Journal of Web Engineering, Vol. 10, No.4 (2011) 316-352

USING WEB QUALITY MODELS AND A STRATEGY FOR PURPOSE-ORIENTED EVALUATIONS

LUIS OLSINA[1], PHILIP LEW[2], ALEXANDER DIESER[1], BELEN RIVERA[1]

[1]*GIDIS_Web, School of Engineering, National University of La Pampa*
General Pico, La Pampa 6360, Argentina
olsinal@ing.unlpam.edu.ar, *alexander.dieser@gmail.com*, *riveramb@ing.unlpam.edu.ar*

[2]*School of Computer Science and Engineering, Beihang University*
Beijing, China
philiplew@gmail.com

Received November 3, 2011
Revised December 21, 2011

Web applications and their quality evaluation has been the subject of abundant research. However, instantiated models have been used mostly for the purpose of understanding, rather than improving. In this paper, we propose utilizing a quality modeling framework together with a non-intrusive evaluation strategy to instantiate quality models with the specific purpose of not only to model and understand a Web application in-use, but also to improve it. Starting with a quality modeling framework, our approach instantiates models for both quality in use and external quality, resulting in a requirements tree for both followed by evaluation and then combined with a mechanism to develop relationships between them. Interpreting these relationships viz. 'depends on', and 'influences', in alignment with the ISO 25010 quality life cycle model, is the basis for continuous improvement. This is illustrated with a case study showing the underlying strategy from model instantiation to application improvement.

Key words: Quality models, Evaluation strategy, SIQinU strategy, Quality in use, Actual usability, External quality, Improvement.
Communicated by: O. Diaz & S. Auer

1 Introduction

Today's Web applications (WebApps) containing complex business logic and sometimes critical to operating the business, are now requiring increased focus on understanding and improving their quality. Consequently, evaluating their quality has taken on increased importance. For instance, to evaluate quality for newer generations of WebApps requires understanding in detail how they are different from older generations or from conventional software applications as user requirements, expectations, and behavior can be somewhat different. Over the years, various researchers have examined the differences between WebApps and conventional software/web applications ([22], among others). In addition, the recent ISO 25010/25012 standards [11, 13] have officially been issued which assist practitioners and researchers in specifying and evaluating system and data quality requirements.

Extensions of these models for WebApps have also been proposed in [18].

In the end, the goal is to develop quality WebApps that meet the needs of their users in real contexts of use. One of the first steps in evaluation is to define non-functional requirements, usually, by means of quality models and also by a quality framework to structure relationships among them. Quality models can represent different entities categories such as product, system, system in use, among others as resource and process. The (software/web) products are entities at early phases of process life cycle (e.g., source code, UML diagrams, textual documents, etc); information systems are executable products in a specified context (e.g. an e-commerce WebApp in a testing environment), which can include hardware and software together; and information systems in use are the aforementioned systems but operated by real users in real contexts of use, i.e. while users perform the daily application tasks in a real environment and infrastructure.

Quality models usually categorize and specify product, system or system-in-use quality into characteristics that are further subdivided into sub-characteristics and attributes (or properties). Further, the attributes can be measured by metrics and evaluated by indicators [25]. The quoted ISO standards and other researchers ([3, 19] among others) have proposed with the objective of evaluation, quality models and different views of quality such as external quality (EQ) and quality in use (QinU). The EQ view, which is specified by a quality model (e.g. 8 characteristics in ISO 25010, issued in March, 2011), can be measured and evaluated by dynamic attributes of the running code, i.e. when the module or full software/web application is executed in a computer or network system simulating the actual environment as close as possible. The QinU view, which is specified by a quality model (e.g. 5 characteristics in the ISO 25010), can be measured and evaluated by the extent to which the system or WebApp in-use meets specific user's needs when performing the application tasks in the actual, specific context of use. Moreover, ISO 25010 states that the relationship *influences* exists between EQ and QinU views, or vice versa, *depends on* (also called *is determined by*) between QinU and EQ. Nevertheless, ISO 25010 is intended as a general framework to be adapted based on a specific information need and context, i.e. for evaluating WebApps. In addition, some of ISO model concepts, while founded strongly in theory, are difficult to realize in a real situation particularly when it comes to evaluating and improving QinU.

Assuming that the main goal in evaluating a system quality is to ultimately improve it, a key issue is how to relate QinU evaluation results to properties intrinsic to the WebApp itself in order to make improvements. In modeling terms, QinU characteristics and attributes need to be related to EQ characteristics and attributes. And after recommended improvement actions based on EQ evaluation were performed to the WebApp, does the software's new version have a positive impact on its QinU?

To answer this question for WebApps (for both system and system-in-use entity categories), we have developed a specific approach based on two main pillars, namely: i) a *quality modeling framework* which includes the EQ and QinU generic models and their relationships; and ii) a *non-intrusive specific-purpose evaluation strategy*, which in turn relies on three capabilities viz. a well-established measurement and evaluation process, a measurement and evaluation conceptual framework utilizing an ontological base, and methods and tools instantiated from both the conceptual framework and process. The former pillar we developed is called 2Q2U (*Quality, Quality in use, actual Usability and User experience*) modeling framework, while the later is called SIQinU (*Strategy for understanding and Improving Quality in Use*).

Regarding the 2Q2U quality modeling framework (where its first version was developed in 2010 [18] and its updated version is introduced in Section 2), we began by first proposing to augment the ISO 25010 standard by including *information quality* as a characteristic of EQ because this is a critical characteristic of WebApps. We further proposed to include *learnability in use* as a characteristic of *actual usability* to account for the learning process and the importance of context of use during learning. 2Q2U relies on the ISO 25010 premise that the relationships *depends on* and *influences* exist between EQ and QinU. Using this premise, we further utilize 2Q2U to instantiate models for both EQ and QinU specifically for the purpose of improving the QinU of a WebApp. However, the ISO 25010 premise that QinU *depends on* EQ and in turn EQ *influences* QinU is very general, not specifically explored, and there is no description on implementing or using these relationships for purposeful evaluations.

Regarding the second pillar, we devised SIQinU (*Strategy for understanding and Improving Quality in Use*) [17], a strategy for improving quality that also uncovers these relationships in a systematic way. Starting with QinU (of a real WebApp-in-use), we design specific user tasks and context of use, and through identifying problems in QinU, we determine EQ attributes that could be related to these QinU weakly performing indicators. Then, after deriving EQ attributes related to the QinU problems, we evaluate EQ and derive a benchmark to be used as a basis to make improvements. Once improvement recommendations are made based on poorly performing EQ measurements (related to the poorly performing QinU indicators), a new version of the WebApp (system) is completed and evaluated again for its EQ to establish a delta from the initial benchmark. Then we re-evaluate QinU of the updated system to determine the improvements resulting in QinU from the improvements made at the EQ level, thus leading to a cyclic and iterative strategy for improvement and development of relationships. These relationships between EQ and QinU are not just for understanding but developed with the primary objective of improving the system with respect to poorly performing QinU attributes. Thus, information regarding *depends on* and *influences*, as depicted in the ISO 25010 quality lifecycle model, are extracted in the process of improving the WebApp. Armed with these relationships, designers can then design/improve software at the EQ level knowing the impact on QinU. Through employing SIQinU, this work, in particular, focuses on improving the *actual usability* of WebApps from an end user viewpoint when executing real tasks in a real context.

Aligning with the ISO 25010 models on the quality life cycle, SIQinU combined with 2Q2U utilizes the ISO premise that if quality can be improved at the EQ point of view, this influences and most likely also improves quality from the QinU viewpoint. The proposed SIQinU strategy utilizes the 2Q2U framework for modeling requirements, non-intrusively collects user behavior data, and provides an integrated means to use quality models in a real context to evaluate EQ and QinU for WebApps with a primary objective of improvement through an evaluation process [2] and methods that are consistent and repeatable.

Consistent and repeatable are paramount characteristics of SIQinU in order to iteratively improve a WebApp. This is gained primarily through a non-functional requirements ontological component of the C-INCAMI (*Contextual-Information Need, Concept model, Attribute, Metric and Indicator*) conceptual framework [20, 25], which enables us to instantiate a project (our project consisted of improving an actual WebApp) that includes defining the information need, purpose, user viewpoint, entity category and concrete entity, context, and so forth. Ultimately, the particular contributions of

this research are:

- A procedure to concretely instantiate quality models based on 2Q2U and C-INCAMI for both QinU and EQ views for the purpose of not only understanding but also improving WebApps.
- A joint use of a software/web system quality modeling framework (2Q2U) together with a specific-purpose strategy (SIQinU) in order to explore the *influences* and *depends on* relationships (from ISO 25010) considering the QinU/EQ/QinU improvement cycle.
- A measurement and evaluation strategy, SIQinU, to guide the improvement with an illustration of the improvement results (including the found relationships between views) through a case study.

As a general contribution of this research, we highlight how the proposed approach based on the two pillars, namely: i) a *quality modeling framework*; and ii) a *measurement an evaluation strategy* (i.e., using the same abovementioned three principles of a well-established process, a measurement and evaluation conceptual framework, and methods and tools), can be adapted for different information needs of an organization and for different entities categories such as resource, process, system, etc., in a flexible yet structured manner.

Following this introduction, Section 2 summarizes quality models and framework (the first pillar of our approach) regarding the updated 2Q2U version in light of recent standards and related literature. Section 3 gives an overview of the six-phased SIQinU strategy (the second pillar). Section 4 demonstrates our procedure for using 2Q2U in conjunction with SIQinU to purposefully instantiate quality models, which in turn is utilized for understanding and improving a real WebApp. In Section 5, we illustrate the instantiated models with the strategy in a case study in the context of evaluating EQ and QinU for a WebApp with the goal of improvement while deriving possible relationships between EQ and QinU. Section 6 reviews related work and delineates opportunities for improvement which are the motivation for this research. Finally, Section 7 draws our main conclusions and outlines future work.

2 The 2Q2U Quality Modeling Framework

As mentioned above, one of the first activities in a measurement and evaluation project is to define the non-functional requirements, basically, using as input a quality model. A quality model, which is targeted for a quality focus and entity category, can be defined as the set of characteristics and sub-characteristics, and their hierarchical relationships that provide the basis for specifying a non-functional requirements structure and its further evaluation or estimation. Quality models can be intended for different entities categories such as resource, process, product, system, system in use, among others, such as project or service. In turn, an entity category can be defined as the object category that is to be characterized by measuring its attributes or properties. Note that in an instantiated quality model, attributes are combined or related accordingly to its (sub-)characteristics. Furthermore, any given measurement and evaluation project can intervene more than an entity category (e.g. a system and a system in use), each with a different quality model. Therefore, establishing relationships among quality models (or views) is necessary as well. A quality modeling

framework can be used to tackle these issues.

So in the next sub-section, we present a general quality modeling framework which connects target entity categories with quality models, and in turn relates quality models by *influences* and *depends on* relationships. Note that the 2Q2U quality modeling framework is a subset of this general framework. Then, in sub-section 2.2, we summarize the first version of the 2Q2U quality framework. Its updated version, in the light of the recently issued ISO 25010 standard and related literature is presented in sub-section 2.3. We also give some details of the included characteristics and sub-characteristics for both EQ and QinU models.

2.1 A Basic Quality Modeling Framework

Fig. 1 shows the schema for a basic quality modeling framework, which is taken and slightly adapted from [27]. Note that target entity categories of quality models and their relationships are also considered in the recent ISO 25010 standard ([11], p. 28, Fig. C.3).

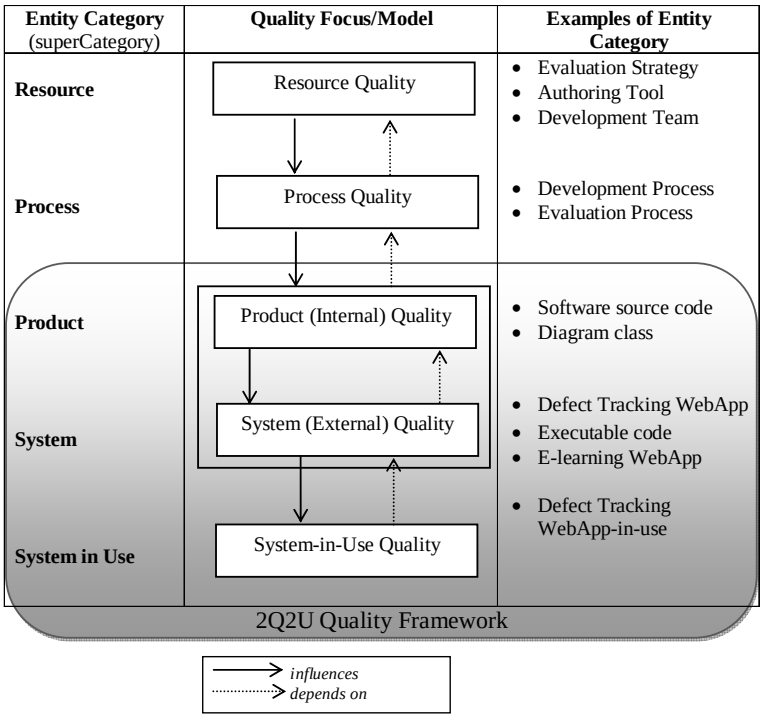


Figure 1: Basic quality modeling framework regarding target entity (super) categories and quality focuses/models. In the bottom the 2Q2U quality modeling framework is highlighted.

In Fig. 1 there are three columns. The first represents the super-category of main entities in a software/web production line; the third column illustrates a bunch of examples of entity categories; and the second one, specifies the quality focus respectively. The quality focus (for a given information need) is in general terms the root characteristic (calculable concept) of a quality model.

In a broader sense, different quality models are intended for different super-categories of entities such as product, system, system in use, among others such as resource and process. In a narrower sense, for a super-category there are many categories of entities. For example, for the resource super-category, we can identify more specific entity categories such as "tool", "strategy", "software team" – which is a human resource-, etc. In turn, for a "strategy" we can identify a "measurement and evaluation strategy", or "development strategy". An entity is a concrete object that belongs to an entity category; for instance, in [29] we have recently evaluated two concrete entities, namely: GQM$^+$Strategies [1] and GOCAME [25] (*Goal-Oriented Context-Aware Measurement and Evaluation*) measurement and evaluation (M&E) strategies. So the "measurement and evaluation strategy" entity category was considered as a resource for a software line of production, which can be employed in quality assurance activities.

Other examples of entity super-categories (see Fig. 1) are system and system in use, as defined in the Introduction Section. Besides, more specific entity categories for system such as "Defect Tracking WebApp", or "e-learning WebApp" can be identified. Particularly, JIRA (www.atlassian.com) is a concrete entity for the former entity category. In Section 5, we will illustrate the instantiated EQ and QinU models for both entity categories viz. "Defect Tracking WebApp" and "Defect Tracking WebApp-in-use", exploring also the relationships between both quality models.

Regarding the relationships between views on quality focuses/models, ISO references the *influences* and *depends on* [11, 14]. Relationships in Fig. 1 indicate that the resource quality *influences* the process quality; the process quality *influences* both product and system quality; and system quality *influences* system-in-use quality. Similarly, the *depends on* relationship can be interpreted; for instance, it can be said from the evaluation point of view that by evaluating the quality in use, feedback for improving the system can be provided, and so forth. In Section 5, we will illustrate the QinU/EQ/QinU improvement cycle by using the SIQinU strategy and tailoring the 2Q2U models.

Last but not least, we note that other entity super-categories such as service and project can be identified, which are not represented in Fig. 1. A service can be placed after the process category, and a project can be seen as an organization resource. Also it can be considered as an entity category that embraces (aggregates) all other entity categories, i.e. resource, process, product, system, and system in use. For example, a given M&E project can include and relate sub-projects for each entity category, where in turn each sub-project deals with non-functional requirements, measurement, and evaluation projects accordingly [2, 25]. Ultimately, since these two entity categories are not depicted in Fig. 1, and other concerns were set aside, we titled this subsection as a "basic quality modeling framework" for a production line of an organization.

Finally, in Fig. 1, we depict a shadowed rectangle labeled 2Q2U quality framework, described in the next section.

2.2 First Version (v1.0) of 2Q2U: An Overview

WebApps, a combination of information (content), functionality and services have become one of the most predominant forms of software implementation and delivery. Due to these evolutions, quality and quality in use have taken on increased importance. Moreover, *user experience*, a relatively new term in

this Web Engineering discipline, combined with *usability*, *information quality*, and *quality in use*, have all recently come to the research forefront especially for WebApps due to the shift in emphasis to satisfying the end user. However, based on draft or issued standards and related literature that we reviewed by the end of 2009, it was difficult for us to understand their relationships and we observed a lack of consensus in meaning as well. Specifically, reviewing ISO standards at that moment as the 25010 draft standard [12] (intended to replace to [14]), the 25012 [13], and other current works by researchers in the field of quality in use, usability and user experience such as Bevan [3, 4] and Hassenzahl [10], among others, it was still not totally clear where characteristics such as information quality, learnability in use, actual usability and user experience fit in regarding quality modeling.

In that context, we developed in early 2010 the first version of the 2Q2U (*Quality, Quality in use, actual Usability and User experience*) modeling framework, as an extension of the ISO 25010 quality models and framework, trying to be as compliant as we could with standards. The rationale of adding or adapting characteristics and concepts was thoroughly discussed in [18], and illustrated with a pharmacy prescription system from the EQ standpoint.

Summarizing the first version of 2Q2U extension, first, we added two characteristics, namely: *information quality* (for the internal/external quality view), and *learnability in use* (for the QinU view). Second, we supplemented two new concepts for the QinU model, namely: *actual usability* and *actual user experience*, to which characteristics and sub-characteristics can be related and new models created in a flexible way.

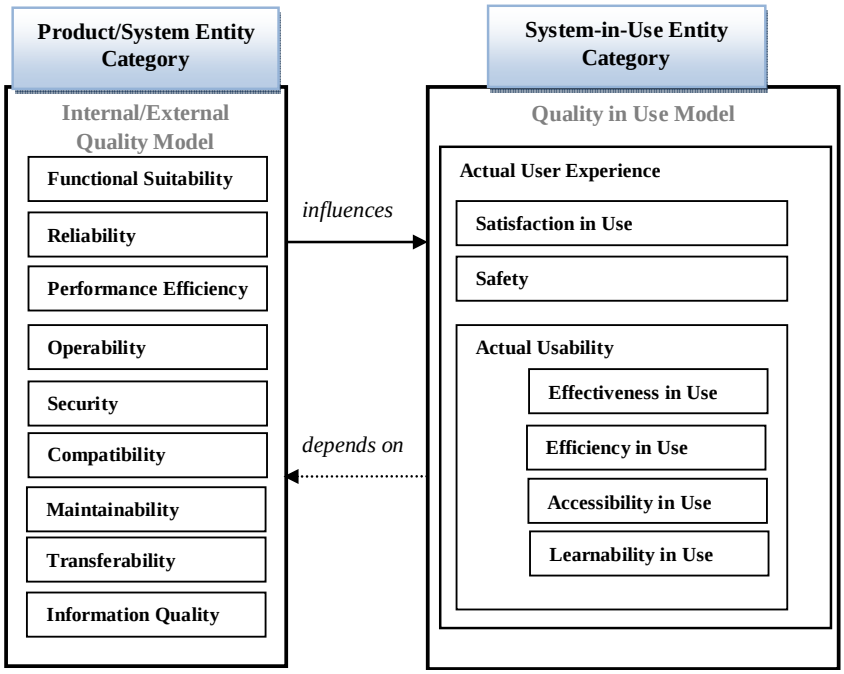


Figure 2: Quality model characteristics (with some sub-characteristics) and relationships between views in the 2Q2U (v1.0) quality modeling framework

Aimed at adding details to the quality focuses shown in Fig. 1 for product/system/system-in-use quality, Fig. 2 depicts the main model characteristics and relationships included in the 2Q2U (v1.0) quality framework. Note that many sub-characteristics of characteristics are not represented for conciseness reasons. In addition, below we define the added characteristics and concepts, which are not part of the quoted ISO standards:

Information quality, defined as the degree to which the software/WebApp provides accurate, suitable, accessible and legally compliant information. This new characteristic becomes part of the internal/EQ model.

Learnability in use, defined as the degree to which specified users can learn efficiently and effectively while achieving specified goals in a specified context of use. This new sub-characteristic becomes part of the *actual usability* characteristic in the *quality in use* model.*Actual Usability*, 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. Note: *Actual usability* is measured and evaluated in a real operational environment where real users perform actual specified tasks. This embraces only performance or 'do' goals in contrast to hedonic or 'be' goals as discussed in [18].

Actual User Experience, defined as the degree to which specified users can achieve actual usability, safety, and satisfaction in use in a specified context of use. Note: *Actual User Experience* is evaluated not only by measures and indicators of user performance that account for 'do' goals –as in *actual usability*-, but also by means of satisfaction instruments to account for 'be' goals (see details in [18]).

It is worthy to remark that in both versions of 2Q2U the relationships *influences* and *depends on* depicted in figures 2 and 3 have the same semantics as those stated in ISO 25010 standard.

Finally, particular 2Q2U models can in turn be instantiated in a purposeful way relying on the C-INCAMI nonfunctional requirement specification and context specification components of the SIQinU strategy. As shown later, we can combine attributes to sub-characteristics of the selected characteristics to fully instantiate the particular model and view, resulting in a requirements tree, for further measurement and evaluation purposes.

2.3 Current Version (v2.0) of 2Q2U: An Overview

The main aim for revising and updating the 2Q2U v1.0 is in light of the officially issued ISO 25010 standard in March, 2011 [11], a recently published report [19], and related literature such as for example [7], which deals with a quality model for mashup WebApps. Considering this recent ISO standard compared with the draft version [12], we have observed substantial changes both for characteristic/sub-characteristics definitions and characteristics included in quality models –mainly for QinU. Again, in upgrading 2Q2U, we tried to be as compliant as we could with the current ISO 25010 standard and considering also new evaluation characteristics for the current WebApps, while maintaining the same 2Q2U v1.0 quality modeling framework rationale. Below we summarize the main inclusions/deletions giving some definitions and details –mainly for *information quality* and *functional quality*. However, a thorough discussion and justification for the new 2Q2U version changes will be drawn in a separate paper, as well as a case study for mashup WebApps.

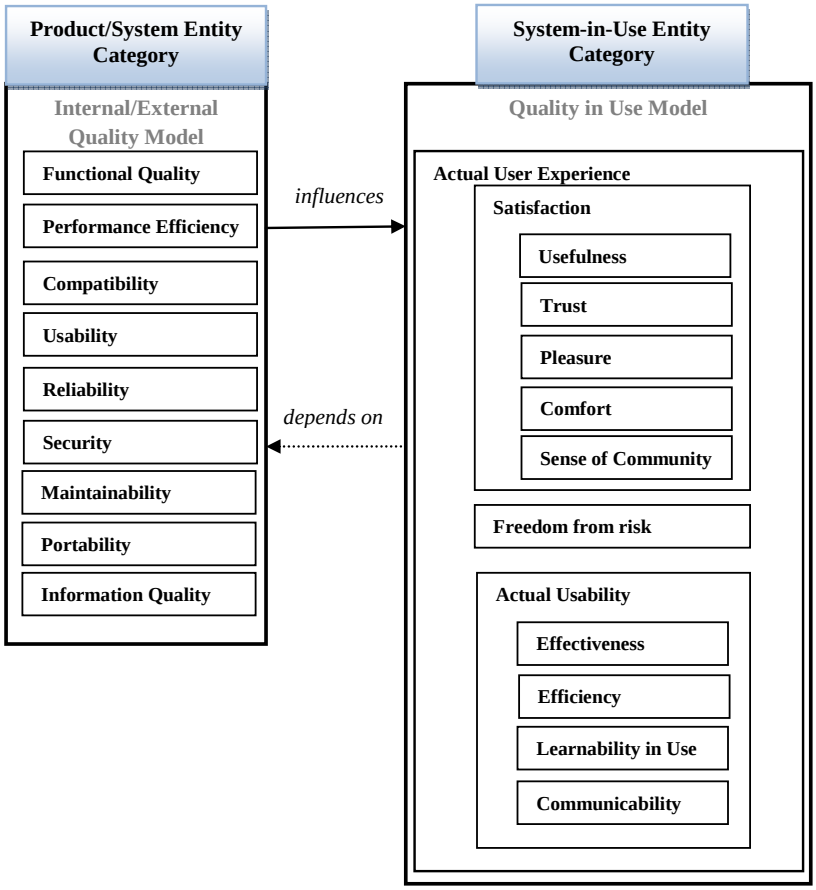


Figure 3: Quality model characteristics (with some sub-characteristics) and relationships between views in 2Q2U (v2.0).

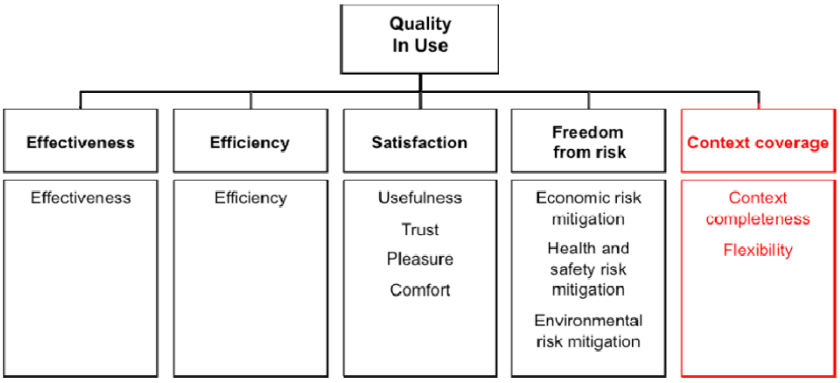


Figure 4: Quality in use model in ISO 25010 standard (taken from [11]).

Summarizing the second version of 2Q2U (see Fig. 3), first, we added to the QinU model two new

sub-characteristics, namely: *communicability* (as part of *actual usability*), and *sense of community* (as part of *satisfaction*), meanwhile we deleted the ISO *context coverage* characteristic (see Fig. 4) because in our approach this is represented in the non-functional requirements specification component, independently of quality models, as we will highlight in sub-section 3.2.1. Second, we rephrased the *functional suitability* characteristic given in the ISO product quality model as *functional quality* and rearranged its sub-characteristics; also we have eliminated two sub-characteristics of *information quality* and rephrased its definition. And third, we also rephrased the ISO *usability* definition –among other EQ sub-characteristics- in order to increase clarity and reduce ambiguity since in a nutshell they are defined in terms of EQ and QinU at the same time.

Table 1: Definition of *Information Quality* characteristics, sub-characteristics, and potential attributes (*in italic*). Adapted from [24]

Calculable Concept/ *Attribute*	**Definition**
1 Content Accuracy	Degree to which a product or system delivers information that is correct, credible and current.
1.1 Correctness	Degree to which information is correct both semantic and syntactic for a given language.
1.1.1 Syntactic correctness	Degree to which the content meets the rules of well-formedness for a given formal natural language system.
1.1.2 Semantic correctness	Degree to which the content is comprehensive, unambiguous and meaningful in context for a given formal natural language system.
1.2 Credibility	Degree to which the information is reputable, objective, and verifiable.
1.2.1 Authority (synonym: Reputability)	Degree to which the source of the information is trustworthy.
1.2.2 Objectivity	Degree to which the content (i.e., information or facts) is unbiased and impartial.
1.2.3 Verifiability (synonym: Traceability)	Degree to which the owner and/or author of the content can be verified.
2 Content Suitability	Degree to which a product or system delivers information with the right coverage, added value, and consistency, considering the specified user tasks and goals.
2.1 Value-added	Degree to which the information can be novel, beneficial, and contribute to causing a reaction for a given user and task at hand.
2.1.1 Novelty (synonym: Freshness)	Degree to which the information is fresh and contributes to make new decisions for an intended user goal.
2.1.2 Beneficialness	Degree to which the information is advantageous and contributes to make better decisions for an intended user goal.
2.1.3 Reactiveness	Degree to which the information is compelling and contributes to causing a reaction for an intended user goal.
2.2 Coverage	Degree to which the information is appropriate, complete and concise for the task at hand for an intended user.
2.2.1 Appropriateness	Degree to which the information coverage fits to an intended user goal.
2.2.2 Completeness	Degree to which the information coverage is the sufficient amount of information to an intended user goal.
2.2.3 Conciseness	Degree to which the information coverage is compactly represented without being overwhelming.
2.3 **Consistency**	Degree to which the content is consistent to the application's piece of information with respect to the intended user goal.

Below we give further details on the above issues:

Information quality, defined as the degree to which a product or system delivers accurate and suitable information which meets stated and implied needs when used under specified conditions. This EQ characteristic was also in 2Q2U v1.0 (see definition in the previous sub-section). In 2Q2U v1.0 there were four sub-characteristics, but now there remain only two sub-characteristics viz. *information accuracy* and *information suitability* (see Table 1). Hence, the former *content accessibility* sub-characteristic was moved in v2.0 to *usability* (likewise it is in ISO [11]); and *content legal compliance* was eliminated from the 2Q2U model due to the same reasons that ISO mentioned to remove every "compliance" characteristic from models, i.e. "compliance with standards or regulations that were sub-characteristics in ISO/IEC 9126-1 are now outside the scope of the quality model as they can be identified as part of requirements for a system" ([11], in p.21). Note that in Table 1 there are definitions for extra sub-characteristics and potential attributes, because in Section 5 some of them are used in the case study illustration.

Functional quality, defined in 2Q2U v2.0 as the degree to which a product or system provides accurate and suitable functions which meet stated and implied needs when used under specified conditions. Functional quality in our EQ model includes the three sub-characteristics stated in "functional suitability" by ISO 25010, i.e., correctness, appropriateness and completeness, but they are re-arranged in our representation while adding increased granularity. As shown in Table 2, now *functional quality* has only two sub-characteristics viz. *functional accuracy* and *functional suitability* (note the parallelism of terms with info quality). For example, *functional suitability* has in turn two sub-characteristics: *added-value* (represented with two potential attributes as *integratedness* and *beneficialness*), and functional *coverage* (represented with two potential attributes as *appropriateness* and *completeness*). For example, added-value [7] and particularly *integratedness* (defined as degree to which the product or system is made up of data/functional elements or views that behave in a synchronized and harmonious manner as a whole for the task at hand for a given user) can be useful for evaluating mashup WebApps from the composition standpoint.

Usability, which is rephrased in the 2Q2U EQ model as the degree to which the product or system has attributes that enable it to be understood, learned, operated, error protected, attractive and accessible to the user, when used under specified conditions. In the current product ISO quality model is defined as "degree to which a product or system can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use", which is a statement closer to the QinU semantic rather than in terms of EQ, showing a lack of separation of concerns between the terms of both views. In 2Q2U v2.0 *actual usability* (a characteristic of QinU as shown in Fig 3) is rephrased as degree to which specified users can achieve specified goals with effectiveness, efficiency, learnability in use, and without communicability breakdowns in a specified context of use.

Communicability [30] (as part of *actual usability* that evaluate pragmatic or do-goals [18]) is a new added sub-characteristic in the 2Q2U QinU model, which is defined as the degree to which specified users can achieve specified goals without communicative breakdowns in the interaction in a specified context of use. Also we added *sense of community* (as part of *satisfaction* that evaluate hedonic or be-goals), which is defined as the degree to which a user is satisfied when meeting, collaborating and communicating with other users with similar interest and needs. The latter sub-characteristic can be especially useful for evaluating a novel aspect of the satisfaction characteristic for social WebApps.

Ultimately, as indicated above, a thorough discussion of the specific reasons for including or rephrasing (sub-)characteristics in the EQ and QinU models is a bit outside of this sub-section and will be drawn in a separate paper.

Lastly, as mentioned in the Introduction Section our proposed approach relies on two pillars, namely: i) a *quality modeling framework*; and ii) a *measurement an evaluation strategy*.

In this section we have presented a general quality modeling framework (Fig. 1), which connects target entity categories with quality models and their relationships. Particularly, we have provided an overview of 2Q2U as a product/system/system-in-use quality modeling framework useful for not only specifying internal, EQ and QinU models represented by characteristics and sub-characteristics but also relating quality models each other by means of the *includes* and *depends on* relationships.

The second approach pillar for propose-oriented evaluations, i.e. M&E strategies are analyzed in the next section, stressing mainly the SIQinU strategy.

Table 2: Definition of *Functional Quality* characteristic, sub-characteristics, and potential attributes (*in italic*).

Calculable Concept/ *Attribute*	**Definition**
1 Functional Accuracy	Degree to which a product or system provides functions which are correct and credible.
1.1 Correctness	Degree to which a component/function provides the correct results with the stated degree of precision and consistency.
1.1.1 Precision	Degree to which the result is the exact value.
1.1.2 Consistency	Degree to which the result is within the stated set of values.
1.2 Credibility (synonym: Trustfulness)	Degree to which a component/function is reputable and verifiable.
1.2.1 Reputability (Synonym: Authority)	Degree to which the source (owner) of a component/function is trustworthy.
1.2.2 Verifiability synonym: Traceability)	Degree to which the enterprise/developer/version/date of a component/function can be corroborated.
2 Functional Suitability	Degree to which a product or system provides functions with added value and the correct coverage (with regard to appropriateness and completeness), considering the specified user tasks and goals.
2.1 Added-value	Degree to which the product or system is integrated and beneficial for the task at hand to a given user.
2.1.1 Integratedness	Degree to which the product or system is made up of data/functional elements or views which behave in a synchronized and harmonious manner as a whole for the task at hand for a given user.
2.1.2 Beneficialness	Degree to which the product or system is advantageous and contributes to make better decisions for an intended user goal.
2.2 Coverage	Degree to which a set of components/functions is appropriate and complete for the task at hand for an intended user.
2.2.1 Appropriateness	Degree to which the functional coverage fits to an intended user goal.
2.2.2 Completeness	Degree to which the functional coverage is the sufficient amount of functions for an intended user goal.

3 Overview of the SIQinU Strategy

3.1 Introduction to M&E Strategies

So far, we have developed two M&E strategies, namely: GOCAME (*Goal-Oriented Context-Aware Measurement and Evaluation*) and SIQinU (*Strategy for understanding and Improving Quality in Use*). In chronological order, we first developed GOCAME [25], which is a multi-purpose strategy that follows a goal-oriented and context-sensitive approach in defining and performing M&E projects. GOCAME is a multi-purpose strategy because can be used to evaluate (i.e. "understand", "predict", etc.) the quality for not only product, system and system-in-use entity categories but also for other ones such as resource and process, by using their instantiated quality models accordingly. Moreover, the evaluation focus can vary, i.e. ranging from "external quality", "quality in use" to "cost" (or even "capability quality" as we did in [29]). However, GOCAME does not incorporate improvement cycles as in SIQinU. Rather it can be used to understand the current or further situation (as an evaluation snapshot) of concrete entities. Recently, in mid-2010 we developed SIQinU [17], which is a specific-purpose and context-sensitive strategy to incrementally and continuously improve a WebApp-in-use's QinU by means of mapping actual usage problems to measurable EQ attributes –that are inherent to a WebApp-, and then by performing improvement actions that enable evaluators assessing the gain both at EQ and QinU levels. SIQinU is a specific-purpose strategy because it can be used to evaluate (i.e. just for the purpose of "understand" and "improve") the quality for only system-in-use and system entity categories by using their instantiated QinU and EQ models respectively.

Briefly outlined, the GOCAME strategy follows a goal-oriented approach in which all the activities are guided by an agreed information need; this information need is intended to satisfy particular nonfunctional requirements of some entity category (and in the end a concrete entity) for a particular purpose and stakeholder's viewpoint. The nonfunctional requirements are represented by concept models including high-level calculable concepts (e.g. EQ) as in 2Q2U's quality models, 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. Data and information coming from measurements and evaluations are used for analysis and recommendation activities and ultimately for the decision making process.

GOCAME is based on three main principles or capabilities, namely: i) a conceptual framework utilizing an ontological base; ii) a well-defined M&E 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 a 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 developed the C-INCAMI (*Contextual-Information Need, Concept model, Attribute, Metric and Indicator*) framework and its components [20, 25] based on our metrics and indicators ontology. Note that SIQinU re-uses totally C-INCAMI conceptual framework and its 6 components (commented in sub-section 3.2.1).

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, sequences and parallelisms, roles, 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 [2], which is also compliant with both the C-INCAMI conceptual base and components. Note that SIQinU re-uses to some extent the GOCAME activities; however, there are particular phases and activities in SIQinU (summarized in sub-section 3.2.2) that are not included in GOCAME.

Finally, GOCAME's third principle is methods and tools, which can be instantiated from both the conceptual framework and process, as for example the WebQEM methodology [28] and its tool called C-INCAMI_tool. M&E methods and tools are also re-used by SIQinU. However, other methods and techniques that are not included in GOCAME such as those for changing and improving the current WebApp version are needed (this is summarized in sub-section 3.2.3).

So in a nutshell, SIQinU is in alignment with GOCAME regarding its M&E conceptual framework, activities and methods. Since the aim of this paper is illustrating the use of 2Q2U and SIQinU for purpose-oriented evaluations (sections 4 and 5), we need first to overview the three SIQinU principles before providing a deeper comprehensive explanation later in the document.

3.2 SIQinU Strategy

SIQinU utilizes the 2Q2U quality modeling framework for quality models and the C-INCAMI conceptual framework for instantiating M&E projects in a purposeful way. SIQinU, non-intrusively collects user behavior data from a real context of use, and provides an integrated means to evaluate QinU and EQ for WebApps with a primary objective of improvement through an evaluation process and methods that are consistent and repeatable.

SIQinU collects user task data from log files [6] that were derived through for example adding snippets of code in a real WebApp-in-use that allow recording user activity, with the aim to derive nonfunctional requirements measures and indicators for QinU, thus leading us to understand the current QinU satisfaction levels met. Then, by performing a preliminary analysis EQ requirements that can affect QinU are derived, followed by proposed recommendations for improvement. After performing the changes on the WebApp, and after conducting studies with the same user group in the same daily environment (context) with the new (and improved) version, an assessment of the improvement gain can be gauged.

With this knowledge of improvement, we can then extract relationships in a cyclic manner by isolating changes in EQ attributes that affected QinU attributes in a positive way. Thereby, each

iteration between QinU and EQ can result in continued improvement. Ultimately, in the process of using SIQinU, we are able to gain insight regarding the *depends on* and *influences* relationships for the particular 2Q2U instantiated models, and their characteristics and attributes driven by our purpose to improve.

3.2.1 C-INCAMI Conceptual Framework

The C-INCAMI framework defines the concepts and relationships for the M&E domain. It is designed to satisfy a specific information need in a given context defining all concepts and relationships that are used along all the M&E activities. The framework has six components:

i) *M&E project definition*,

ii) *Nonfunctional requirements specification*,

iii) *Context specification*,

iv) *Measurement design and implementation*,

v) *Evaluation design and implementation*, and

vi) *Analysis and recommendation specification.*

Of particular interest to illustrate this article in instantiating quality models for the purposes of understanding and improvement are C-INCAMI's Nonfunctional Requirements Specification (NFRS), and Context Specification components, both shown in Fig. 5. For conciseness reasons we describe only these two, while the others can be found in [25]. In addition, all the terms (Table 3), terms' attributes (Table 4), and terms' relationships (Table 5) are defined, and highlighted below in italic.

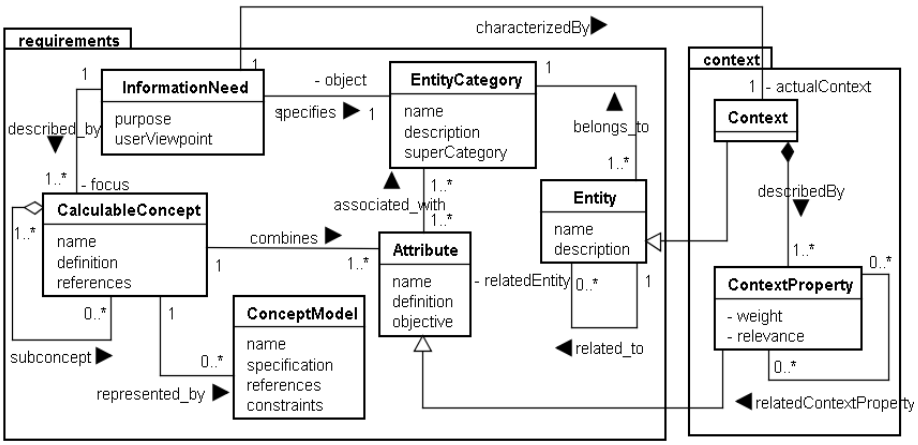


Figure 5: C-INCAMI Nonfunctional requirements and context specification components

In a broad sense, the NFRS component (labeled requirements in Fig. 5) specifies the *Information Need* of any M&E project; that is, the *purpose* (e.g. "understand", "predict", "control", "improve" etc.) and the *userViewpoint* (e.g. "developer", "beginner user", etc). In turn, it *focuses* on a

CalculableConcept (whose *name* is for instance "QinU") and *specifies* the *EntityCategory* to evaluate –e.g. a resource, process, product, etc. *superCategory*-, by means of a concrete *Entity* –e.g., the "amazon.com shopping basket". A calculable concept is defined as an abstract relationship between attributes of an entity and a given information need.

On the other hand, the selected calculable concept and its *subconcepts* can be *represented by* a *Concept Model* (e.g. in "QinU model") where the leaves of an instantiated model (e.g. a requirements tree) are *Attributes associated with* an entity category. In fact, the concrete entity *belongs* to an entity category (e.g. whose *name* is "e-book shopping basket").

The context specification component (labeled context in Fig. 5) delineates the state of the situation of the entity to be assessed with regard to the information need (see *characterizedBy* relationship). *Context*, a special kind of *Entity* in which related relevant entities are involved, can be quantified through its related entities that may be resources –as a network infrastructure, a working team, user tasks, etc.-, the organization or the project itself, among others. To describe the context, properties of the relevant entities, which are also attributes called *ContextProperties* are used. In our case, the context is particularly important regarding QinU requirements as instantiation of requirements must be done consistently in the same context in order to evaluations and improvements be accurately measured and compared. (Recall that in 2Q2U v2.0 we deleted the ISO *context coverage* characteristic in Fig. 4, since as shown here the context specification is rather independent of quality models).

Lastly, the measurement component relates the terms that allow specifying and using the metrics that quantify attributes. While the evaluation component includes the concepts and relationships intended to specify the evaluation design and implementation. Note 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 degree the stated information need was met.

Table 3: Definition of terms of Fig. 5, for the non-functional requirements and context specification components

Concept Name	Definition
Attribute (synonyms: Property, Feature)	A measurable physical or abstract property of an entity category.
Calculable Concept (synonym: Characteristic, Dimension, Factor)	Abstract relationship between attributes of entity categories and information needs.
Concept Model (synonyms: Factor , Feature Model)	The set of sub-concepts and the relationships between them, which provide the basis for specifying the concept requirements and its further evaluation or estimation.
Context	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.
ContextProperty (synonyms: Context Attribute, Feature)	An attribute that describes the context of a given entity; it is associated to one of the entities that participates in the described context.
Entity	A concrete object that belongs to an entity category.
EntityCategory (synonym: Object)	Object category that is to be characterized by measuring its attributes or properties.
InformationNeed	Insight necessary to manage objectives, goals, risks, and problems.

Table 4: Definition of terms' attributes of Fig. 5, for the non-functional requirements and context specification components

Concept	Attribute	Description
Attribute	name	Name of an attribute to be identified.
	definition	An unambiguous description of the attribute meaning.
	objective	Goal or purpose to measuring this attribute.
Calculable Concept	name	Name of a calculable concept to be identified.
	definition	An unambiguous description of the calculable concept meaning.
	references	References to bibliographical or URL resources, where additional and authoritative information of a given calculable concept can be referred.
Concept Model	name	Name of a concept model to be identified.
	specification	A formal or semiformal representation of a concept model.
	references	References to bibliographical or URL resources, where additional and authoritative information of a given concept model can be referred.
	constraints	The specific restrictions to a graph structure.
Context Property	weight	The relative value of the context property within a given context description.
	relevance	The pertinence of using the context property to describe the context for a particular information need or contextual entity.
Entity	name	Name of an entity to be identified.
	description	An unambiguous description of the entity meaning.
Entity Category	name	Name of an entity category to be identified.
	description	An unambiguous description of the entity category meaning.
	superCategory	The super category of an entity category.
Information Need	purpose	The goal for performing the evaluation, which can be for instance "understand", "predict", "improve", "control", etc.
	userViewpoint	The intended stakeholder as "developer", "final user", etc. from which the focus concept (and model) will be evaluated.

Table 5: Definition of relationships between terms of Fig. 5, for the non-functional requirements and context specification components

Name	Description
associated_with	One or more measurable attributes are associated with one or more entities categories.
belongs_to	An entity belongs to an entity category.
characterizedBy	An information need is characterized by the context.
combines	A calculable concept combines (associates) one or more measurable attributes.
described_by	One or more calculable concepts are defined in order to satisfy a concrete information need. So, a calculable concept describes a concrete information need.
describedBy	A context is described by its context property.
relatedContextProperty	A context property may be composed of none or more related context property.
related_to	An entity can be related to none or more related entities.
represented_by	A calculable concept can be represented by none or several concept models.
subconcept	A calculable concept may be composed of none or several sub-concepts, which are in turn calculable concepts.
specifies	An information need specifies an entity category.

3.2.2 The SIQinU Process

The SIQinU process for understanding and improving QinU has six main phases (specified in Fig. 6 using the SPEM notation [31]), which are briefly described below with phase (Ph.) reference numbers:

(Ph. I) Specify Requirements and Evaluation Criteria for QinU. By taking into account the recorded data of the WebApp's usage, we re-engineer QinU requirements. This embraces designing tasks, defining user type, specifying usage context and characteristics for QinU (using e.g. the 2Q2U v2.0 QinU model), particularly, for *actual usability* as we will detail in Section 5. Based on these specifications, metrics (for measurement) and indicators (for evaluation) should be agreed.

(Ph. II) Perform QinU Evaluation and Conduct Preliminary Analysis. As established in Phase I, data is collected pertaining to the tasks with associated sub-goals noting for instance the *effectiveness*, *efficiency* and *learnability in use* and their related measures to perform the task goal and sub-goals. Depending on the WebApp-in-use's ability to collect the data, we also collect the date/time the data is gathered, errors, task and sub-task completion and accuracy, and user group type, among others. Ph II includes collecting data, performing calculations/evaluations, and conducting the preliminary analysis.

(Ph. III) Derive/Specify Requirements and Evaluation Criteria for EQ. Based on Phases I and II, we derive (aligned with our assumption that possible improvement is needed) and design the EQ requirements, i.e. characteristics and attributes, followed by their metrics and indicators in order to evaluate the WebApp (e.g. the JIRA v.1 entity) through its inspection. For instance, a focus on EQ characteristics (using e.g. the 2Q2U v2.0 EQ model), *usability* and *information quality* can be used to determine possible effects on *actual usability*. Choosing the EQ requirements is based on the purpose and associated EQ attributes that are possibly related to the problems identified in Phase II.

*(Ph. IV) Perform EQ Evaluation and Analysis***.** Based on Phase III, we measure and evaluate these EQ attributes by inspection (i.e. on the system rather than in the WebApp-in-use) based on the metrics and indicators previously selected.

(Ph. V) Recommend and Perform Improvement Actions for EQ. Based on Phase IV, for the EQ attributes that require improvement, we make improvement recommendations for modifying the WebApp, e.g. JIRA version 1 to 1.1. After the changes have been made, then we can re-evaluate using again Phase IV to note the EQ improvement gain from the previous benchmarked evaluation score.

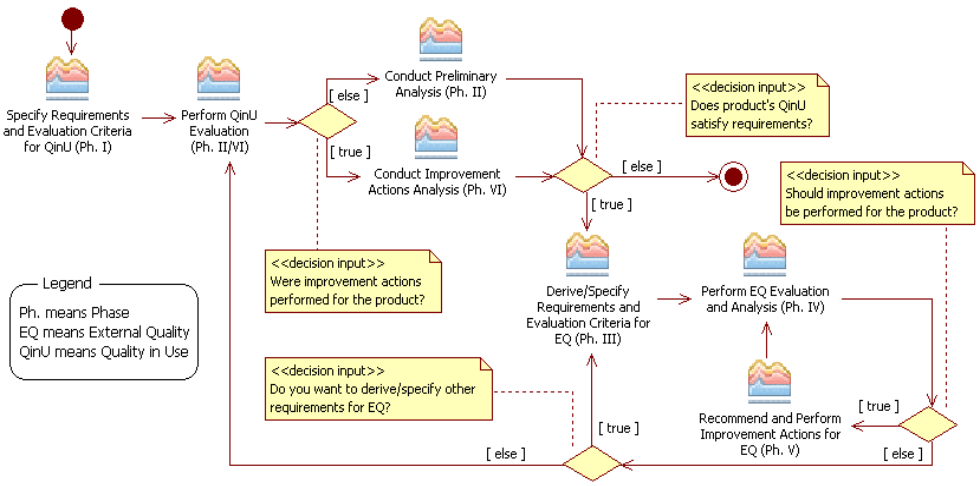


Figure 6: Process overview for understanding and improving quality in use (SIQinU)

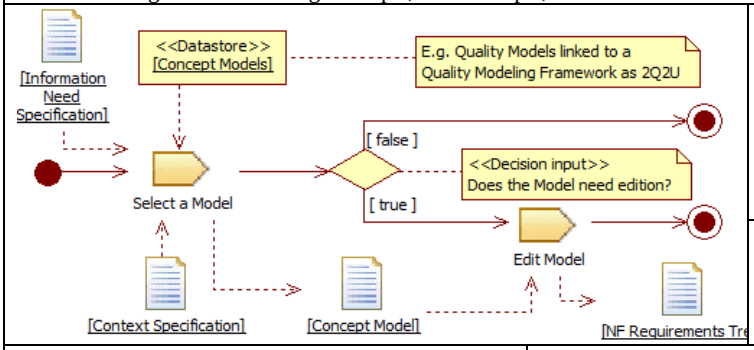
Table 6: SIQinU Phases, activities and work products

Phases (Ph.)	**Activities involved**	**Artifacts (Work Products)**
Ph. I Specify Requirements and Evaluation Criteria for QinU	i) Establish QinU Information Need; ii) Specify QinU sub-project Context; iii) Design Tasks; iv) Select QinU Concept Model; v) Design QinU Measurement and Evaluation; vi) Design Preliminary Analysis	- QinU Information Need spec. - QinU Context specification - Task/sub-tasks specification - QinU NFR tree - QinU metrics and indicators specification - Analysis design
Ph. II Perform QinU Evaluation and Conduct Preliminary Analysis	i) Collect and parse data of tasks/sub-tasks; ii) Quantify QinU Attributes; iii) Calculate QinU Indicators; iv) Conduct preliminary analysis.	- Parsed data file - Measure and indicator values for QinU - QinU preliminary analysis report
Ph. III Derive/ Specify Requirements and Evaluation Criteria for EQ	i) Establish EQ Information Need; ii) Specify EQ sub-project Context; iii) Select EQ Concept Model; iv) Design EQ Measurement; v) Design EQ Evaluation	- EQ Information Need specification - EQ Context specification - EQ NFR tree - EQ metrics and indicators specification
Ph. IV Perform EQ Evaluation and Analysis	i) Quantify EQ Attributes; ii) Calculate EQ Indicators; iii) Conduct an EQ analysis and identify parts of the WebApp that need improvement.	- Measure and indicator values for EQ - EQ Analysis report (and new report after re-evaluation)
Ph. V Recommend and Perform Improvement Actions for EQ	i) Recommend improvement actions; ii) Design Improvement Actions; iii) Perform Improvement Actions; iv) Evaluate Improvement Gain (Ph IV).	- EQ Recommendations report - Improvement plan - New application version
Ph. VI Re-evaluate QinU and Analyze Improvement Actions	i) Evaluate QinU again; ii) Conduct Improvement Action analysis; iii) Develop *depends on* and *influences* relationships between EQ improvements and QinU.	- New measure and indicator values for QinU - QinU Improvement analysis report - EQ/QinU attribute relationship table

(Ph. VI) Re-evaluate QinU and Analyze Improvement Actions. Once the new version has been used by real users in their daily tasks and context, then, it is necessary to evaluate QinU again to determine the effects of what was improved for the WebApp's EQ on QinU. This enables us to develop relationships between EQ improvements done in Ph. V and positive impacts in QinU. We can then continue this improvement in other parts of the application as well as continue to develop more granular relationships.

Additionally, in Table 6 we provide an abridged description for each SIQinU phase (Ph.) with the respective activities and work products or artifacts produced. Furthermore, in Table 7 an activity template in which the objective and definition, input and output artifacts, roles, sub-activities with depicted interdependencies, pre- and post-conditions are documented for the *Select a Concept Model* activity. This activity specified in a generic form in the template of Table 7, can be used for both in Ph. I –i.e. activity (iv) *Select QinU Concept Model*, in Table 6- and in Ph. III –i.e. activity (iii) *Select EQ Concept Model*-. The reader can refer to [17] for more details of the SIQinU process.

Table 7: Process template in which information and views are documented for the *Select a Concept Model* activity

<table>
<tr><td colspan="2">Activity: Select a Concept Model</td><td>Activity Code: Not shown</td></tr>
<tr><td colspan="3">Objective: to have as result a requirements tree that will be used for measurement, evaluation and analyses.</td></tr>
<tr><td colspan="3">Description: taking into account the evaluation focus, the entity category and the user viewpoint from the Information Need Specification document, as well as the Context Specification document, a Concept Model must be selected from a repository. Note that for example a quality model can be linked to a quality modeling framework. If the selected model is not totally suitable to the NFR manager needs either because it does not have all the required relations among sub-concepts, or because it has no attributes (i.e. it is not previously instantiated) the model should be edited. Editing should take into account model constraints, and the adding and/or removing concepts, sub-concepts, attributes and relationships accordingly.</td></tr>
<tr><td colspan="2"> 
</td><td>Sub-activities / Interdependencies (shown in the left activity diagram):
• Select a Model
• Edit the Model

Involved Roles:
• Evaluation Requester
• NF Requirements Manager</td></tr>
<tr><td>Input Artifacts:
• Concept Models repository
• Information Need Specification
• Context Specification</td><td colspan="2">Output Artifacts:
• Nonfunctional Requirements Tree (i.e. the selected/edited characteristics, sub-characteristics and attributes)</td></tr>
<tr><td>Pre-conditions: there is a repository with concept models.</td><td colspan="2">Post-conditions: the instantiated requirements tree.</td></tr>
</table>

3.2.3 The Methods

While activities state ‘what’ to do methods, on the other hand, describe ‘how’ to perform these activities, which in turn can be automated by tools. In addition a methodology is a set of related methods. Because the above process includes activities such as specify the nonfunctional requirements, design and implement the measurement and evaluation, and so on, we have envisioned a methodology that integrates all these M&E 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 M&E process activities produces the different outcomes. Particularly, the WebQEM (*Web Quality Evaluation Method*) methodology [28] and its associated C-INCAMI_tool were instantiated from the conceptual framework and process. The methodology supports a feature-driven evaluation approach, relying on experts and/or end users to evaluate and analyze different views of quality such as EQ and QinU for system and system-in-use applications. Note that WebQEM can be used for any quality focus of any entity category.

Besides WebQEM, SIQinU also employs particular methods and techniques for instance for changing the WebApp version (see in Table 6, e.g. Ph. V, *Perform Improvement Actions* activity), which can range from parameterized reconfigurations, code restructuring, refactoring (as used in [26])

to architectural redesign. The eventual method employed depends on the scope of the improvement recommendation as well as the resources of the owner/developer and the desired effect.

3.3 Evaluation Strategies beyond SIQinU and GOCAME

As above mentioned, we have developed so far two M&E strategies, namely: GOCAME and SIQinU. Both strategies rely on the three summarized capabilities, namely: i) a conceptual framework, which is made up of a sound conceptual base and components that group its elements; ii) a well-established M&E process; and, iii) methods and tools instantiated from both the framework and process, which perform the particular activities.

As the reader can surmise, if we also take into account the quality modeling framework depicted in Fig. 1, many other strategies beyond SIQinU and GOCAME can be developed by reusing the above three capabilities. Let us consider the *influences* and *depends on* relationships between views on quality focuses/models. Particularly, let's examine the relationships indicating that the resource quality (e.g. a new integrated tool) influences the process quality (e.g. a development process) or vice versa the process quality depends on the resource quality. Hence, to understand the actual quality and explore these relationships, a new strategy (that re-uses the three capabilities, and adapts mainly the process and methods to this current situation) should be envisioned. Moreover, the strategy can give support to a comparison mechanism, where the improvement gain of using "the new integrated tool" compared the "commonly used tool" can be gauged with respect to improvements obtained in the process.

Ultimately, by enhancing the strategies scope beyond the project level, in order to give support to M&E information needs at different organizational levels (i.e. not only at tactic –project- level but also at strategic level), a new strategy that links different levels of information needs can be built as well. For this new strategy, the three above-mentioned capabilities can be a valuable asset as well.

4 Using 2Q2U and SIQinU for Purpose-Oriented Evaluations

In this section we illustrate the procedure to concretely instantiate quality models based on 2Q2U and SIQinU for both QinU and EQ views for the purpose of not only understanding but also improving WebApps (the first specific contribution indicated in the Introduction Section). Our vision for this research is that understanding is the means and improvement is the ultimate goal. Hence, SIQinU is used to evaluate (i.e. just for the purpose of "understand" and "improve") the quality for WebApp-in-use and WebApp entity categories by using their instantiated QinU and EQ models respectively for measurement, evaluation and analysis. The instantiated QinU and EQ models and their *influences* and *depends on* relationships come from the 2Q2U quality modeling framework shown in Section 2.

In order to instantiate models for M&E in a consistent, repeatable and purpose-oriented way, we utilize as procedure the SIQinU's process and C-INCAMI M&E framework capabilities.

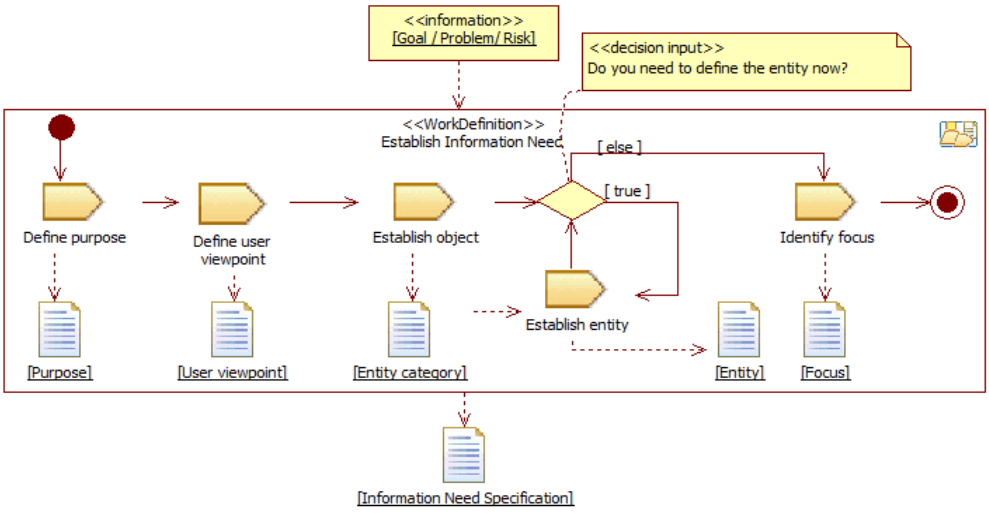


Figure 7: Activities to *Establish the Information Need* (as used in GOCAME).

Of particular use in instantiating quality models for the purposes of "understand" and "improve" is the C-INCAMI's NFRS and Context Specification components (recall Fig. 5), and the following three related generic processes: i) *Establish Information Need*, (depicted in Fig. 7); ii) *Specify Context*; and iii) *Select a Concept Model*, which template and activity diagram was shown in Table 7. We call them generic activities for the nonfunctional requirements specification process, because these activities – used as such in GOCAME- are accordingly specialized in the SIQinU strategy for QinU or EQ, e.g. *Establish QinU Information Need* in Fig. 8, or *Establish EQ Information Need* in Fig 9, and so forth, as *Select a QinU Concept Model*, or *Select an EQ Concept Model* respectively.

It is worth mentioning the correspondence and consistency of terms between the C-INCAMI conceptual framework and the workflow of activities. For instance, labels of activities and artifacts in Fig. 7 stem or are made up from the labels of terms, attributes and relationships of the nonfunctional requirements specification component shown in Fig. 5.

Reading the activity diagram in Fig. 7, we can say that to establish the *Information Need* we first have to define the *purpose* (e.g. "understand") and *userViewpoint* (e.g. "beginner tester") and establish the *object* (where the output of this task is the *EntityCategory*, as for example a "WebApp-in-use", and the concrete *Entity* is "JIRA-in-use v.1"). In order to identify the *focus* of the information need, the "QinU" calculable concept is instantiated. Examining the activity diagram within Table 7, we can say that the *Information Need* and *Context* specification documents are inputs to the select a model activity. Also, there is another input to select a model from the *Concept Models* <<datastore>>. This is to say, knowing beforehand the focus, purpose, user, and entity category a "QinU model" can be chosen, for example, from the 2Q2U quality modeling framework (e.g. from the 2Q2U v2.0). In addition, editing the model may be necessary to remove concepts, *sub-concepts*, and add *attributes* accordingly.

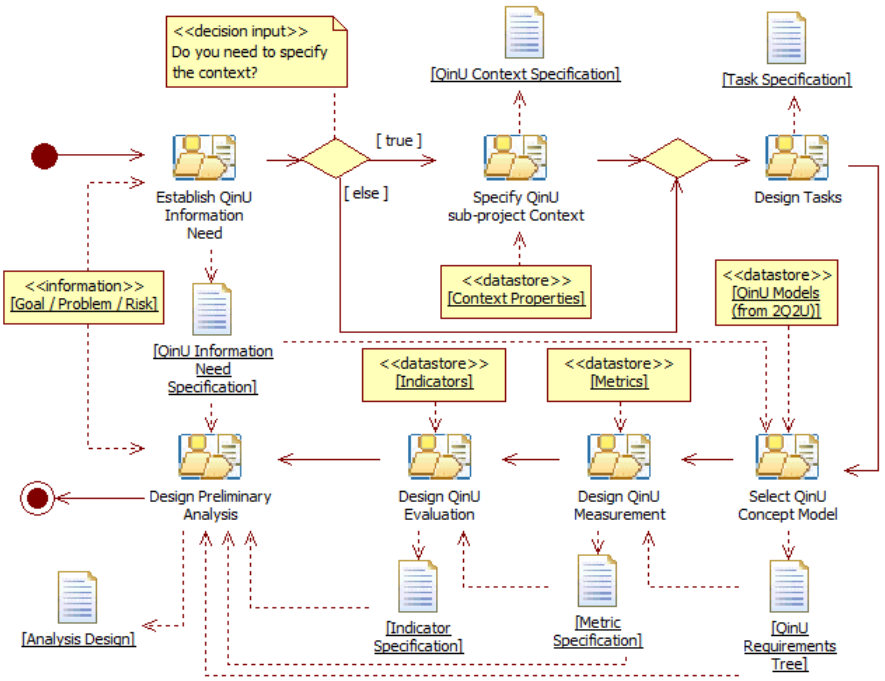


Figure 8: Activities for the *Specify Requirements and Evaluation Criteria for QinU* process (SIQinU Phase I)

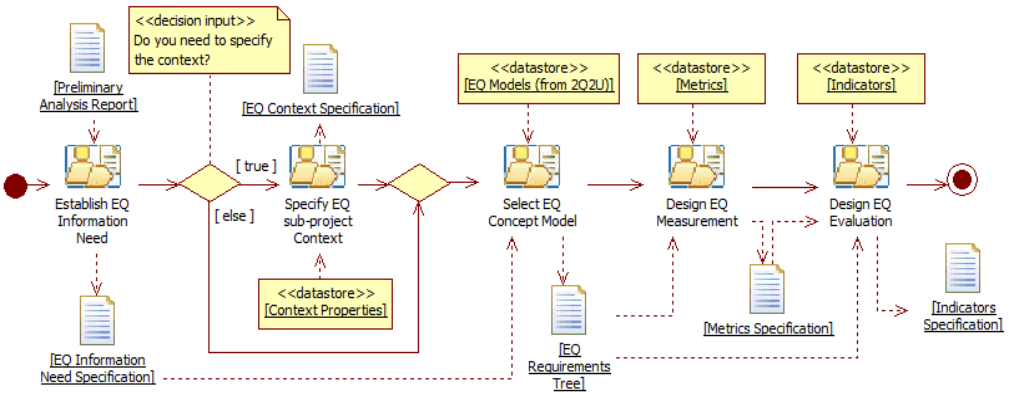


Figure 9: Activities for the *Derive/Specify Requirements and Evaluation Criteria for EQ* process (SIQinU Phase III)

Particularly for SIQinU, Fig. 8 shows the activities for the *Specify Requirements and Evaluation Criteria for QinU* phase (Ph. I). Note this diagram corresponds to the textual form shown in the 1st row of Table 6. To the establish the QinU information need activity, the purpose is to "understand" the current situation of "JIRA-in-use v.1", and in Ph VI, the purpose is to "understand" the ulterior situation –or also called "improve"- of the enhanced system-in-use, i.e. "JIRA-in-use v.1.1" both for

the "beginner tester" viewpoint. The specify QinU sub-project *context* activity deals with the selection of *context properties* such as "number of users", "user demographics", "number of tasks", "type of task", "network bandwidth", "screen size", "screen resolution", "room lightness", among others. Because, *context* is a special kind of *entity* (recall Fig. 5), the names of categories of entities (e.g. user, task, infrastructure, environment, etc.) and concrete entities should be recorded in the M&E project as well.

Fig. 8 shows also the select a QinU concept model activity. In this case the "QinU model" must be chosen from the 2Q2U quality modeling framework, and edited regarding the evaluation and strategy aim. SIQinU is a non-intrusive strategy used just for M&E of performance (do-)goals from the QinU standpoint. In the right side of Fig. 10, the instantiated characteristic and sub-characteristics from 2Q2U v2.0 are depicted, which will be used in the JIRA case study of Section 5. In addition, the resulting QinU requirements tree in which sub-concepts *combine* attributes is shown in the 1st column of Table 8.

On the other hand, Fig. 9 shows the activities for the *Derive/Specify Requirements and Evaluation Criteria for EQ* phase (Ph. III) –see also the 3rd row of Table 6. To establish the EQ information need activity the purpose is to "understand" the current situation of "JIRA v.1", and in Ph IV (in the re-evaluation process), the purpose is to "understand" the ulterior situation of the improved WebApp, i.e. "JIRA v.1.1" both regarding the "final user" viewpoint, since the evaluation is made by inspection where experts play the role of final users. Fig. 9 shows also the select an EQ concept model activity. In this case the "EQ model" must be chosen from the 2Q2U quality modeling framework, and edited accordingly.

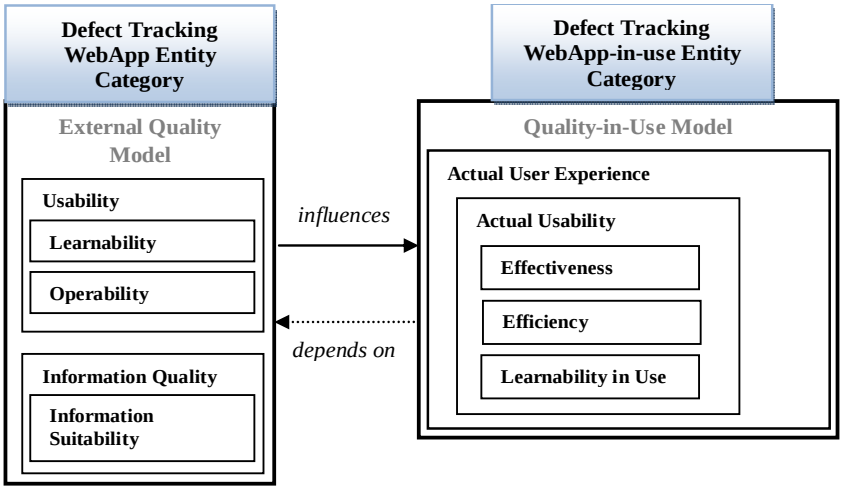


Figure 10: 2Q2U v2.0 model instantiation with the EQ and QinU characteristics and sub-characteristics used in the JIRA case study

In the left side of Fig. 10, the instantiated EQ characteristic and sub-characteristics from 2Q2U v2.0 are depicted, which will be used in the JIRA case study as well. In addition, the resulting EQ requirements tree, in which sub-concepts *combine* attributes is shown in the 1st column of Table 9.

Note that we could have selected other characteristics for both EQ and QinU, but we purposely

instantiated these characteristics and sub-characteristics from the 2Q2U modeling framework as per our information need via the SIQinU's process and C-INCAMI M&E framework capabilities. Both requirements trees are inputs to the implementation of the measurement and evaluation (phases II, IV and VI) which produce measurement and indicator values. These values are the basis for further analysis and recommendations. On the other hand, the *influences* and *depends on* relationships for instantiated QinU and EQ models are now explored in light of concrete attributes. Particularly, we can hypothesize the precise relationships between EQ and QinU attributes that contribute to improvement as we discuss later on.

Ultimately, the separation of concerns between the quality modeling framework and strategy –as we propose in this paper-, result in a flexible yet consistent and logical approach.

5 A Quality Improvement Lifecycle using SIQinU: The JIRA Case Study

This section illustrates mainly the second and third contributions indicated in the Introduction Section, namely, the joint use of the instantiated requirements trees from 2Q2U together with the systematic use of SIQinU in order to uncover the *influences* and *depend on* relationships considering the QinU/EQ/QinU improvement cycle, using also excerpts of a case study conducted in mid-2010. This case study was also thoroughly illustrated in [17]. But we elaborate and emphasize in this section particularly with regard to the above contributions.

The case study examined JIRA, a defect reporting and issue tracking 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 in our chosen context of use. We studied approximately 50 beginner users in a real work environment in their daily routine of reporting defects in a software testing department –in a real company specializing in software quality and testing. 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. The beginner user group was chosen because this was the client's primary concern, as is with most software users, in getting new users up to speed and productive as soon as possible.

In [17] we used the 2Q2U first version (sub-section 2.2). As an update, here we illustrate the case study with 2Q2U second version (sub-section 2.3). This has no real semantic implications but rather a change of some characteristics or sub-characteristics names. Looking at Fig. 10 to the EQ instantiated model, the name "Usability" –defined also in sub-section 2.3- is the former "Operability" in v1.0, and the current sub-characteristic "operability" is the former "ease of use". Note that "Information quality" and "information suitability" remain unchanged. Regarding the QinU instantiated model, the former characteristics "Effectiveness in use" and "Efficiency in use" now are labeled "Effectiveness" (defined as: degree to which specified users can achieve specified goals with accuracy and completeness in a specified context of use), and "Efficiency" (defined as: degree to which specified users expend appropriate amounts of resources in relation to the effectiveness achieved in a specified context of use). The rest of characteristics in the QinU model (Fig. 10) remain unchanged and they were defined in Section 2. Finally the names of all attributes in [17] remain also the same in tables 8 and 9, as well

as their meanings. So for this case study, we maintained backward compatibility between instantiated models for both 2Q2U versions.

Next, we discuss some aspects of the SIQinU phases; mainly those aimed at showing models instantiation, EQ and QinU improvement gains and potential uncovered relationships. In Ph. I, we start by using 2Q2U to purposefully instantiate a QinU model from the "Actual Usability" standpoint. We first want to understand the current situation of the concrete entity (JIRA v.1) from the beginner tester user viewpoint performing the above mentioned task (note that the task was in turn decomposed into 5 sub-tasks [17]). To do this, we design the QinU requirements as shown in the right part of Fig. 10, resulting in the requirements tree shown in the column 1 of Table 8.

Using the requirements tree from Ph. I of our instantiated model, we conduct the QinU evaluation as Ph. II of SIQinU (see Table 6). This evaluation is done for each attribute specified in Ph. I, and results in a global evaluation for JIRA v.1 of 53.3% as shown in Table 8.

Note that in the Ph. I activities shown in Fig. 8: *Design QinU measurement* and *Design QinU evaluation* per each attribute of the *QinU requirements tree specification* we selected a metric and its elementary indicator (see details of design of metrics and indicators in [17]). All decision criteria for QinU indicators are designed with three acceptability ranges in the percentage scale, namely: a value within 70-90 (a marginal –gray- range) indicates a need for improvement actions; a value within 0-70 (an unsatisfactory –dark gray- range) means changes must take place with high priority; a score within 90-100 indicates a satisfactory level –light gray- for the analyzed attribute.

Table 8: QinU requirements tree and evaluation results of JIRA-in-use v.1 and JIRA-in-use v.1.1 (after modifications of the WebApp). The legend EI stands for Elementary Indicator; P/GI for Partial/Global Indicator

	JIRA v.1		JIRA v.1.1	
Characterisitcs and Attributes	EI	P/G I	EI	P/G I
1. Actual Usability		53.3%		67.0%
1.1. Effectiveness		73.2%		86.7%
1.1.1. *Sub-Task Correctness*	86.4%		91.9%	
1.1.2. *Sub-Task Completeness*	87.9%		95.5%	
1.1.3. *Task Successfulness*	45.5%		72.7%	
1.2. Efficiency		29.3%		42.8%
1.2.1. *Sub-Task Correctness Efficiency*	37.4%		44.3%	
1.2.2. *Sub-Task Completeness Efficiency*	37.5%		47.3%	
1.2.3. *Task Successfulness Efficiency*	13.1%		36.8%	
1.3. Learnability in use		57.3%		71.6%
1.3.1. *Sub-Task Correctness Learnability*	78.8%		75.1%	
1.3.2. *Sub-Task Completeness Learnability*	26.4%		77.3%	
1.3.3. *Task Successfulness Learnability*	66.7%		62.5%	

Based on the evaluation, using the defined indicators, we are able to determine which attributes had low performance or problems, e.g., *Sub-task completeness learnability* which had an unsatisfactory rating of 26.4%. We can rank them in terms of priority by lowest performing at the top, or depending on our requirements, which may weight other attributes more heavily, do a more complex priority ranking accordingly.

SIQinU in this case study using JIRA was implemented in a non-intrusive way through interpretation of log files and automated application and calculation of indicators thereby resulting in a

list of problem areas (those attributes that rated unsatisfactory). Note that other complementary techniques could be used such as traditional observational techniques [23] to understand user problems while interacting with the selected tasks and to help derive EQ attributes. Also, the introduced "Communicability" characteristic in the 2Q2U v2.0 QinU model (discussed in sub-section 2.3) can be evaluated by the communicability evaluation method [30], which usually relies on observational and intrusive mechanisms. However a trade-off between costs and benefits should be carefully considered when conducting QinU studies. SIQinU was envisioned primarily as a non-intrusive strategy from the QinU point of view. (Recall that SIQinU collects user task data from log files that were derived through for example adding snippets of code in a real WebApp-in-use that allow recording user activity).

Results from the analysis in Ph. II are then used to derive EQ requirements for Ph. III. Note that *Conduct preliminary analysis* activity produces the *Preliminary Analysis Report* which is input to the *Establish EQ Information Need* activity in Fig. 9.

Table 9: EQ derived requirements tree and evaluation results of JIRA v.1 (before) and JIRA v.1.1 after implementing improvements on the concrete WebApp (in SIQinU Ph. V). The legend EI stands for Elementary Indicator; P/GI for Partial/Global Indicator

	JIRA v.1		JIRA v.1.1	
Characterisitcs and Attributes	EI	P/G I	EI	P/G I
External Quality		38%		74%
1. Usability		30%		60%
1.1. Learnability		26%		59%
1.1.1. Feedback suitability		38%		38%
1.1.1.1. Navigability feedback completeness	33%		33%	
1.1.1.2. Task progress feedback appropriateness	30%		30%	
1.1.1.3. Entry form feedback awareness	50%		50%	
1.1.2. Helpfulness		15%		80%
1.1.2.1. Context-sensitive help availability	20%		80%	
1.1.2.2. Help completeness	10%		80%	
1.2. Operability		34%		61%
1.2.1. Controllability		80%		80%
1.2.1.1. Permanence of main controls	60%		60%	
1.2.1.2. Stability of main controls	100%		100%	
1.2.2. Error management		0%		30%
1.2.2.1. Error prevention	0%		30%	
1.2.3. Data entry ease		23%		73%
1.2.3.1. Defaults	10%		50%	
1.2.3.2. Mandatory entry	10%		80%	
1.2.3.3. Control appropriateness	50%		90%	
2. Information quality		45%		88%
2.1. Information suitability		45%		88%
2.1.1. Consistency	40%	40%	90%	90%
2.1.2. Information coverage		50%		85%
2.1.2.1. Appropriateness	50%		90%	
2.1.2.2. Completeness	50%		80%	

Examining the relevant QinU problems associated with the task/sub-tasks and screens in the WebApp, and using 2Q2U we (as experts) derived the EQ model which included "Usability", and "Information Quality" characteristics. This concern results in deriving a requirements tree of EQ

attributes for those particular QinU attributes that had problems (see the left hand column of Table 9). Note that in the first three SIQinU phases, in going from QinU requirements to QinU problems and then eventually to EQ requirements as just commented, the quality lifecycle model is thereby deployed and the *depends on* relationship (seen in Fig. 10) is expanded to include attributes of our instantiated models.

To sum up, the selected task *Entering a new defect* in the JIRA case study included 5 sub-tasks, each with 2-3 associated JIRA v.1 screens –as example, one screenshot is shown in Fig. 11. Regarding our proposal to develop relationships between QinU and EQ attributes, designing the task with sub-task components and associated screens for each sub-task is an important and critical undertaking forming the basis for the relationships developed. Doing so enables us (as experts) to map problems found in Ph. II to EQ attributes of information quality, usability, functional quality, among others to decide which attributes, sub-characteristics and characteristics will intervene (Phase III).

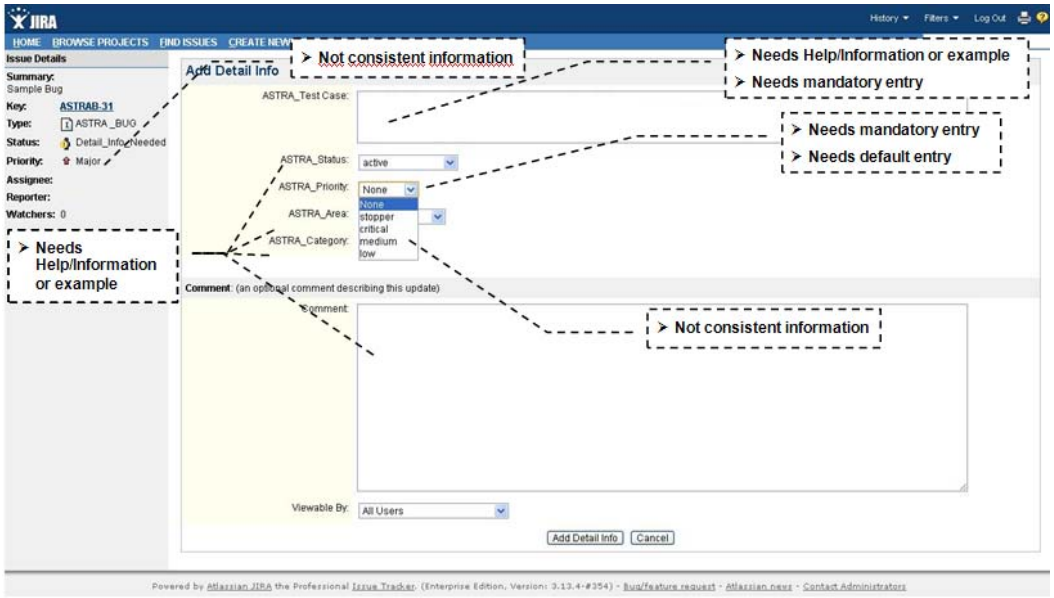


Figure 11: JIRA v.1 *Add Detail Info* screen highlights issues that could impact for sub-task correctness ratings.

In other words, mapping (deriving) is an important aspect of SIQinU as it allows us to identify properties of the system from an EQ point of view that are specifically related to those problems identified from the QinU task/sub-tasks point of view. Basically, this is done through mapping from sub-task to screen, and from screen to properties of the WebApp screens (see Fig. 11) that contain features of "Usability" and "Information quality" to our case study.

Then, in Ph. IV (*Perform EQ Evaluation and Analysis*), we evaluate JIRA v.1 by inspection using the EQ instantiated requirements tree whereby each attribute is evaluated based on the defined indicators. Table 9 shows the results of the EQ evaluation in the JIRA v.1 column. Examining the evaluation and the indicators which show that some attributes perform lowly, we then make recommendations for improvement (see in Table 6, Ph.V: *Recommend and Perform Improvement*

Actions for EQ). These recommendations are given to the M&E project sponsor whereupon evaluators/developers may choose a variety of methods to make the changes depending on the resources they have available. This could range from a complete restructuring of code to simple menu configuration changes.

After the evaluation-driven improvements are made on the WebApp (now named JIRA v.1.1) in Ph.V, we re-evaluate EQ (in Ph. IV) using the same EQ requirements and note the improvement gain. Note that this SIQinU loop between phases V and IV is depicted in Fig. 6. More than likely, some but not all of the improvements will have been made due to resource availability and difficulty to make the recommended changes. This second evaluation leads to a linkage between specific improvements made to the WebApp and the affected EQ attribute which was hopefully improved.

With the improved system, we re-evaluated QinU in Ph. II/VI using the same instantiated QinU requirements tree and note the changes. Hopefully, since there were improvements from the EQ standpoint, there will be improvements in the QinU of the WebApp-in-use, thus enabling us to begin to understand the *influences* relationships between the instantiated EQ and QinU attributes. Table 8 shows the QinU evaluation for the JIRA-in-use v.1.1 showing noticeable improvement in many attributes and an improvement from 53.3% to 67% for the global indicator for "Actual Usability".

Table 10: *Actual usability* evaluated attributes for JIRA v.1 and v.1.1 with improvements gain

Attributes	JIRA v.1	JIRA v.1.1	Change (+ is improvement)
1.1.1. *Sub-Task Correctness*	86.4%	91.9%	5.5%
1.1.2. *Sub-Task Completeness*	87.9%	95.5%	7.6%
1.1.3. *Task Successfulness*	45.5%	72.7%	27.2%
1.2.1. *Sub-Task Correctness Efficiency*	37.4%	44.3%	6.9%
1.2.2. *Sub-Task Completeness Efficiency*	37.5%	47.3%	9.8%
1.2.3. *Task Successfulness Efficiency*	13.1%	36.8%	23.7%
1.3.1. *Sub-Task Correctness Learnability*	78.8%	75.1%	-3.7%
1.3.2. *Sub-Task Completeness Learnability*	26.4%	77.3%	50.9%
1.3.3. *Task Successfulness Learnability*	66.7%	62.5%	-4.2%
		Average Change	13.7%

Comparing each QinU attribute in more detail, Table 10, shows all attributes noted improvement with the exception of *Task Successfulness Learnability* and *Sub-task Correctness Learnability*. 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-in-use v.1.

The next activity of this final phase for this cycle of SIQinU –recall Table 6, activity *iii)*- involves examining possible relationships between EQ and QinU attributes based on our two versions of JIRA. Table 13 (table format of Fig. 12) shows the relationships derived. Relationships derived can then be used as input to the next iteration of SIQinU whereby those identified *depends on* and *influences* can then be purposefully instantiated in Ph. I of subsequent SIQinU cycles.

Note that phases IV to VI, in going from EQ requirements to QinU improvement, the quality lifecycle model is thereby deployed and the *influences* relationship is again exemplified to include specific attributes of our instantiated models with possible degrees of relationship, although not

statistically done at this point. Further statistical studies could be done for instance, by isolating one particular attribute in EQ and QinU and going through the SIQinU cycle of improvement. In illustrating the SIQinU improvement cycle, namely QinU/EQ/QinU and instantiating with the purpose of understanding and improving, we therefore can explore relationships between EQ and QinU not only for the WebApp under study, but for WebApps in general and use the strategy to continue improvement in a consistent way.

Based on this, with our defined task of *Entering a new defect*, Table 11 lists out the "Actual usability" attributes ranked in terms of improvement gain. The higher levels of improvement possibly indicate that changes made at the EQ level had a greater influence. The indicator mapping sets the stage for the interpreting the possible relationships between the EQ attributes and QinU attributes specified in our 2Q2U v2.0 model instantiation. For example, in our mapping, we set greater than 20% improvement to *highly related* (dark gray), 5-20% to *somewhat related* (medium gray), and below 5% to *little or no relationship* (light gray). We chose this calibration as an initial benchmark to be re-calibrated and improved when more historic data is available.

Table 11: Analysis of *Actual usability* attributes ranked by improvement

Ranking by Evaluation	change (+ is improvement)
1.3.2. *Sub-Task Completeness Learnability*	50.9 %
1.1.3. *Task Successfulness*	27.3 %
1.2.3. *Task Successfulness Efficiency*	23.8 %
1.2.2. *Sub-Task Completeness Efficiency*	9.8 %
1.1.2. *Sub-Task Completeness*	7.6 %
1.2.1. *Sub-Task Correctness Efficiency*	6.8 %
1.1.1. *Sub-Task Correctness*	5.6 %
1.3.1. *Sub-Task Correctness Learnability*	-3.7%
1.3.3. *Task Successfulness Learnability*	-4.2%

As depicted in Table 11, the last 2 rated attributes, *Sub-task Correctness Learnability* and *Task Successfulness Learnability*, did not improve. A possible explanation for this is that due to metric design (not shown in this paper), with data collected over a 12 week period that the learning process did not improve as much as expected. It is possible that these beginner users possibly did not ramp up their learning during this time period and that if the case study had been longer, we may have seen different behavior and hence measurements.

Now, we take those with a high level of improvement in QinU, and map those high levels of improvement to the changes made in the WebApp from an EQ viewpoint. Table 12 shows QinU attributes that improved, followed by the EQ attributes that were improved, and the improvement recommendation. Note that following SIQinU, the improvement recommendation is then used as input to designing how to implement the improvement and then executing or carrying out the improvement. For this case study, implementing the improvement was done through changing the JIRA configuration parameters, rather than actually changing any source code. In other instances, depending on the WebApp under evaluation, the tools/methods available to the manager, and the time/resources available, more improvements could be made.

From Table 12, it can be seen that an EQ improvement in *Help completeness* (1.1.2.2) of 70% possibly resulted in tangible real in-use improvements for *Sub-task completeness learnability* of

50.9%, while an EQ improvement in *Context sensitive help* (1.1.2.1) of 60% possibly resulted in tangible real in-use improvements for *Sub-task completeness efficiency* of 9.8%. On the other hand, some EQ changes resulting in EQ improvement showed little or no QinU improvement influence. Thus, we cannot say with 100% certainty that changes made in the properties of JIRA (i.e in its EQ attributes) made a definite impact on a particular QinU attribute, but we can say that it had a positive influence.

The weakness of our analysis is that we are only able to hypothesize the precise relationships between EQ and QinU attributes, but we cannot quantify the exact contribution from each because we made more than one change at a time. If we made only one change, and then measured QinU for JIRA v.1.1, then we could make a more precise hypothesis for a one-to-one or one-to-many relationship. Most likely, those uncovered would be one-to-many, as one-to-one relationships probably are rare in this situation.

Table 12: Changes made from JIRA v.1 to v.1.1 based on recommendations

Related QinU Attribute	QinU Attribute Improvement	EQ Related Attribute	EQ Attribute improvement	Improvement Recommendation
1.3.2 Learnability in use: Sub-task completeness learnability	50.9%	2.1.1 Information quality.InfoSuitability.Consistency	50%	Change fields to have most important information before the details
1.3.2 Learnability in use: Sub-task completeness learnability	50.9%	1.1.2.2 Learnability.Helpfulness.HelpCompleteness	70%	Add context sensitive help
1.1.3 Effectiveness: Task Successfulness	27.3%	1.2.1.2 Operability.Controllability.StabilityofMainControls	0%	Make all valid operations available and disable or don't show invalid-NOT DONE
1.1.3 Effectiveness: Task Successfulness	27.3%	1.1.1.2 Learnability.FeedbackSuitability.Task Progress Feedback Appropriateness	0%	Make the status on left all the time-NOT DONE
1.2.2 Efficiency: Sub-task completeness efficiency	9.8%	1.1.2.1 Learnability.Helpfulness.Context-sensitve help availability	60%	Add context sensitive help
1.2.2 Efficiency: Sub-task completeness efficiency	9.8%	1.2.3.3 Operability.DataEntryEase.ControlAppropriateness	40%	Reduce workload on this substep by using dropdowns instead of text boxes
1.2.2 Efficiency: Sub-task completeness efficiency	9.8%	1.2.3.1 Operability.Data Entry Ease.Defaults	40%	Add defaults
1.2.2 Efficiency: Sub-task completeness efficiency	9.8%	1.2.3.2 Operability.DataEntryEase.MandatoryEntry	70%	Make important fields to completeness mandatory
1.1.2 Effectiveness: Sub-task completeness	7.6%	2.1.1 Information quality.InfoSuitability.Consistency	50%	Reduce workload-remove one comments field
1.1.1 Effectiveness: Sub-task correctness	5.6%	2.1.2.1 Information quality.InfoSuitability.InfoCoverage. Appropriateness	40%	Reduce workload on this substep
1.1.1 Effectiveness: Sub-task correctness	5.6%	1.2.2.1 Operability.Error Mgmt.Error Prevention	30%	Add context sensitive help
1.1.1 Effectiveness: Sub-task correctness	5.6%	1.2.3.1 Operability.Data Entry Ease.Defaults	40%	Change fields to have default and make mandatory because they are critical to defect description correctness and completeness
1.1.1 Effectiveness: Sub-task correctness	5.6%	2.1.1 Information quality.Information suitability.Consistency	50%	Change fields to have most important information before the details
1.1.1 Effectiveness: Sub-task correctness	5.6%	1.1.2.2 Learnability.Helpfulness.HelpCompleteness	70%	Add context sensitive help
1.1.1 Effectiveness: Sub-task correctness	5.6%	2.1.1 Information quality.InfoSuitability.Consistency	50%	Ensure priority codes match
1.1.1 Effectiveness: Sub-task correctness	5.6%	1.2.2.1 Operability.Error Mgmt.Error Prevention	30%	Eliminate non valid platform combinations
1.1.1 Effectiveness: Sub-task correctness	5.6%	1.2.3.3 Operability.DataEntryEase.ControlAppropriateness	40%	Move so view is not blocked
1.1.1 Effectiveness: Sub-task correctness	5.6%	2.1.2.1 Information quality.InfoSuitability.InfoCoverage.Completeness	30%	Add context sensitive help

To be able to quantify relationships with certainty would require many more case studies. Although this is just one case study, we have taken the first step to exploring specific attributes for both EQ and QinU determining where the lines/relationships exist. As such, we can use the results gained in this research to develop hypothesis for *influences* (EQ-QinU) relationships among their attributes and continue to correlate the results in a more definitive statistical way with more studies.

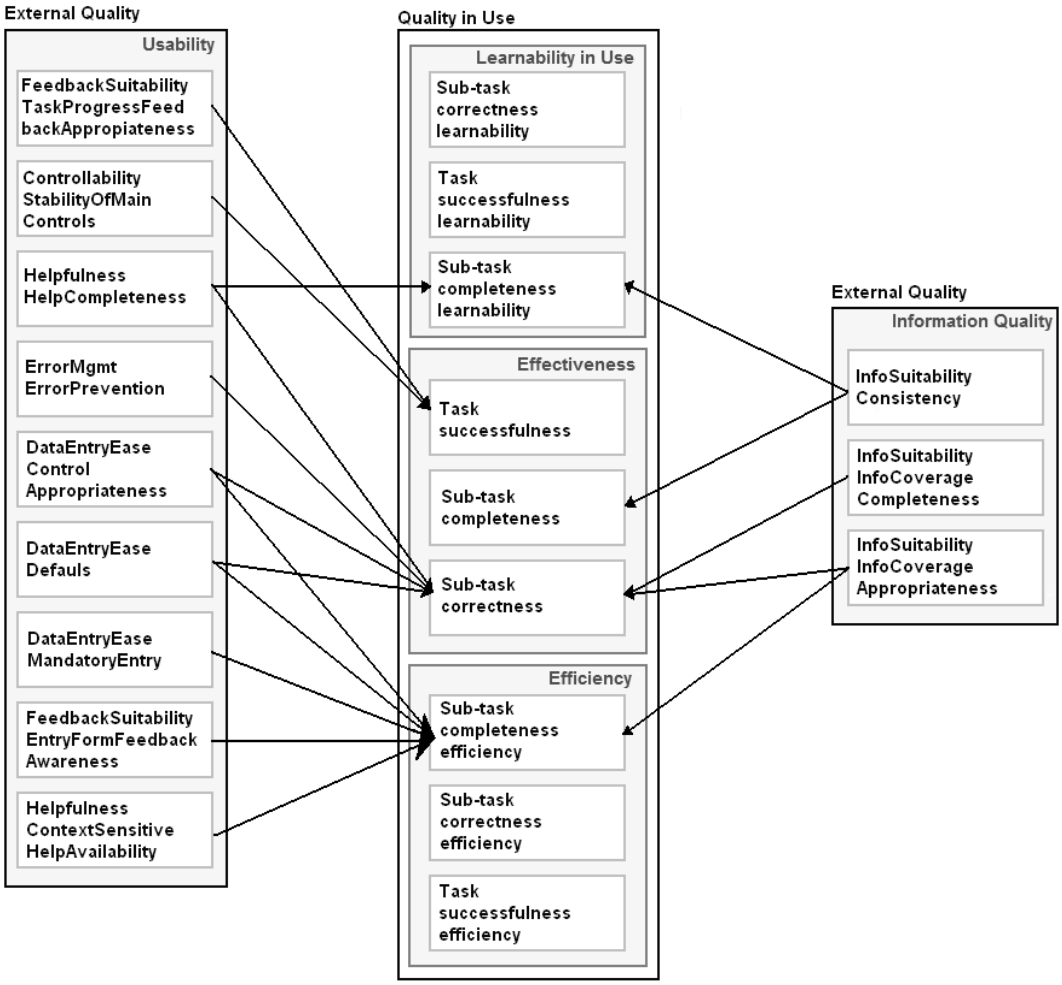


Figure 12: Hypothetical relationships between EQ and QinU attributes

This case study was only the foundation for future work leading to the strength of SIQinU which is that if we wish to continue and achieve greater precision in the relationships between the EQ and QinU attributes, we can continue to use SIQinU with a higher level of granularity by only making one small improvement change at a time in Ph. V, and then measuring the effects on QinU in Phase VI.

Table 13: Table format of Fig. 12 showing the *influence* relationships between EQ and QinU attributes.

EQ Attribute	QinU Attribute
1.1.2.2 Learnability.Helpfulness.HelpCompleteness	Learnability in Use: Sub-task completeness learnability
2.1.1 Information Quality.InfoSuitability.Consistency	
1.2.1.2 Operability.Controllability.StabilityMainControls	Effectiveness: Task Successfulness
1.1.1.2 Learnability.Feedback Suitability.TaskProgressFeedbackAppropriateness	
1.1.1.3 Learnability.Feedback Suitability.EntryFormFeedbackAwareness	Efficiency: Sub-task completeness efficiency
1.1.2.1 Learnability.Helpfulness.Context-sensitiveHelpAvailability	
1.2.3.1 Operability.DataEntryEase.Defaults	
1.2.3.2 Operability.DataEntryEase.MandatoryEntry	
1.2.3.3 Operability.DataEntryEase.ControlAppropriateness	
2.1.1 Information Quality.InfoSuitability.Consistency	Effectiveness: Sub-task completeness
1.1.2.2 Learnability.Helpfulness.HelpCompleteness	Effectiveness: Sub-task correcteness
1.2.2.1 Operability.Error Mgmt.Error Prevention	
1.2.3.1 Operability.DataEntryEase.Defaults	
1.2.3.3 Operability.DataEntryEase.ControlAppropriateness	
2.1.2.2 Information quality.InfoSuitability.InfoCoverage.Completeness	
2.1.2.2 Information quality.InfoSuitability.InfoCoverage.Appropriateness	

Lastly, Fig. 12 and Table 13 summarize the *influence* relationships found in this first case study. Ultimately, recommendations and changes made in Ph. V had an overall positive impact on the real usage of the WebApp-in-use evaluated through the QinU in phases II and VI for JIRA v.1 and JIRA v.1.1 respectively. We can now recommend these changes in other parts (and tasks) of the application. Notably:

- *Add context sensitive help*
- *Change fields to have most important information before the details*
- *Make all valid operations available and disable or don't show invalid choices*
- *Reduce workload by using dropdowns instead of text boxes when possible*
- *Change fields to have default and make mandatory when appropriate*
- *Ensure consistency of information*
- *Eliminate non valid combinations*
- *Ensure view is not blocked from drop-down lists*
- *Let user know the status at all times*

6 Related Work and Discussion

In this paper, we instantiate quality models using a quality modeling framework with the specific purpose of not only understanding but also of improving a WebApp and its use; and then we combine these quality model instantiations with a purpose-oriented strategy to carry out evaluations to ultimately accomplish improvement. As such, based on our examination of existing research, there has been progress in the individual elements such as modeling and evaluation, but limited focus on using in a systematic way tailored models stemming from a quality modeling framework and a tailored strategy for the purpose to improve WebApps and their use while incrementally considering the

QinU/EQ/QinU cycle and taking advantage of the *depends on* and *influences* relationships outlined by ISO 25010.

Regarding improvement strategies by evaluation for WebApps, in [26] authors present an approach for incremental EQ improvement. Their work uses the results from EQ evaluation to make changes and improvements in a WebApp through WMR (*Web Model Refactoring*) but the EQ requirements were not mapped from real QinU problems and also a strategy for continual improvement is only loosely defined. In [8] authors propose a systematic approach to specify, measure and evaluate QinU, but the outcomes were only used to understand the current QinU satisfaction level for an e-learning WebApp, without proposing any improvement strategy. Conversely, the GQM$^+$Strategies approach [1] is an integrated strategy for defining and satisfying measurement goals, but does not give explicit steps to guide the evaluation and improvement. A notable aspect of GQM$^+$Strategies is that gives support to information needs' goals at different organizational levels, i.e. not only at tactic –project- level but also at strategic –business- level. Lastly, in [9] authors present a generic usability evaluation process which can be instantiated into any model-driven web development process, but no improvement strategy is discussed.

Regarding the derivation of EQ characteristics and attributes from QinU requirements and problems, there is a related initiative [21], which focuses on employing a Bayesian method in order to find out *influence* relationships of EQ characteristics on QinU characteristics. However, this work has limited practical benefit because the derivation is theoretical rather than using a real context of use. Moreover, there is no integrated improvement strategy, but rather just a derivation technique.

There are many works aimed at increasing WebApp quality by establishing automated procedures for product improvement during development stages. Meanwhile others use user evaluation at early lifecycle stages. As an example of the former, [5], design patterns that influence quality are included at the conceptual modeling phase and implemented into the WebApp code by means of model transformations. The latter is illustrated by the TRUMP methodology [4], which defines a set of methods to apply to each of the phases and processes described in [15]. This methodology allows evaluating (testing) by a set of users, with an early version of the application, identifying usability problems and then establishing usability requirements for improving the product. Thus in the end, the product is re-evaluated by users to observe whether usability goals were achieved, but there is no strategy for continual improvement through connecting EQ and QinU.

Last but not least, it was highlighted in the Introduction Section as a general contribution of this research that our approach is based on two pillars, namely: i) a *quality modeling framework*; and ii) a *measurement an evaluation strategy*, which in turn is grounded in three principles viz. the C-INCAMI conceptual framework, a well-established process, and methods and tools. Also we indicated in sub-section 3.3 this approach can be adapted for different information needs of an organization, embracing different quality focuses and entities categories such as resource, process, system, system-in-use, in a flexible yet structured manner. Again, based on our examination of existing literature, a full-fledged approach, which considers together and integrates appropriately the above two pillars and principles, has been neglected.

Summarizing in particular the existing research, there lacks attention for using quality modeling frameworks and their instantiation in conjunction with strategies for the goal of improvement. Given

that, this work aims at using the 2Q2U modeling framework –which is a subset of the general quality framework introduced in subsection 2.1- to instantiate quality models for both QinU and EQ, followed by a specific purpose-oriented strategy that uses these models and purposely performs evaluations with the end goal in mind: improving the WebApp and its use in a real context. And through the improvement cycle, potential relationships are drawn between EQ and QinU which are useful not only for the improvement of the WebApp under study, but also possibly applicable to other WebApps leading to further research areas.

7 Concluding Remarks

SIQinU can iteratively be used with increasing levels of granularity by making singular, or related improvement changes in Ph.V, and then measuring the consequence (*influences*) on QinU in Ph.VI. With the ultimate objective to improve the QinU of WebApps-in-use, we first used 2Q2U to purposely instantiate models for both QinU and EQ as per one of our main contributions. For EQ, our model included *information quality*, a characteristic proposed in an earlier work [18] to supplement ISO 25010 to account for the particular quality characteristics of WebApps. For QinU, our model included *learnability in use*, also proposed in that work to supplement ISO 25010 to account for the time dimension of learning and for the specific task being carried out. On the other hand, 2Q2U was envisioned as a subset of a general quality modeling framework, which connects target entity categories with quality models and their relationships.

Other contribution is the proposed SIQinU. SIQinU uses the purposefully instantiated quality models from 2Q2U as a starting point, and then implements a consistent and repeatable strategy to improve the QinU of a WebApp-in-use. SIQinU's six phases can be used to iteratively evaluate from both QinU and EQ points of view with the ultimate goal of making improvement. By employing the SIQinU's C-INCAMI conceptual framework and process, consistency and repeatability are guaranteed for each iteration for continued improvement. To illustrate its practicality, we employed SIQinU in a case study using JIRA, a well-known defect tracking system. QinU was measured and evaluated using a task specifically designed to collect information at the sub-task level so that specific screens and their properties (EQ attributes) could be identified for potential problems leading to poor performance in QinU.

Our final contribution results from carrying out SIQinU, where we were able to map EQ characteristics and attributes to QinU attributes with the goal of ultimately achieving real improvement not only for JIRA but for WebApps and software design in general. Based on this premise, we used these relationships and the improvements made to develop a list of recommendations for improving WebApps. These relationships (currently at exploratory stage) and list of improvements can be further validated through additional case studies in the formation of one-to-one and one-to-many relationships between improvements in EQ and resulting effects in QinU.

Acknowledgements

This paper is a substantial extension made of that published in [16] considering the kind invitation of ICWE 2011 chairs to resubmit the enhanced manuscript. In addition, we thank the support given by Science and Technology Agency of Argentina, in the PAE-PICT 2188 project at Universidad Nacional de La Pampa. We also thank the State Key Lab of Software Development Environment, Beijing University of Aeronautics and Astronautics, Beijing, 100191, China; Supported Project (No.SKLSDE-2010ZX-16).

References

1. Basili V., Lindvall M., Regardie M., Seaman C., Heidrich J., Munch J., Rombach D., Trendowicz A.: Linking software development and business strategy through measurement. IEEE Computer, 43(4), pp. 57–65, 2010.
2. Becker P., Molina H., Olsina L.: Measurement and Evaluation as quality driver. In: Journal ISI (Ingénierie des Systèmes d'Information), Special Issue "Quality of Information Systems", Lavoisier, Paris, France, 15(6), pp. 33-62, 2010.
3. Bevan N.: Extending quality in use to provide a framework for usability measurement. In: LNCS 5619, Springer, HCI Int'l 2009, San Diego, USA, pp. 13-22, 2009.
4. Bevan N., Bohomolni I.: Incorporating user quality requirements in the software development process. In: Proceedings of 4th International Software Quality Week Europe, Brussels, pp. 1192-1204, 2000.
5. Brambilla M., Comai S., Fraternali P., Matera M.: Designing Web Applications with WebML and WebRatio. In: Web Engineering. Modelling and Implementing Web Applications. Springer HCIS, Ch. 9, pp. 221-261, 2008.
6. Burton M., Walther J.: The value of Web log data in use-based design and testing. Journal of Computer-Mediated Communication, 6(3), 2001.
7. Cappiello C., Daniel F., Koschmider A., Matera M., Picozzi M.: A Quality Model for Mashups. In: LNCS 6757, Springer, Web Engineering, 11th Int'l Conference on Web Engineering (ICWE2011), Auer S., Díaz O., Papadopoulos G. (Eds.), Paphos, Cyprus, pp. 137-151, 2011.
8. Covella G., Olsina L.: Assessing Quality in Use in a Consistent Way. In ACM proceedings, Int'l Congress on Web Engineering, (ICWE'06), SF, USA, pp. 1-8, 2006.
9. Fernandez A., Insfran E., Abrahão S.: Integrating a Usability Model into a Model-Driven Web Development Process. In: LNCS 5802, Springer, 10th International Conference on Web Information Systems Engineering (WISE 2009), pp. 497-510, 2009.
10. Hassenzahl M.: User experience: towards an experiential perspective on product quality, IHM; V.339, Proc. 20th Int'l Conference of the Assoc. Francophone d'Interaction Homme-Machine, pp 11-15, 2008.
11. ISO/IEC 25010: Systems and software engineering. Systems and software Quality Requirements and Evaluation (SQuaRE). System and software quality models, 2011.
12. ISO/IEC CD 25010.3, Systems and software engineering. Systems and software Quality Requirements and Evaluation (SQuaRE). System and software quality models, 2009
13. ISO/IEC 25012: Software engineering -- Software product Quality Requirements and Evaluation (SQuaRE) -- Data quality model, 2008
14. ISO/IEC 9126-1. International Standard, Software Engineering - Product Quality - Part 1: Quality Model, 2001.
15. ISO 13407: User centered design process for interactive systems, 1998.

16. Lew P., Olsina L.: Instantiating Web Quality Models in a Purposeful Way, In: LNCS 6757, Springer, Web Engineering, 11th Int'l Conference on Web Engineering (ICWE2011), Auer S., Díaz O., Papadopoulos G. (Eds.), Paphos, Cyprus, pp. 214-229, 2011.
17. Lew P., Olsina L., Becker P., Zhang, L.: An Integrated Strategy to Understand and Manage Quality in Use for Web Applications. Requirements Engineering Journal, Springer, 16 (3), pp. 1-32, DOI 10.1007/s00766-011-0128-x, 2011.
18. Lew P., Olsina L., Zhang L.: Quality, Quality in Use, Actual Usability and User Experience as Key Drivers for Web Application Evaluation, In: LNCS 6189, Springer, 10th Int'l Congress on Web Engineering (ICWE2010), Vienne, Austria, pp. 218-232, 2010.
19. METI, Ministry of Economy - Trade and Industry - Japan, Software Metrics Advanced Project, Investigative Report on Measure for System/Software Product Quality Requirement Definition and Evaluation, March 2011.
20. Molina, H., Olsina, L.: Assessing Web Applications Consistently: A Context Information Approach. IEEE CS, 8th Int'l Congress on Web Engineering, NY, US, pp. 224-230, 2008.
21. Moraga M.A, Bertoa M.F., Morcillo M.C., Calero C., Vallecillo A.: Evaluating Quality-in-Use Using Bayesian Networks. In Proc. of QAOOSE 2008, Paphos, Cyprus, 2008.
22. Murugesan S.: Web Application Development: Challenges and the Role of Web Engineering. Chap. 2 in Springer HCIS, Web Engineering: Modeling and Implementing Web Applications. Rossi G, Pastor O, Schwabe D, Olsina L (Eds.), pp. 7-32, 2008.
23. Nielsen, J: Involving Stakeholders in User Testing, Jakob Nielsen's Alertbox, May 24, 2010, Available at http://www.useit.com/alertbox/utest-observers.html, accessed in Jan, 2011
24. Olsina L., Sassano R., Mich L. Towards the Quality of Web 2.0 Applications, In proc. of 8th Int'l Workshop on Web-oriented Software Technology (IWWOST 2009) held at Int'l Congress on Web Engineering (ICWE09), San Sebastian, Spain, V. 493, pp. 3-15, CEUR (ceur-ws.org), ISSN 1613-0073, 2009.
25. Olsina L., Papa F., Molina H.: How to Measure and Evaluate Web Applications in a Consistent Way. In: Modelling and Implementing Web Applications. Springer HCIS, Ch. 13, pp. 385–420, 2008.
26. Olsina L., Rossi G., Garrido A., Distante D., Canfora G.: Web Applications Refactoring and Evaluation: A Quality-Oriented Improvement Approach, Journal of Web Engineering, Rinton Press, US, 7 (4), pp. 258-280, 2008.
27. Olsina, L.; Lafuente, G. Pastor, O., Towards a Reusable Repository of Web Metrics, Journal of Web Engineering, Rinton Press, US, 1 (1), pp. 61-73, ISSN 1540-9589, 2002.
28. Olsina L., Rossi G.: Measuring Web Application Quality with WebQEM, IEEE Multimedia, 9 (4), pp. 20-29, 2002.
29. Papa M.F. Becker P., Olsina L.: Assessing Integrated Measurement and Evaluation Strategies: A Case Study, To appear in: IEEE Xplore proceeding: 7th Central Eastern European Software Engineering Conference (CEE-SECR 2011), Moscow, Russia, 2011.
30. Prates, R., de Souza, C.; Barbosa, S.: A method for communicability evaluation of user interfaces. Interactions, ACM, 7(1), pp. 31-3, 2000.
31. SPEM. Software Process Engineering Metamodel Specification. Doc./02-11-14., Ver.1.0, 2002.