Instantiating Web Quality Models in a Purposeful Way

Philip Lew[1] and Luis Olsina[2]

[1] School of Software, Beihang University, China
[2] GIDIS_Web, Engineering School, Universidad Nacional de La Pampa, Argentina
philiplew@gmail.com, olsinal@ing.unlpam.edu.ar

Abstract. Web applications and their quality evaluation has been the subject of abundant research. However, models have been used mostly for the purpose of understanding, rather than improving. In this work, we propose utilizing a quality modeling framework to instantiate quality models with the specific purpose to not only to understand the current situation of an entity, but also to improve it. Our approach instantiates models for both external quality and quality in use, resulting in a requirements tree for both followed by evaluation and then combined with a mechanism to develop relationships between them. Hence, improving is driven by understanding these relationships, namely, 'depends on', and 'influences' in alignment with the ISO 25010 quality life cycle model. This is illustrated with a case study showing the underlying strategy from model instantiation to application improvement.

Keywords: Quality improvement, quality in use, actual usability, external quality, SIQinU strategy.

1 Introduction

Today's web-based applications (WebApps) containing complex business logic and sometimes critical to operating the business, are now requiring increased focus on understanding and improving their quality. One of the first steps to evaluate quality is to define nonfunctional requirements usually through quality models. The ISO 25010 standard [11] describes one such model for general usage in specifying and evaluating software quality requirements. However, ISO 25010 is intended as a general guideline 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 measuring and evaluating quality in use (QinU).

But the main goal in evaluating software quality is to ultimately improve. However, independent from models, conducting evaluations in real situations particularly for QinU is difficult to realize. Therefore, a key issue is 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 external quality (EQ) characteristics and attributes. That is to say, does the software's new (and improved) version have a positive impact on its QinU?

To answer this question for WebApps, we began by first proposing to augment the ISO 25010 standard through using the 2Q2U (*Quality, Quality in use, actual Usability* *and User experience*) modeling framework [12] to include *information quality* as a characteristic of internal/EQ because this is a critical characteristic of WebApps. We further proposed to include *learnability in use* as a characteristic of *usability in use* 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.

For this reason, we devised SIQinU (*Strategy for Improving Quality in Use*), a strategy for improving quality that also uncovers these relationships in a systematic way. Starting with QinU, we design specific 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 is completed and evaluated again for its EQ to establish a delta from the initial benchmark. Then we re-evaluate QinU to determine the improvements resulting in QinU from the improvements made at the EQ level, thus leading to a cyclic 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 application 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 is a paramount characteristic 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*) framework [15], which enables us to instantiate a project (our project consisted of improving a concrete actual WebApp) that includes defining the information need, user viewpoint, entity, and so forth. Ultimately, the contributions of this research are:

- A procedure to concretely instantiate quality models based on 2Q2U and C-INCAMI for both QinU and EQ for the purpose of not only understanding but also improving a WebApp.
- A concrete strategy, SIQinU, combined with a well-defined and established process to guide the improvement with an illustration of the improvement results through a case study.
- An exploration of the relationships outlined in the 25010 quality lifecycle model, namely influences and depends, gained through the process of improvement in both EQ and QinU instantiated views.

Following this introduction, Section 2 reviews recent related work and delineates opportunities for improvements which are the motivation for this research. Section 3 demonstrates our procedure for using 2Q2U to purposefully instantiate models for improving a WebApp, which can then be utilized by SIQinU to improve a concrete WebApp. In Section 4, we use 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 5 draws our main conclusions and outlines future work.

2 Related Work and Motivation

In this paper, we instantiate quality models for the purpose of improvement and then combine these instantiations with a 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 a strategy and tailored models for the purpose of improvement considering the QinU/EQ/QinU cycle. Regarding improvement strategies by evaluation for WebApps, in [16] 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 lacks a strategy for continual improvement. In [7] 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. Lastly, in [8] 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 [13], which focuses on employing a Bayesian method in order to find out influences 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 achieved 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 [10]. 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.

Summarizing the existing research, there lacks attention for models and their instantiation for the goal of improvement. Given that, this work aims to use the 2Q2U framework [12] to instantiate quality models for both EQ and QinU, followed by a strategy that uses these models and purposely performs evaluations with the end goal in mind: improving the WebApp. 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.

3 Instantiating Quality Models in a Purposeful Way

Our view for this research is that understanding is the means and improvement is the ultimate goal. With modeling and evaluation as steps toward improvement, models must be instantiated with this in mind. So, starting with modeling, we use the 2Q2U modeling framework to instantiate a concept model for both QinU and EQ. With 2Q2U, we include *learnability in use* as a sub-characteristic of QinU and *information quality* as characteristic of EQ. We also define two new concepts *actual user experience* and *actual usability* where the latter encompasses the 'do goals' of the user, as defined by ISO 25010 usability, with the exception of the *satisfaction* sub-characteristic related to 'be goals' and modeled as a characteristic of actual user experience [3, 9]. Details on these definitions and relationships can be found in [12].

To use the instantiated models for measurement and evaluation (M&E) in a consistent and repeatable way, we utilize the C-INCAMI [15] framework. The C-INCAMI framework defines all of the concepts and relationships needed to design and implement M&E processes. C-INCAMI's approach is designed to satisfy a specific information need in a given context defining concepts and relationships that are used along all the M&E activities lending to consistent analysis and results. 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 use for this research in instantiating quality models for the purposes of improvement is C-INCAMI's Nonfunctional Requirements Specification (NFRS), and Context Specification component, shown in Figure 1.

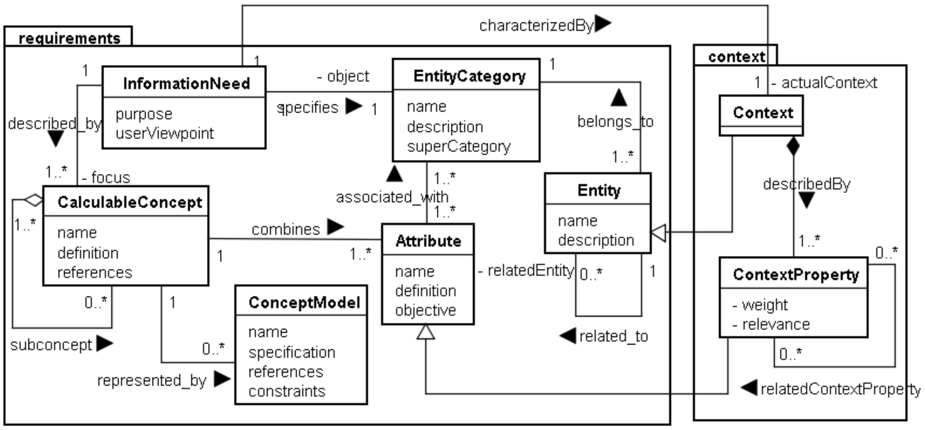


Fig. 1. C-INCAMI Nonfunctional requirements and context specification components

The *NFRS* specifies the *Information Need* of any M&E project; that is, the *purpose* (e.g. "understand", "predict", "improve", etc.) and the *user viewpoint* (e.g. "developer", "final user", etc). In turn, it focuses on a *Calculable Concept* and specifies the *Entity Category* to evaluate –e.g. a resource, process, product, etc.-, by means of a concrete *Entity* –e.g., the amazon.com shopping basket. A calculable concept can be defined as an abstract relationship between attributes of an entity and a given information need. In our case, the purpose is to improve, a WebApp, regarding its EQ and QinU, and its actual usability as the calculable concept. This can be represented by a *Concept Model* where the leaves of an instantiated model (e.g. a requirements tree) are *Attributes* associated with an *Entity*.

Context Specification delineates the state of the situation of the entity to be assessed with regard to the information need. *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, life-cycle process-, the organization or the project itself, among others. In our case, the context is particularly important regarding QinU requirements as instantiation of requirements must be done consistently in the same context.

For this paper, regarding SIQinU, we utilize 2Q2U to create the models combined with the NFRS and Context Specification components of C-INCAMI to conduct the evaluations as an integral part of the strategy. Using the C-INCAMI terminology and framework as shown in Figure 1, the information need is to understand and improve, while the entity is a particular WebApp who's category is Web Application. In particular, later in the case study illustration, we exhibit an actual WebApp as well. The concept model used is the instantiated 2Q2U framework shown in Figure 3 while the calculable concepts are *usability in use*, *information quality*, and *operability*. In particular, our objective is to understand both the current and future evaluations of the entity in order to determine improvement after changes were made to the current version. Employing the 2Q2U framework, we purposely instantiate it with operability and information quality as EQ characteristics to be related to the QinU characteristic, actual usability combined with sub-characteristics: *efficiency in use*, *effectiveness in use*, and *learnability in use* as shown in Figure 3.

Table 1. SIQinU Phases, activities and work products

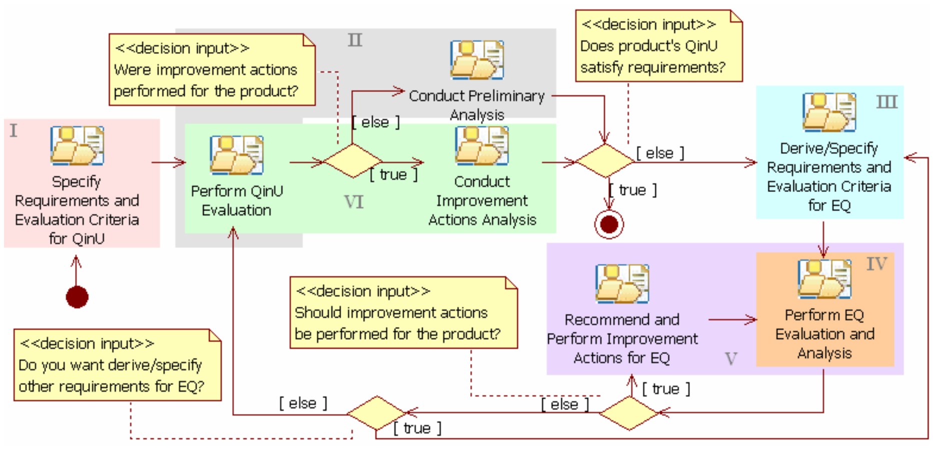
Phase (Ph.)	Description/Activities	Work Products
Ph. I Specify Requirements and Evaluation Criteria for QinU	Taking into account the recorded data of the WebApp's usage, we re-engineer QinU requirements. This embraces designing tasks, defining user type, specifying usage context and characteristics, particularly, for *actual usability* as defined in [12]. Activities include: i) Establish Information Need; ii) Specify Project Context; iii) Design Tasks; iv) Select QinU Concept Model; v) Design QinU Measurement and Evaluation; vi) Design Preliminary Analysis	- Information Need and Context specification -QinU NFRS tree -QinU Metrics and Indicators specification -Task/sub-tasks specification
Ph. II Perform QinU Evaluation and Conduct Preliminary Analysis	As per Ph. I, data is collected purposely targeting QinU attributes for improvement. Depending on the WebApp's ability to collect the data, we also collect the date/time the data is gathered, errors, task and sub-task completion and accuracy, etc. It includes: i) Collect and parse data pertaining to tasks with their sub-tasks; ii) Quantify QinU Attributes; iii) Calculate QinU Indicators; iv) Conduct preliminary analysis.	-Measure and indicator values for QinU -QinU preliminary analysis report
Ph. III Derive/ Specify Requirements and Evaluation Criteria for EQ	Based on Ph. I and II, we derive EQ requirements, i.e. characteristics and attributes, with their metrics and indicators in order to understand the current WebApp's quality. In our case study, a focus on *operability* and *information quality* was used to determine possible effects and specifically improve *actual usability*. Activities include: i) Determine EQ Concept Model; ii) Design EQ Measurement; iii) Design EQ Evaluation	-EQ NFRS tree -EQ Metrics and Indicators specification
Ph. IV Perform EQ Evaluation and Analysis	Activities include: 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
Ph. V Recommend, Perform Improvement Actions, and Re-evaluate EQ	Using the EQ attributes that require improvement, we make improvement recommendations for modifying the WebApp, i.e. version 1 to 1.1. Activities include: i) Recommend improvement actions; ii) Design Improvement Actions; iii) Perform Improvement Actions; iv) Evaluate Improvement Gain to note improvement from benchmark in Ph. IV	-EQ Recommendations report - Improvement plan -New app version -EQ Analysis report
Ph. VI Re-evaluate QinU and Analyze Improvement Actions	Once the new version has been used by real users, we evaluate QinU again to determine the influence of what was improved for the WebApp's EQ on QinU. This provides insight to further develop the *depends* and *influences* relationships. Activities include: i) Evaluate QinU again to determine level of improvement from Ph.II; ii) Conduct Improvement Action analysis; iii) Develop depends and influences relationships between EQ improvements and QinU.	-Measure and indicator values for QinU -QinU Improvement analysis report -EQ/QinU attribute relationship table

Note that we could have selected other characteristics for both EQ and QinU, but we purposely instantiated these characteristics as per our information need via the C-INCAMI NFRS component. Later in Section 4, we fully derive the requirements tree for each of the sub-characteristics at a more detailed level in illustrating the case study.

The proposed SIQinU strategy utilizes the 2Q2U framework for quality models and the C-INCAMI 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 [2] and methods that are consistent and repeatable.

SIQinU collects user behavior data from log files [6] that were derived through for example adding snippets of code in a real WebApp-in-use that allow recording that data, with an 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 we derive EQ requirements that can affect QinU, and propose recommendations for improvements. 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 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.

SIQinU's phases are illustrated in Figure 2, combined with Table 1 which provides a brief description with Phase (Ph.) reference numbers, activities and work products.



<<decision input>> Were improvement actions performed for the product?
II
Conduct Preliminary Analysis
<<decision input>> Does product's QinU satisfy requirements?
I
Specify Requirements and Evaluation Criteria for QinU
Perform QinU Evaluation
[else]
[true]
VI
Conduct Improvement Actions Analysis
[else]
[true]
III
Derive/Specify Requirements and Evaluation Criteria for EQ
IV
Perform EQ Evaluation and Analysis
Recommend and Perform Improvement Actions for EQ
V
<<decision input>> Should improvement actions be performed for the product?
<<decision input>> Do you want derive/specify other requirements for EQ?
[else]
[true]
[else]
[true]

Fig. 2. Process overview for understanding and improving quality in use (SIQinU)

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. In addition, we can continue to iterate the SIQinU improvement cycle to gain further insight and granularity adding a temporal component for later study.

4 A Quality Improvement Lifecycle Using SIQinU: A Case Study

This section illustrates the quality improvement lifecycle via SIQinU using excerpts of a case study conducted in mid-2010. It examined JIRA (www.atlassian.com), a defect reporting WebApp in commercial use in over 14,500 organizations in 122 countries. JIRA's most common task, *Entering a new defect*, was evaluated in order to provide the most benefit, since entering a new defect represents a large percentage of the total usage of the application. We studied 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. Next, we discuss some aspects of the above 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 entity (JIRA v.1) from the *beginner user* viewpoint performing the above mentioned task. To do this, we design the QinU requirements as shown in the right part of Figure 3, resulting in the requirements tree shown in the column 1 of Table 2.

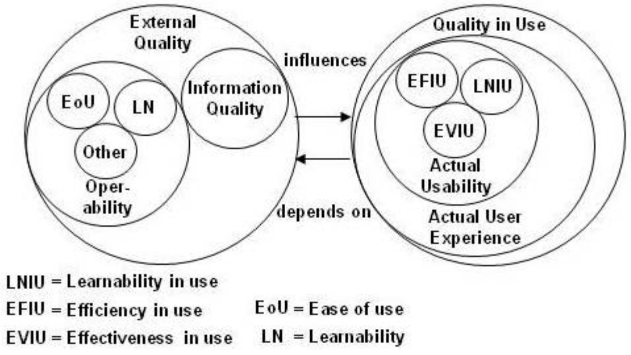
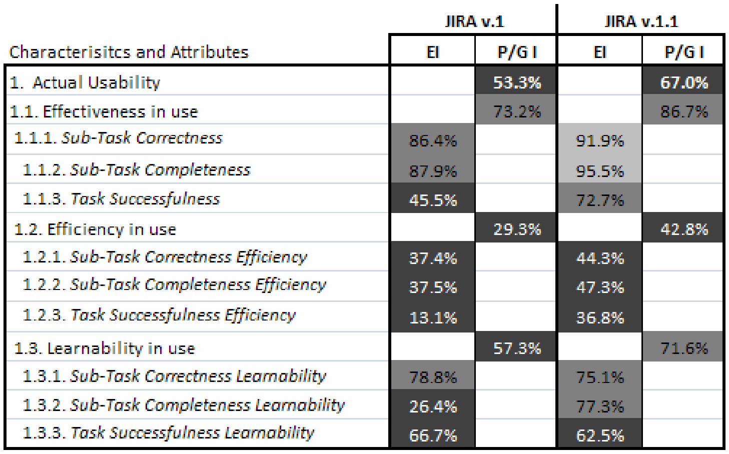


Fig. 3. 2Q2U model instantiation with some EQ and QinU characteristics shown

Using requirements from Ph. I of our instantiated model, we conduct the QinU evaluation as Ph. II of SIQinU. This evaluation is done for each attribute specified in Ph. I, and results in a global evaluation of 53.3% as shown in Table 2. Note that in *designing the QinU evaluation* (Ph. I activity shown in Table 1) for each attribute of the requirements tree we specified an elementary indicator 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 2. QinU requirements tree and evaluation results of JIRA v.1 (before) and JIRA v.1.1 (after modifications). 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 in use		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 in use		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 list of problem areas (those attributes that rated unsatisfactory). Note that other complementary techniques could be used as traditional observational techniques [14] to understand user problems while interacting with the selected tasks and to help deriving EQ attributes. However a trade-off between costs and benefits should be carefully considered.

Results from Ph. II (from the *conduct preliminary analysis* activity) are then used to derive EQ requirements for Ph. III. 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 *operability*, and *information quality* characteristics. This results in deriving a requirements tree of EQ attributes for those particular QinU attributes that had problems (see the left hand column of Table 3).

Note that in the first 3 phases, in going from QinU requirements to QinU problems and then eventually to EQ requirements as just discussed, the quality lifecycle model is thereby deployed and the "depends on" relationship (see Fig. 3) is expanded to include attributes of our instantiated models. Then, in Ph. IV, we evaluate JIRA v.1 by inspection using the EQ instantiated requirements tree whereby each attribute and metric is calculated and evaluated based on the defined indicators. Table 3 shows the results of the EQ evaluation in columns 2-3.

Examining the evaluation and the indicators which show that some attributes perform lowly, we then make recommendations for improvement. These

Table 3. EQ derived requirements tree and evaluation results of JIRA v.1 (before) and JIRA v.1.1 after implementing improvements on the concrete WebApp

	JIRA v.1		JIRA v.1.1	
Characterisitcs and Attributes	EI	P/G I	EI	P/G I
External Quality		38%		74%
1. Operability		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. Ease of use		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%	
. . .	. . .		. . .	

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.

In Ph.V, after the evaluation-driven improvements are made on the WebApp (now named JIRA v.1.1), we re-evaluate EQ using the same EQ requirements and note the improvement gain. 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 application, we re-evaluated QinU using the same requirements and note the changes. Hopefully, since there were improvements from the EQ standpoint, there will be improvements in the QinU of the WebApp, thus enabling us to begin to understand the relationships between the EQ and QinU attributes instantiated in our model development. Table 2 shows the QinU evaluation for the improved WebApp version of JIRA v.1.1 showing noticeable improvement in many attributes and an improvement from 53.3% to 67% for the global indicator for *actual usability*. Comparing each attribute in more detail, Table 4, 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 v.1.

Table 4. 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%

The next activity of this final phase (recall Table 1) for this cycle of SIQinU involves examining possible relationships between EQ and QinU attributes based on our two versions of JIRA. Table 7 shows an excerpt of 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.

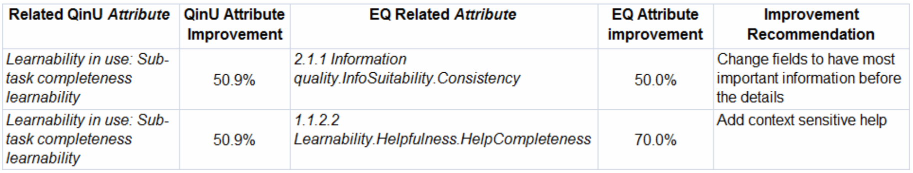
Note that Phases IV to VI, in going from EQ requirements to QinU improvement, the quality lifecycle model is thereby deployed and the 'influences' relationship is again exemplified to include specific attributes of our instantiated models with possible degrees of relationship, although not statistically 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 5 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.

Table 5. Analysis of actual usability attributes ranked by improvement

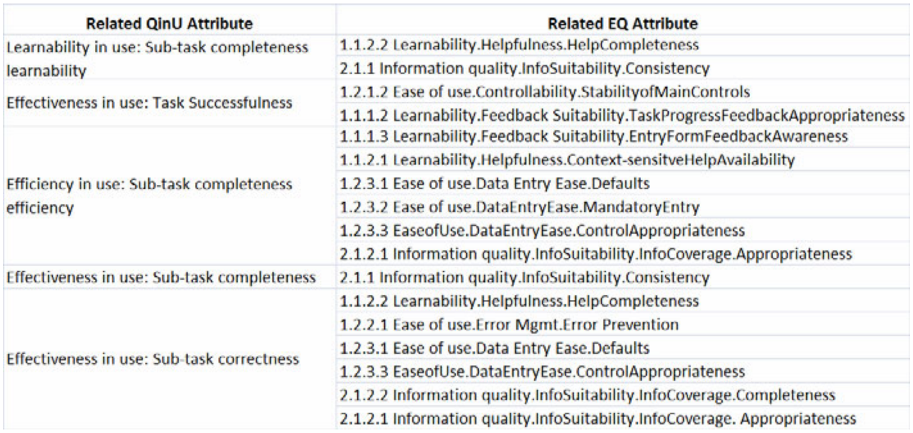
Ranking by Evaluation	change (+ is improvement)
1.3.2. *Sub-Task Completeness Learnability*	50.9 %
1.1.3. *Task Successfulness*	27.3 %
1.2.3. *Task Successfulness Efficiency*	23.8 %
1.2.2. *Sub-Task Completeness Efficiency*	9.8 %
1.1.2. *Sub-Task Completeness*	7.6 %
1.2.1. *Sub-Task Correctness Efficiency*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

Table 6. Changes made from JIRA v.1 to v.1.1 (excerpt) based on recommendations



Related QinU *Attribute*	QinU Attribute Improvement	EQ Related *Attribute*	EQ Attribute improvement	Improvement Recommendation
Learnability in use: Sub-task completeness learnability	50.9%	*2.1.1 Information quality.InfoSuitability.Consistency*	50.0%	Change fields to have most important information before the details
Learnability in use: Sub-task completeness learnability	50.9%	*1.1.2.2 Learnability.Helpfulness.HelpCompleteness*	70.0%	Add context sensitive help

Table 7. Relationships developed during the JIRA case study for EQ and QinU attributes



Related QinU Attribute	Related EQ Attribute
Learnability in use: Sub-task completeness learnability	1.1.2.2 Learnability.Helpfulness.HelpCompleteness
	2.1.1 Information quality.InfoSuitability.Consistency
Effectiveness in use: Task Successfulness	1.2.1.2 Ease of use.Controllability.StabilityofMainControls
	1.1.1.2 Learnability.Feedback Suitability.TaskProgressFeedbackAppropriateness
Efficiency in use: Sub-task completeness efficiency	1.1.1.3 Learnability.Feedback Suitability.EntryFormFeedbackAwareness
	1.1.2.1 Learnability.Helpfulness.Context-sensitveHelpAvailability
	1.2.3.1 Ease of use.Data Entry Ease.Defaults
	1.2.3.2 Ease of use.DataEntryEase.MandatoryEntry
	1.2.3.3 EaseofUse.DataEntryEase.ControlAppropriateness
	2.1.2.1 Information quality.InfoSuitability.InfoCoverage.Appropriateness
Effectiveness in use: Sub-task completeness	2.1.1 Information quality.InfoSuitability.Consistency
Effectiveness in use: Sub-task correctness	1.1.2.2 Learnability.Helpfulness.HelpCompleteness
	1.2.2.1 Ease of use.Error Mgmt.Error Prevention
	1.2.3.1 Ease of use.Data Entry Ease.Defaults
	1.2.3.3 EaseofUse.DataEntryEase.ControlAppropriateness
	2.1.2.2 Information quality.InfoSuitability.InfoCoverage.Completeness
	2.1.2.1 Information quality.InfoSuitability.InfoCoverage. Appropriateness

The indicator mapping sets the stage for the interpreting the possible relationships between the EQ attributes and QinU attributes specified in our 2Q2U 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.

As depicted, 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 for space reasons), 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.

By taking those attributes with a high level of improvement in QinU and mapping those high levels of improvement to the changes made in the WebApp from an EQ viewpoint gives us insight into what particular changes in EQ attributes have an effect on QinU. For instance, Table 6 shows 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%. 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 the application

(EQ attributes) made a definite impact on a particular QinU attribute, but we can say that it had a positive influence.

Regarding the EQ and QinU attributes' relationships, we cannot quantify the exact contribution from each because we made more than one change at a time. However, if we had made only one change, and then measured QinU for JIRA v.1.1, we may have uncovered one to one relationships, although this is unlikely as it is more plausible that several EQ attributes will influence any particular QinU attribute. Table 7 summarizes these relationships developed in this case study. Each of these relationships can be further examined in future case studies.

5 Concluding Remarks

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 consequence (*influences*) on QinU in Ph.VI. Our ultimate objective is to improve the QinU of WebApps-in-use. With this goal in mind, we first used 2Q2U to purposely instantiate models for both EQ and QinU as per one of our main contributions. For EQ, our model included *information quality*, a characteristic proposed in an earlier work [12] to supplement ISO 25010 to account for the particular characteristics of WebApps whose quality is dependent on information quality as well. 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.

Using the purposefully instantiated quality models as a starting point, our second contribution is the proposed SIQinU, a consistent and repeatable strategy to improve the QinU. SIQinU uses the models generated by 2Q2U through its six phases toward evaluating a WebApp from both EQ and QinU points of view with the ultimate goal of making improvement. It also employs the C-INCAMI framework in order to guarantee consistency for usage in an iterative manner for continued improvement. Note that SIQinU relies also on a well-established process [2]. To illustrate its usage, we employed SIQinU in a case study using JIRA, a well-known defect tracking system 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.

Lastly, as our final contribution, in carrying out SIQinU, 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.

Acknowledgments. Thanks to the support by National Basic Research Program of China (No. SKLSDE-2010ZX-16), and PAE-PICT 2188 project at UNLPam, from the Science and Technology Agency, Argentina.

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), 57–65 (2010)
2. Becker, P., Lew, P., Olsina, L.: Strategy to Improve Quality for Software Applications: A Process View. In: To appear in ACM Proceedings of ICSE, Int'l Conference of Software and System Process (ICSSP), Honolulu, Hawaii, USA, May 21-22 (2011)
3. Bevan, N.: Extending quality in use to provide a framework for usability measurement. In: Kurosu, M. (ed.) HCD 2009. LNCS, vol. 5619, pp. 13–22. Springer, Heidelberg (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 Web Engineering. In: Modelling and Implementing Web Applications, Ch. 9, pp. 221–261. Springer HCIS, Heidelberg (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. Covella, G., Olsina, L.: Assessing Quality in Use in a Consistent Way. In: ACM Proceedings, Int'l Congress on Web Engineering (ICWE 2006), SF, USA, pp. 1–8 (2006)
8. Fernandez, A., Insfran, E., Abrahão, S.: Integrating a Usability Model into Model-Driven Web Development Processes. In: Vossen, G., Long, D.D.E., Yu, J.X. (eds.) WISE 2009. LNCS, vol. 5802, pp. 497–510. Springer, Heidelberg (2009)
9. Hassenzahl, M.: User experience: towards an experiential perspective on product quality. In: Proc. 20th Int'l Conference of the Assoc. Francophone d'Interaction Homme-Machine, IHM, vol. 339, pp. 11–15 (2008)
10. ISO 13407: User centered design process for interactive systems (1998)
11. ISO/IEC 25010: Systems and software engineering. Systems and software Quality Requirements and Evaluation (SQuaRE). System and Software Quality Models (2011)
12. Lew, P., Olsina, L., Zhang, L.: Quality, Quality in Use, Actual Usability and User Experience as Key Drivers for Web Application Evaluation. In: Benatallah, B., Casati, F., Kappel, G., Rossi, G. (eds.) ICWE 2010. LNCS, vol. 6189, pp. 218–232. Springer, Heidelberg (2010)
13. 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)
14. Nielsen, J.: Involving Stakeholders in User Testing, Jakob Nielsen's Alertbox (May 24, 2010), `http://www.useit.com/alertbox/utest-observers.html` (accessed in January 2011)
15. Olsina, L., Papa, F., Molina, H.: How to Measure and Evaluate Web Applications in a Consistent Way. In: Modelling and Implementing Web Applications, Ch. 13, pp. 385–420. Springer HCIS, Heidelberg (2008)
16. Olsina, L., Rossi, G., Garrido, A., Distante, D., Canfora, G.: Web Applications Refactoring and Evaluation: A Quality-Oriented Improvement Approach. Journal of Web Engineering 4(7), 258–280 (2008)