 \) to \( n \), where \( n \) is the number of sub-tasks of the given task

for \( i = 1 \) to \( m \), where \( m \) is the number of screens for sub-task \( j \)

Interpretation: \( 0 \leq \text{TNFC} \leq 3 \), more is better.

Scale: Numeric

Unit: Completeness level (Note: this metric can be converted to percentage unit, i.e. TNFC /0.03).

As this is an indirect metric, related metric and attribute were identified:

Attribute: Screen Navigation Feedback

Direct Metric: Navigation Feedback Completeness Level (NFC)

Objective: Determine the screen meet the criteria for navigation feedback completeness. Note: This metric is similar to the breadcrumb or path capability available in many WebApps.

Measurement Method (Type: Objective): The screen is inspected to determine the rating (0-3), where evaluators through inspection observe the availability of the previous (backward), current, and next location (forward) mechanism. Screens should support the completeness of this navigation feedback.

Scale: Numerical

Unit: Completeness level.

4.3.3. Design EQ Evaluation

Similar to Phase I, one indicator per each attribute and concept of the EQ requirements tree should be identified by the indicators expert. In our case, when elementary, partial and global indicators were designed, new acceptability ranges (DecisionCriterion in Fig. 2) were agreed between the evaluation requester and the indicators expert. The three acceptability ranges in the indicator percentage scale were:

- a value within 60-80 (a marginal –yellow- range) indicates a need for improvement actions;
- a value within 0-60 (an unsatisfactory –red- range) means changes must take place with high priority;
- a score within 80-100 indicates a satisfactory level –green- for the analyzed attribute. Note that this indicator mapping does not necessarily have to be the same as the QinU mapping (e.g. may have different range thresholds), but rather should meet the information need and goals of the evaluation requester.

4.4. Phase IV: Perform EQ Evaluation and Analysis

Based on metric and indicator specifications obtained in Phase III, the measurement and evaluation of the EQ requirements and the analysis of the current product situation are performed. This phase is similar to Phase II, but now for EQ. The involved activities are shown in Fig. 16. Note the similarity with Phase II (recall Fig. 13), but for Phase IV the Collect Data activity is not performed, i.e., it is just carried out in Phase II to obtain user data usage from log files in a non-intrusive way. In this phase, the measurement and evaluation activities are done by inspection.

Figure 16. Overview for Perform EQ Evaluation and Analysis phase (Ph. IV).
Once each attribute is measured by the data collector in Quantify Attributes activity and all indicators are calculated in Calculate Indicators activity, the data analyzer role should Conduct EQ Analysis. The latter activity generates an EQ analysis report with information that allows us to identify, for instance, parts of the application that need improvement from the EQ viewpoint. In Table 7 (columns 2 and 3) we can see the EQ evaluation results from JIRA (v.1) case study. Note, for example, that some attributes such as Error prevention (coded 1.2.2.1.) and Context-sensitive help availability (coded 1.1.2.1) need improvement with high priority. Also we can observe that for some elementary indicators (attributes), no improvement is needed, e.g. Stability of main control (coded 1.2.1.2).

Table 7. EQ evaluation of JIRA, both before (v.1) and after (v.1.1) implementing improvements.

<table>
<thead>
<tr>
<th>Characteristics and Attributes</th>
<th>JIRA v.1</th>
<th>JIRA v.1.1</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Operability</td>
<td>26%</td>
<td>30%</td>
</tr>
<tr>
<td>1. Learnability</td>
<td>19%</td>
<td>19%</td>
</tr>
<tr>
<td>1.1. Feedback suitability</td>
<td>33%</td>
<td>33%</td>
</tr>
<tr>
<td>1.1.1. Navigability feedback completeness</td>
<td>33%</td>
<td>33%</td>
</tr>
<tr>
<td>1.1.1.2. Task progress feedback appropriateness</td>
<td>30%</td>
<td>30%</td>
</tr>
<tr>
<td>1.1.3. Entry form feedback awareness</td>
<td>20%</td>
<td>20%</td>
</tr>
<tr>
<td>1.2. Helpfulness</td>
<td>35%</td>
<td>35%</td>
</tr>
<tr>
<td>1.2.1. Context-sensitive help availability</td>
<td>80%</td>
<td>80%</td>
</tr>
<tr>
<td>1.2.2. Help completeness</td>
<td>10%</td>
<td>10%</td>
</tr>
<tr>
<td>1.2. Ease of use</td>
<td>24%</td>
<td>24%</td>
</tr>
<tr>
<td>1.2.1. Controllability</td>
<td>10%</td>
<td>10%</td>
</tr>
<tr>
<td>1.2.1.1. Permanence of main controls</td>
<td>60%</td>
<td>60%</td>
</tr>
<tr>
<td>1.2.1.2. Stability of main controls</td>
<td>100%</td>
<td>100%</td>
</tr>
<tr>
<td>1.2.2. Error management</td>
<td>0%</td>
<td>0%</td>
</tr>
<tr>
<td>1.2.2.1. Error prevention</td>
<td>0%</td>
<td>0%</td>
</tr>
<tr>
<td>1.2.3. Data entry ease</td>
<td>65%</td>
<td>65%</td>
</tr>
<tr>
<td>1.2.4. Mandatory entry</td>
<td>10%</td>
<td>10%</td>
</tr>
<tr>
<td>2. Information quality</td>
<td>75%</td>
<td>75%</td>
</tr>
<tr>
<td>2.1. Information suitability</td>
<td>33%</td>
<td>33%</td>
</tr>
<tr>
<td>2.1. Consistency</td>
<td>0%</td>
<td>0%</td>
</tr>
<tr>
<td>2.1.2. Information coverage</td>
<td>8%</td>
<td>8%</td>
</tr>
<tr>
<td>2.1.2.1. Appropriateness</td>
<td>50%</td>
<td>50%</td>
</tr>
<tr>
<td>2.1.2.2. Completeness</td>
<td>50%</td>
<td>50%</td>
</tr>
</tbody>
</table>

4.5. Phase V: Recommend, Perform Improvement Actions, and Re-evaluate EQ

Considering the previous EQ analysis report generated in Conduct EQ Analysis activity (Phase IV), we make recommendations to improve the application for those EQ attributes that needed improvement. After the recommended changes were completed in the current WebApp and a new version generated, we re-evaluate the EQ to determine the improvement gain between both product versions. The activities for this phase are shown in table 8 and described below.

4.5.1. Recommend Improvement Actions

Based on the EQ analysis report generated in Phase IV, the Recommend Improvement Actions activity is carried out by the recommender in order to produce a recommendations report. This document has a set of recommendations for which attributes of the WebApp can be improved. For instance, a ranking of elementary indicators scored from weaker –i.e. that fell in the unsatisfactory acceptability level-, to stronger, but which did not fall in the satisfactory or green level can be listed. Then, the evaluation requester can prioritize recommendations made for improvement action. Considering the case study, in this activity, some of the recommendations listed in the recommendations report were the following:

- For increasing the satisfaction level of Defaults attribute (1.2.3.1): Change fields to have default and make mandatory because they are critical defect description correctness and completeness.
- For increasing the satisfaction level of Error Prevention attribute (1.2.2.1): Add context sensitive help and eliminate non valid platform combinations.

**Table 8.** Process template in which information and views are documented for the **Recommend, Perform Improvement Actions, and re-evaluate EQ** activities

<table>
<thead>
<tr>
<th>Activity</th>
<th>Code (in Fig. 1): Ph. V</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Objective:</strong> improve the current application version and determine the improvement gain from the EQ standpoint.</td>
<td></td>
</tr>
<tr>
<td><strong>Description:</strong> Considering the EQ analysis report generated in Conduct EQ Analysis activity (Phase IV), the recommender makes recommendations to improve the current application, and the Maintenance Project Manager produces an improvement plan to enhance the current WebApp. After the recommended changes were implemented by the developer and a new version generated, a re-evaluation of the EQ is performed to determine the improvement gain between both application versions.</td>
<td></td>
</tr>
</tbody>
</table>

**Sub-activities:**
- Recommend Improvement Actions
- Design Improvement Actions
- Perform Improvement Actions
- Evaluate Improvement Gain (i.e. in Phase IV)

**Involved Roles:**
- Recommender
- Maintenance Project Manager
- Developer
- Data Analyzer

**Input Artifacts:**
- EQ Analysis Report
- Current Application Version

**Output Artifacts:**
- EQ Recommendations Report
- Improvement Plan
- New Application Version
- New EQ Analysis Report (from Phase IV)

**Pre-conditions:** there are EQ attributes with low level of satisfaction met, so improvement actions are needed to enhance the current software/web application version.

**Post-conditions:** the Phase V finishes when the EQ attributes met the agreed satisfaction level.

### 4.5.2. Design Improvement Actions

Based on the previous recommendations report, the maintenance project manager produces an improvement plan indicating how to actually change the application. This ‘how’ implies planning methods and techniques to be used to actually accomplish the improvement actions in the next activity (Perform Improvement Actions). Methods and techniques for changing the WebApp can range from parameterized reconfigurations, code restructuring, refactoring (as made in [24]) to architectural redesign. The eventual method employed depends on the scope of the improvement recommendation as well as the resources of the evaluation requester and the desired effect. The expected effect may include an application easier to operate and learn, faster to run, more secure, among many other aspects.

For example, taking into account the two improvement recommendations listed in the above activity, the improvement plan included:

- **Recommendation:** Add context sensitive help to improve the Error Prevention (1.2.2.1) attribute.  
  **Action taken:** Defect steps moved to next screen on Add Detail info, with help, examples shown to aid user.

- **Recommendation:** Eliminate non valid platform combinations to improve Error Prevention (1.2.2.1).  
  **Action taken:** Help provided and invalid combinations not allowed.
• Recommendation: Change fields to have default and make mandatory because they are critical defect description correctness and completeness to improve the Defaults (1.2.3.1) attribute. 

Action taken: Done where possible.

4.5.3. Perform Improvement Actions

With the improvement plan the developer of the WebApp performs changes accordingly, resulting in a new application version (see the activity flow in table 8). The ABC developer of our JIRA case study made some of the recommended changes, including those shown above, resulting in a new product version termed JIRA v.1.1. This new JIRA version had many other improvements not shown, one of which was the reduction of workload through eliminating one sub-task and moving more related items together to make the overall task design more efficient. Thus, JIRA v.1.1 only has 4 sub-tasks, instead of 5. Because JIRA does not give access to its source code, the developer could not enact all the changes that were recommended; so only some improvements were made. Rather, through changing its configuration, they were able to perform most of the changes. Note that some recommended changes that could not be made were due to the application under study, and not due to SIQinU.

4.5.4. Evaluate Improvement Gain

Once changes were made, the WebApp can be re-evaluated by inspection to determine which attributes have been improved, which have not, and get a score which can be compared to the outcomes of Phase IV. The activities involved are: Quantify Attributes, Calculate Indicators and Conduct EQ Analysis. The output is a new EQ analysis report in which the changes made between the WebApp versions are compared to determine the improvement gain.

In Table 7 (columns 4 and 5) we can see the results obtained when JIRA v.1.1 was evaluated from EQ viewpoint. As can be seen from comparing the 2 evaluations (columns 2 and 3 with 4 and 5), the overall Partial Indicator for Ease of use improved significantly from 34% to 61% with improvements in many of the individual attributes and an overall improvement in the Global Indicator from 38% to 74%. The next and final Phase examines how these improvements in EQ affect QinU in a real context of use.

4.6. Phase VI: Re-evaluate Quality in Use and Analyze Improvement Actions

Once the new application version (generated in Phase V, particularly in Perform Improvement Actions activity) has been used by the same user group type in its same real context of use that the previous version, then, we are able to perform the QinU re-evaluation (in a similar way to Phase II, recall Fig. 13) to determine if what was improved from the EQ viewpoint had a positive quality-in-use effect using the identical tasks.

The activities involved are the same that in Phase II, namely: Collect Data, Quantify Attributes and Calculate Indicators. Finally Conduct Improvement Actions Analysis activity (see Fig. 1) is performed to determine the improvement gain. These activities are described below according to the JIRA case study.

4.6.1. Collect Data, Quantify Attributes and Calculate Indicators

When JIRA v.1.1 was evaluated from the EQ viewpoint, and its satisfaction level was achieved, then this new release was used by real users in the same real context of use as JIRA v.1. After 12 weeks (the same time period), we performed the QinU re-evaluation (using the same non-functional requirements, metrics and indicators designed in Phase I) to determine if what was improved from the EQ viewpoint had a positive quality-in-use effect with the same task (Entering a new defect). Following the SIQinU activities involved in Phase VI, we collect the data (i.e., the data collector using the same parser tool), quantify the attributes and calculate all the indicators in a similar way as in Phase II. In table 6, columns 4 and 5, we show the evaluation results for JIRA v.1.1.

4.6.2. Conduct Improvement Actions Analysis

Once all indicators were calculated, data analyzer looks at each particular attribute’s change for each part of the task being executed by the user group noting the difference to calculate quantified improvement between both WebApp versions from the QinU viewpoint. Table 9 shows the attributes and
indicator values regarding the QinU requirements tree depicted in Fig. 9 for JIRA v.1 and JIRA v.1.1 with the right most columns showing the change.

Table 9. QinU attributes satisfaction level for JIRA v.1 and JIRA v.1.1 with improvements.

<table>
<thead>
<tr>
<th>Attributes</th>
<th>v.1</th>
<th>v.1.1</th>
<th>Change</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1.1.1.1</td>
<td>86.4%</td>
<td>90.2%</td>
<td>5.5%</td>
</tr>
<tr>
<td>1.1.1.1.2</td>
<td>87.9%</td>
<td>95.3%</td>
<td>8.6%</td>
</tr>
<tr>
<td>1.1.1.1.5</td>
<td>45.5%</td>
<td>72.2%</td>
<td>27.1%</td>
</tr>
<tr>
<td>1.1.1.2.1</td>
<td>37.8%</td>
<td>45.5%</td>
<td>6.9%</td>
</tr>
<tr>
<td>1.1.1.2.2</td>
<td>37.5%</td>
<td>47.3%</td>
<td>9.1%</td>
</tr>
<tr>
<td>1.1.1.2.3</td>
<td>15.1%</td>
<td>39.0%</td>
<td>23.7%</td>
</tr>
<tr>
<td>1.1.1.2.5</td>
<td>32.4%</td>
<td>55.1%</td>
<td>22.7%</td>
</tr>
<tr>
<td>1.1.1.3.2</td>
<td>45.6%</td>
<td>77.2%</td>
<td>31.6%</td>
</tr>
<tr>
<td>1.1.1.3.3</td>
<td>44.2%</td>
<td>74.0%</td>
<td>30.7%</td>
</tr>
</tbody>
</table>

As we can see from table 9, all attributes noted improvement with the exception of Task Successfulness Learnability and Sub-task Correctness Learnability. A possible explanation for this is that due to metric design, with data collected over a 12 week period that the learning process did not improve as much as expected. Remembering our earlier comments that temporal analysis over time is important as user behavior can change over time, these beginning users possibly were not able to ramp up their learning during this time period. And, on the other hand, if the case study had been longer, we may have seen different behavior and hence measurements. However, their negative change was small compared to the positive changes in the other attributes resulting in an overall average change of attributes evaluation of 13.7%. While the indicators show that most of the attributes in JIRA v.1.1 still need some or significant improvement, there has been notable improvement from JIRA v.1. Again, as previously mentioned, due to the limitations in changing code in JIRA, configurations changes enabled many changes and improvements, but not all, it is possible that JIRA v.1.1 could be even better if more changes could be made. In other instances, depending on the software under evaluation, the tools/methods available to the maintenance project manager, and the time/resources available, more improvements could be made.

5. RELATED WORK

In recent years, the ISO/IEC has worked on a new project, called SQuaRE (Software product Quality Requirements and Evaluation), that proposes integrating many ISO standards related to quality models, M&E processes, etc. Although ISO 25000 [15] has guidelines for the use of the new series of standards, the documents aimed at specifying M&E processes are not issued yet. So the standards for software measurement process (ISO 15939 [14]), and the process for evaluators (ISO 14598-5 [13]) are still in effect and considered the most recent. Taking into account these standards, the process for measurement has two core activities, namely: Plan the measurement process, and Perform the measurement process [14]. The evaluation process involves five activities: Establishment of evaluation requirements, Specification of the evaluation, Design of the evaluation, Execution of the evaluation plan, and Conclusion of the evaluation [13]. We have observed that there is so far no single ISO standard that specifies in an integrated way the M&E process and approach as a whole.

Other work, worthy to mention, is the CMMI (Capability Maturity Model Integration) [7] de facto standard, which provides support for process areas such as Measurement and Analysis, among others. It aligns information needs and measurement objectives with a focus on goal-oriented measurement – following to some extent the GQM (Goal Question Metric) [2] approach and the [14] measurement process. Although CMMI specifies (specific/generic) practices to accomplish the given process area goals, a process model itself is not defined. To a certain extent, it represents practices (i.e. actions/activities) without explicitly establishing sequences, parallelisms, control points, etc. Some specific practices for Measurement and Analysis are for example: Establish measurement objectives, Specify measures, Obtain measurement data, and Analyze measurement data. However, a clear distinction between M&E processes is missing in addition to lacking a robust conceptual base for its terms.

Regarding improvement strategies for evaluation of WebApps, we can consider the work in [24], where authors present an approach for incremental EQ improvement whereby the results from EQ
evaluation were used to make changes and improvements in a WebApp through WMR (Web Model Refactoring) in a systematic way. But although a set of activities is considered, the underlying process is neither well defined nor modeled regarding process views. On the other hand, in [8] a systematic approach to specify, measure and evaluate QinU was discussed. However, the process used is not explicitly shown, and the outcomes were used just for understanding the current situation of the QinU for an e-learning application, without proposing any improvement strategy.

In addition, it is worthy to mention the GQM+Strategies [3] –which is based on the GQM approach-, as an integrated strategy that allows defining and assessing measurement goals at different organization levels, but it does not specify formal process views to conduct the evaluation and improvement lifecycle as we have shown as an integral component of SiQinU. Also, since issued, the GQM model was at different moments enlarged with proposals of processes and methods. However, [17] pointed out GQM is not intended to define metrics at a level of detail suitable to ensure that they are trustworthy, in particular, whether or not they are repeatable. Moreover, an interesting GQM enhancement, which considers indicators has recently been issued as a technical report [12]. This approach uses both the balanced scorecard and the Goal-Question-Indicator-Measurement methods, in order to purposely derive the required enterprise goal-oriented indicators and metrics. It is a more robust approach for specifying enterprise-wide information needs and deriving goals and sub-goals, and then operationalizing questions with associated indicators and metrics. However, this approach is not based on a sound ontological conceptualization of metrics and indicators as in this research. Furthermore the terms “measure” and “indicator” are sometimes used ambiguously, which inadvertently can result in datasets and metadata recorded inconsistently, and so it cannot assure that measurement values (and the associated metadata like metric version, scale, scale type, unit, measurement method, and so forth) are trustworthy, consistent and repeatable for further analysis among projects.

Finally, in [25] authors propose the CQA approach, consisting of a methodology (CQA-Meth) and a tool that implements it (CQA-Tool). They have applied this approach in the evaluation of the quality of UML models such as use cases, class and statechart diagrams. Also authors have connected CQA-Tool to the different tools needed to assess the quality of models. CQA-Tool, apart from implementing the methodology, provides the capacity for building a catalogue of evaluation techniques that integrates the evaluation techniques (e.g. metrics, checklists, modeling conventions, guidelines, etc.), which are available for each software artifact. Compared with our strategies, the CQA approach for instance lacks an explicit conceptual framework from a terminological base. On the other hand, other related work in which authors try to integrate strategic management, process improvement and quantitative measurement for managing the competitiveness of software engineering organizations is documented in [11]. In this work, a process template to specify activities from different views is considered. However, the integration of the three capabilities as made in GOCAME and SiQinU is not explicit and formalized.

6. CONCLUDING REMARKS

Ultimately, the main contribution of this paper is, SiQinU, an integrated specific-purpose strategy –i.e. for understanding and improving QinU-, whose rationale is based on well-defined M&E processes, founded on a M&E conceptual framework backed up by an ontological base, and supported by methods and tools.

In this paper we have specified the process of the SiQinU strategy modeled stressing the functional, informational, organizational and behavioral views. Moreover, to illustrate the SiQinU process, excerpts from a JIRA case study were used where real users were employed to collect data and ultimately prove the usefulness of the strategy in improving the application in a process-oriented systematic means.

We have also shown SiQinU, to be a derivation of GOCAME, based on three foundations viz. the process, the conceptual framework, and methods/tools. Relying on the GOCAME foundation, SiQinU has been defined as a systematic approach with the appropriate recorded metadata of concrete projects’ information needs, context properties, attributes, metrics and indicators. This ensures that collected data are repeatable and comparable among the organization’s projects. Otherwise, analysis, comparisons and recommendations can be made in a less robust, non-consistent, or even incorrect way.

SiQinU, although reusing the general principles of GOCAME, is a specific-purpose goal-oriented strategy with specific activities and methods that are not taken into account in GOCAME. Where
GOCAME is a multi-purpose strategy with general purposes such as “understand”, “predict”, “improve”, “control”, etc., SIQinU objectives are targeted to “understand” and ultimately “improve”. In addition, as discussed in sub-section 3.4, SIQinU was specifically designed to evaluate QinU and EQ for WebApps, from the “do goals” perspective rather than from the “be goals”.

As future work we are planning to extend SIQinU to include processes and methods not only to gather data in a non-intrusive way (as currently it does) but also to gather data using more traditional intrusive methods such as video recording, observations and questionnaires. This could not only help to add robustness to Phase III particularly in the derivation process from QinU problems to EQ attributes, but also supplement the analysis in Phase II, and VI.

Acknowledgments. This work and line of research is supported by the PAE 2007 PICT 2188 project at UNLPam, from the Science and Technology Agency, Argentina.

REFERENCES


