Journal of Web Engineering, Vol. 11, No.3 (2012) 209-246

UPDATING QUALITY MODELS FOR EVALUATING NEW GENERATION WEB APPLICATIONS

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 December 18, 2011
Revised June 25, 2012

Web applications (WebApps) and their quality specification and evaluation have been the subject of abundant research. However, the particular features of new generation WebApps pose new challenges regarding current quality models and their included characteristics and sub-characteristics. For instance, functional added value (e.g. integratedness and beneficialness), learnability in use, communicability, sense of community and, ultimately, user experience are characteristics very often neglected in quality modeling or placed appropriately in quality views. Considering the recently issued ISO 25010/25012 standards and quality models for WebApps, in this work we propose updating our previously developed quality models and framework so-called 2Q2U (*Quality*, *Quality in use*, *actual Usability* and *User experience*) in the light of those features of new generation WebApps. The resulting quality models and framework in conjunction with evaluation strategies contribute towards a flexible, integrated approach to measure and evaluate different eras of WebApps. In order to illustrate the approach a practical case for evaluating a mashup WebApp is conducted.

Key words: Quality models, Quality in use, Actual usability, User Experience, Functional quality, Information quality, Evaluation approach.

1 Introduction

WebApps, a combination of information (content), integrated functionalities and services are fast becoming the most predominant form of software delivery today, and therefore require more focused attention in understanding, evaluating, and especially improving their quality. For instance, to evaluate quality for newer generations WebApps such as social networks [15], mashups [6], mobile [21], RIA (*Rich Internet Applications*) [27], among others, requires comprehending in detail how they are different from older generations or from conventional software applications as user requirements, expectations, and behavior can be somewhat different (for a categorization of newer generation WebApps, see [20], and for a characterization of Web 2.0 applications considering non-functional aspects, see e.g. [23]). Consequently, the particular features of new generation WebApps pose new challenges regarding current quality models and their included characteristics and sub-characteristics, as well as the particular attributes or properties to be measured and evaluated.

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 that structure relationships among them. Quality models can be the focus for different entities categories such as *product*, *system*, *system in use*, among others as resource and process. The products are entities at early phases of a software/Web process life cycle (e.g., textual documents or graphical documents such as images, UML diagrams, source code, etc); the information systems are executable products in a specified context (e.g. mobile or mashup WebApps in a testing environment), which can include hardware and software together; and the 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. Hence, new generation WebApps can be considered as *system* or *system in use* from the evaluated entity categories standpoint.

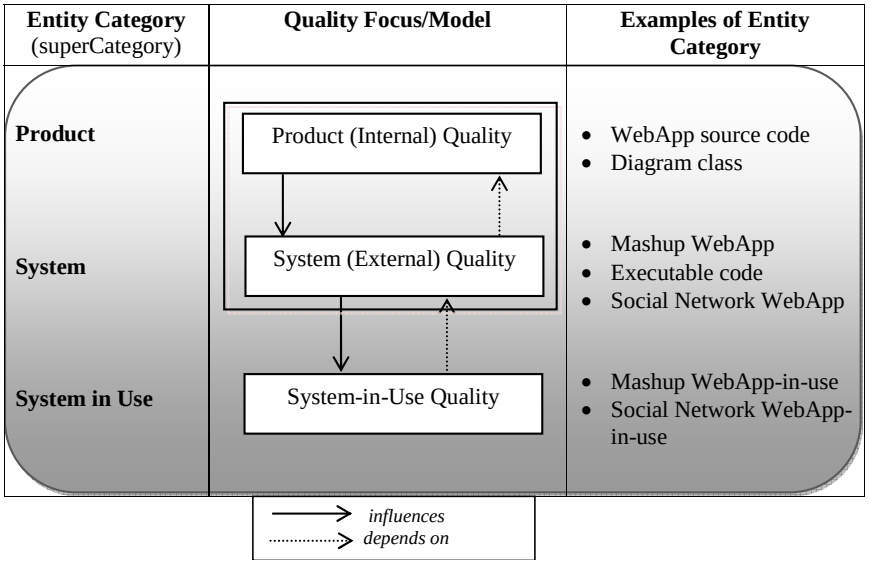


Figure 1: A view of a quality modeling framework regarding target entity (super) categories, quality focuses/models, and relationships between models.

Quality models usually specify product, system or system-in-use quality requirements regarding main *characteristics* that are further subdivided into *sub-characteristics* and *attributes* (or properties). Further, the attributes can be measured by *metrics* and evaluated by *indicators* [24]. For example, ISO standards such as 9126-1 [13] (superseded by 25010 [10]), and other researchers [3, 18, 24] 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 the ISO 25010 standard), can be measured and evaluated by dynamic attributes of the running code, i.e. when a component or full software/WebApp 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, and *depends on* between QinU and EQ. Fig. 1 depicts three columns: The first one represents the super-category of the above mentioned entity categories; the third column illustrates examples of entity categories; and the second one, specifies the quality focus respectively –i.e., the root characteristic of a quality model. Also the quoted relationships between models are depicted.

Due to the evolution of software applications –i.e. systems and systems in use-, characteristics such as *user experience*, 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 place them into quality models; also we observed a lack of consensus in meaning as well. Specifically, reviewing ISO standards at that moment as the 25010 draft standard [11] –officially issued in March, 2011 [10]-, the 25012 standard [12] for data quality requirements, and other current works by researchers in the field of quality in use, usability and user experience such as Bevan [3, 4] and Hassenzahl [8], 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 [11], trying to be as compliant as we could with standards. Summarizing the first version of 2Q2U extension, first, we added two characteristics, namely: *information quality* (for the EQ 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. The rationale of adding or adapting characteristics and concepts in 2Q2U was thoroughly discussed in [17], and will be revisited in the related work section.

However, in light of the officially issued ISO 25010 standard [10], a recently published report [18], and related literature such as [5, 6], which deals with a quality model for mashup WebApps, we updated 2Q2U v1.0 considering also other particular features for new generation WebApps [20]. For example, *functional suitability* (e.g. *integratedness* and *beneficialness*), *communicability*, *sense of community*, etc. are (sub-)characteristics very often neglected in quality models or misplaced in quality views. As result, the 2Q2U v2.0 quality modeling framework was developed, and will be thoroughly discussed in this paper.

Ultimately, the particular contributions of this research are:

- A discussion of strengths and weaknesses of recent issued ISO standards related to software/Web quality models.
- The rationale for including new (sub-)characteristics to the 2Q2U quality modeling framework and their added value for evaluating not only older but also newer generations of WebApps.

- An instantiation, for illustration purposes and to show practical usage of the 2Q2U EQ model with sub-characteristics and measurable attributes to evaluate *functional quality* for mashup WebApps.

As an additional contribution of this research, which was also discussed thoroughly in [22], we highlight how the used *evaluation approach* based on two main pillars, namely: i) a general *quality modeling framework* –where 2Q2U is a subset; and ii) *measurement and evaluation strategies* (which are grounded in turn on three principles viz. a *measurement and evaluation (M&E) conceptual framework*, a *well-established process*, and *methods and tools*), can be adapted for different organizational information needs for different entities categories such as system, system-in-use, resource, etc., in a flexible yet structured manner.

Following this introduction, Section 2 reviews recent related work and delineates opportunities for improvements. Particularly we analyze ISO standards related to system and system-in-use quality models; the 2Q2U v1.0 quality modeling framework, and other up-to-date quality models for new generation WebApps. In Section 3, the foundations for updating the first version of 2Q2U in order to incorporate new features of recent generations WebApps in 2Q2U v2.0 are discussed. In Section 4, we overview the evaluation approach to show how both the quality modeling framework and the employed M&E strategy are used for instantiating models and performing evaluations. This approach is illustrated in Section 5 where the *functional quality* characteristic is instantiated from the EQ viewpoint for the woozor (woozor.com/) mashup WebApp. Finally, Section 6 draws our main conclusions and outlines future work.

2 Related Work and Discussion

This section examines the related work with an eye for improvement opportunities. Particularly, in sub-section 2.1, we describe some strengths and weaknesses of recent issued ISO standards related to software/Web product, system and system-in-use quality models. Then, in sub-section 2.2, we summarize the first version of the 2Q2U quality framework, and provide the reasons for the included characteristics and sub-characteristics to both EQ and QinU models. Also, other up-to-date quality models for new generation WebApps are described in sub-section 2.3, finalizing with the motivation of this research.

2.1 Recent Quality Models in ISO Standards

In the recently issued ISO 25010 standard [10], the concept of product/system quality has been broadened from 6 characteristics (in ISO 9126-1 [13]) to 8, incorporating the same quality characteristics with some amendments. Also it revises the ISO 9126-1 QinU model basically enlarging the number of characteristics from 4 to 5, adding some sub-characteristics as well. In Annex A of [10] (pgs. 21-23), an abridged yet useful comparison with quality models in ISO 9126-1 is made; particularly, table A.1 (cf. [10]) lists the differences between characteristics and sub-

characteristics in 25010 and 9126-1 standards, while also noting the rationale for renaming some of them.

In summary, the 25010 standard is split into 2 quality models. The first is a system/software product (i.e. external/internal [13, 18]) quality model depicted in Fig. 2. In comparison with ISO 9126-1, the *security* characteristic –with 5 totally new sub-characteristics viz. *confidentiality*, *integrity*, *non-repudiation*, *accountability* and *authenticity*-, has been added moving it as a sub-characteristic from the former functionality characteristic. This is a good strategic decision for specifying and evaluating this non-functional requirement for both software and WebApps of different generations. Also the *compatibility* characteristic –with 2 former sub-characteristics viz. *interoperability*, and *co-existence*-, was added.

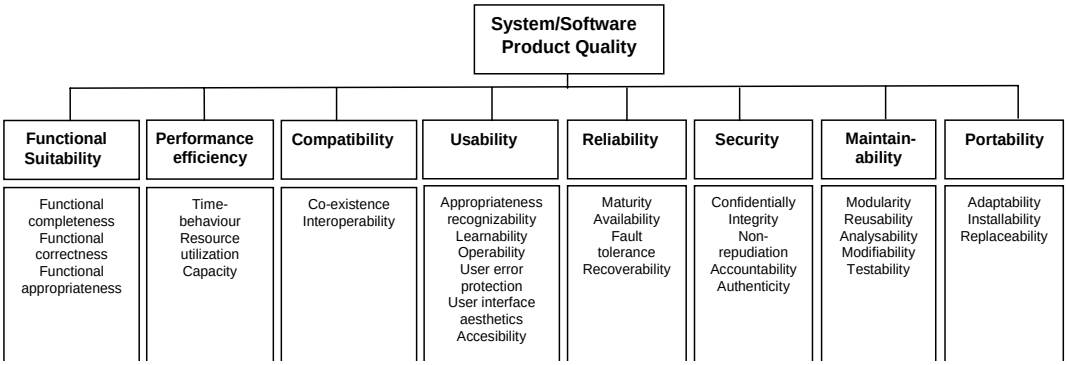


Figure 2: *System/Product Quality* model in ISO 25010 (taken from [10]).

On the other hand, the previous characteristic of *usability* has kept the same name in the system/software quality model but the definition has been rephrased, and also 2 new sub-characteristics –i.e. *user error protection* and *accessibility*- were added. Sub-characteristics such as *learnability* and *operability* have remained while the former understandability name was changed to *appropriateness recognizability*, and likewise attractiveness by the *user interface aesthetic* name. In our humble opinion, a weak aspect is the new definition of *usability* 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 product/system quality, showing a lack of separation of concerns between the meaning of both views, as we will discuss later on.

In addition, we propose that the current *functional suitability* characteristic can be strengthened from both the definition and the included sub-characteristics standpoint, e.g. including the *functional added-value* sub-characteristic which is relevant for some new generation WebApps regarding for instance the *integratedness* attribute, as we will discuss thoroughly in Section 3. Just to grasp the idea, we define *integratedness* as the "degree to which a 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".

The second model in ISO 25010 depicted in Fig. 3 refers to *quality in use* and includes previous

ISO 9126-1 QinU characteristics while adding others. Note that the *effectiveness* and *satisfaction* characteristics from ISO 9126-1 were imported into this newer standard, while productivity has been renamed as *efficiency*, likewise *freedom from risk* characteristic has replaced the older safety name. It is worth mentioning that the two current characteristics viz. *satisfaction* and *freedom from risk* have many prescriptive sub-characteristics. For instance *satisfaction* includes *usefulness*, *trust*, *pleasure* and *comfort* sub-characteristics (see Fig. 3). *Context coverage* is a new ISO 25010 characteristic, but we will provide reasons in Section 4 why context –a relevant issue in any M&E project- should be modeled out of quality models, and considered rather as part of a *context component* integrated with a *non-functional requirements* component.

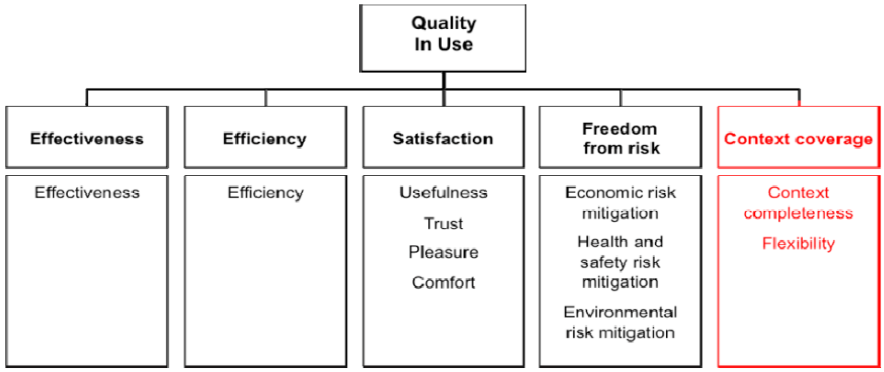


Figure 3: *Quality in Use* model in ISO 25010 standard (taken from [10]).

As can be seen from these figures, *learnability*, an internal/external sub-characteristic of *usability* in ISO 9126-1/25010 has not been moved to the *quality in use* model. However, *learnability* has been rephrased in 25010 as "degree to which a product or system can be used by specified users to achieve specified goals of learning to use the product or system with effectiveness, efficiency, freedom from risk and satisfaction in a specified context of use", which is a statement closer to the QinU semantic rather than in terms of product/system quality, showing again a lack of separation of concerns between the meaning of both views. Additionally, the previous definition "capability of the software product to enable the user to learn its application" was explicit for system quality borrowing the dialogue principles from ISO 9241-110 [14] regarding suitability for learnability.

For WebApps, users are expected to learn intuitively with no user manual. However, *learnability* in ISO 25010 is solely a product/system quality (not explicitly modeled as a QinU –system-in-use-requirement), which does not incorporate evaluation of the process of learning and does not model *learnability* in different real contexts of use such as the domain of the system-in-use, and its target users. Research made by [28] exemplifies the need to examine learning from different user viewpoints, as learning observed for new users is not necessarily related to continued learning. Furthermore, many researchers have determined *learnability* to be linked directly with actual *usability* as summarized by Abran *et al* [1]. Bevan [3] also noted *learnability in use* as part of *usability (in use)*. So in 2Q2U, both v1.0 and v2.0 versions, we included this sub-characteristic in the QinU model, and in sub-section 2.2 we strengthen the debate for including it. Ultimately, by considering *learnability* and *learnability in use* as two different yet related concepts we support a

clear separation of concerns in both quality views.

Furthermore, ISO 25010 does not include *information* (content) *quality* as a characteristic of either model. Recent ISO's intentions are for data to be addressed by a complementary standard, namely ISO 25012 [12]. ISO 25012 is a general data quality model intended to be used to establish *data quality* requirements, and plan and perform data quality evaluations. This standard is intended to be used in conjunction with ISO 25010, but by going to such length to define quality of data, it loses emphasis in using data as information and as a component of a WebApp rather than just data as an entity (category) itself. *Information quality* as a characteristic was researched by [23] whereby they proposed extending ISO 9126-1 with content quality containing four sub-characteristics including *content accuracy*, *content suitability*, *content accessibility*, and *legal compliance*. Rather than relying on separate standards, we adapt their contribution to augment the ISO 25010 standard to include *information quality* as a characteristic of product/system quality because this is a critical characteristic for current generation WebApps.

On the other hand, the term *user experience* is becoming more important as evidenced by the definitions by various researchers. Bevan [4] examined ISO 25010 from the viewpoint of *usability* (in use) and *user experience* (UX) and drew relationships regarding usability as performance in use, and satisfaction as it relates to user experience. Hassenzahl's work [8] in classifying UX in two categories, hedonic and pragmatic is also useful when examining usability (in use) from the *do*, pragmatic viewpoint and the *be*, or hedonic satisfaction viewpoint. To understand the term requires breaking down the word 'user experience' and examining first what experience means.

Experience is a general concept which refers both immediately-perceived events and the wisdom gained in interpretation of events. In the context of UX, it is a sequence of events over time for a user's interaction with the software product. Hassenzahl notes that the time dimension could be either momentary or accumulated and changing over time. The 'moment' perspective does not exclude the accumulation or summary perspective, rather, it adds to it like a continuum.

Examining the 'user' part of the *user experience* concept, Hassenzahl characterizes a user's goals into pragmatic, do goals and hedonic, be goals and assumes the interactive product quality is perceived in two dimensions, pragmatic and hedonic. Pragmatic quality refers to the product's perceived ability to support the achievement of tasks such as paying a bill and focuses on the product's utility and usability in completing tasks that are the 'do-goals' of the user. Hedonic quality refers to the product's perceived ability to support the user's achievement of 'be-goals', such as being happy, or satisfied with a focus on self. He also argues that the fulfillment of be-goals is the driver of experience and that lack of usability or inability to complete do-goals may prevent achieving be-goals, but do-goals are not the end goal of the user. Rather the real goal of the user is "to fulfill be-goals such as being autonomous, competent, related to others, stimulated, and popular through technology use." He also states that pragmatic quality enables achieving hedonic quality be-goals and has no value by itself, but only through enabling accomplishment of be-goals. In summary, *user experience* comes from fulfilling be-goals in the time dimension, at the moment, and in summation over time.

Given that, it's easy to see why UX has become such a buzz word regarding WebApps. Yet, a

common ISO 25010 standard definition for *user experience* is still not available. *Satisfaction*, as noted by Bevan, correlates to Hassenzahl's hedonic goals whereas usability (in use) and its do-goals are related to a user's pragmatic goals. In summary, *usability (in use)*, *satisfaction*, and *user experience*, need clearer relationships in order to model and evaluate them.

Starting from many of the abovementioned raised issues –missing in ISO 25010-, we found possible opportunities for improvement, which were addressed mostly in our 2Q2U v1.0 quality modeling framework, as we describe below.

2.2 First Version (v1.0) of 2Q2U

As indicated in the Introduction Section and in the discussion motivated in the previous sub-section, based on draft or issued standards and related literature that we reviewed by the end of 2009, it was difficult for us to comprehend many relationships of key calculable concepts (characteristics), and also we observed very often a lack of consensus in meaning as well. Specifically, reviewing ISO standards at that moment as the 25010 draft standard [11] (intended to replace to [13]), the 25012 [12], and other quoted works, it was still not totally clear where characteristics such as *information quality*, *learnability in use*, *actual usability* –in the sense of *usability in use-*, and *user experience* fit in regarding quality modeling, especially with regards to WebApps.

In that context, we developed in early 2010 the first version of the 2Q2U (which stands for *internal/external 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. In [17], the rationale for adding or adapting characteristics and calculable concepts was thoroughly discussed, also illustrating details of EQ and QinU models embraced in the 25010 draft standard [11]. However, for a better understanding of the updated 2Q2U (2.0) version described later in Section 3, we hereby include some details of the underlying rationale of 2Q2U v1.0.

Summarizing the first version of 2Q2U, first, we added two characteristics, namely: *information quality* (for the internal/EQ view), and *learnability in use* (for the QinU view). Second, we supplemented two new high-level calculable concepts for the QinU model, namely: *actual usability* and *user experience*, to which characteristics and sub-characteristics can be related and new models created in a flexible way. Aimed at fleshing out the *quality focus/model* column shown in Fig. 1 for product/system/system-in-use entity categories, Fig. 4 depicts the main model characteristics and relationships included in the 2Q2U (v1.0) quality modeling framework. Note that many sub-characteristics of characteristics are not represented in the figure for conciseness reasons. The added characteristics and concepts –which are not part of the quoted ISO standards- are defined below:

- *Information quality*, defined as the degree to which the software/WebApp provides accurate, suitable, accessible and legally compliant information.
- *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 *QinU* 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.
- *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 that in both versions of 2Q2U the relationships *influences* and *depends on* depicted in figures 4 and 5 have the same semantics as those stated in ISO 25010 standard.

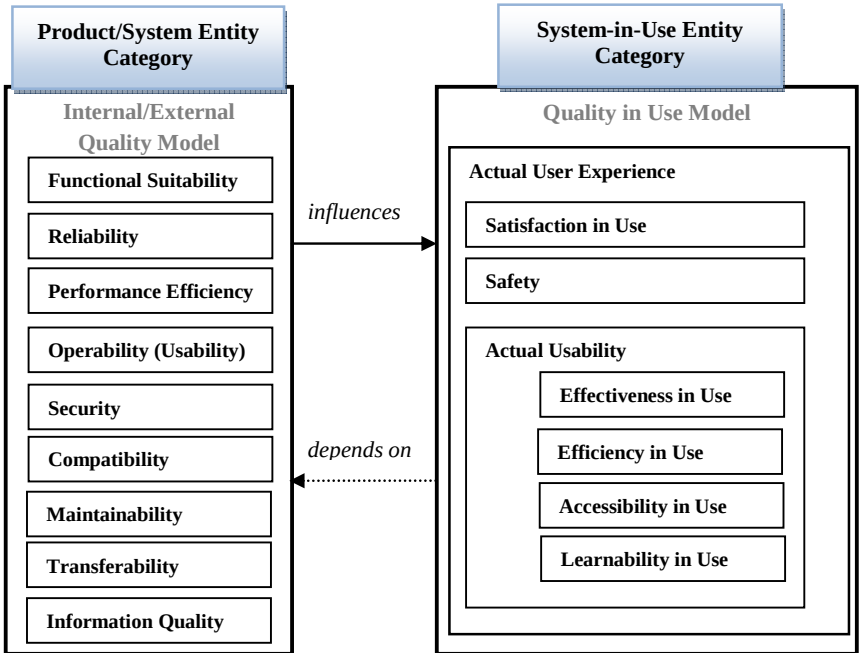


Figure 4: Quality model characteristics (with some sub-characteristics) and relationships between views in the 2Q2U (v1.0) quality modeling framework.

2.2.1 Information Quality in 2Q2U v1.0

The very nature of WebApps is a mixture of content, functions and services. Therefore we argued [19] that the eight ISO system quality characteristics (see Fig. 2) are not well suited, nor were they intended to specify requirements for *information quality*. As commented in sub-section 2.1, there is a need to integrate *information quality* as part of the overall quality of an application, particularly for WebApps, rather than as a separate entity category.

We intentionally use the term ‘information’ to differentiate from ‘data’. Data comes from attribute measurements, facts, formula calculations, etc. and are often organized and represented in databases.

Note that ISO 25012 is a general data quality model intended to be used to establish rather structured *data quality* requirements. On the other hand, information is the meaningful interpretation of data for a given purpose and context. Given that many WebApps are very often content oriented and intended mainly to deliver information –usually unstructured semantically-, the central issue is the ability to specify the *information quality* for WebApps from the internal and EQ viewpoints. This viewpoint is also supported by ISO 9241-110 [14], which relates characteristics of presented information to its dialogue principles. Therefore, we proposed augmenting the internal/external quality model with the *information quality* characteristic, with *content accuracy*, *content suitability*, *content accessibility*, and *content legal compliance* as sub-characteristics. Later in Section 3, Table 1, we present –for brevity reasons- the updated *information quality* definitions, sub-characteristics and some attributes just for the 2Q2U v2.0 EQ model.

It may be argued that *information quality* should be added as a *QinU* characteristic. However, when designing tasks for *QinU* for example for evaluating *efficiency* and *effectiveness in use*, content and functions are embedded in the task design itself rather than as system or product (WebApp) attributes. *Satisfaction* questionnaires can also address *information quality* with specific questions related to its sub-characteristics.

2.2.2 Learnability in Use

As mentioned earlier, *learnability* solely as a product quality characteristic does not incorporate evaluation of the learning process or different usage contexts. More specifically, we examine learning context to strengthen our reasoning to include *learnability in use* as an *actual usability* sub-characteristic, from the user group type and time viewpoints.

Regarding the former, the learning objectives and therefore behavior of different user groups have an impact on the learning process as novice users behave differently than expert users [28]. Ease of learning depends on the user group type and the task being attempted. As an extreme, a quality requirement characteristic to minimize the necessary learning time, or to make the learning time equal to zero depends entirely on who the user is, and what tasks they are trying to do. As another example of how user group types behave differently based on their background and task at hand, requirements for users who are trying to re-learn a task can be difficult to model as a product characteristic. Grossman *et al* [7] also noted several other user group types. Therefore, the dimension of user group types and its influence in *learnability* is of paramount importance.

Regarding the time aspect, learning from different user viewpoints such as initial learning and continued learning as researched by [27] are not necessarily correlated. So, measuring the learning of users must be done in the time dimension as the time delta between initial learning and continued learning has an influence on the *learnability* of the software in a real context. Many *learnability* measures focus on initial *learnability*; for instance, by simply picking some users who have not used the system before and measuring the time it takes them to reach a specified level of proficiency in using it. However, continued *learnability* requires assessing performance over time using a constant user group.

Some may argue that *efficiency* and *effectiveness in use* either combined or solely, can constitute

learnability in use. However, software that is easy to learn is not always efficient to use and vice versa. A WebApp's design evaluated highly as part of the *learnability* EQ characteristic, may lead to less efficient procedures, especially for an experienced user who may find a highly learnable function (for a beginner user) to be cumbersome.

Learnability therefore depends on the domain of the software, its target users and tasks at hand. Hence it cannot solely be determined by inspection, as an EQ characteristic. Bevan also related *learnability in use* as a sub-characteristic of *usability (in use)*, as indicated in sub-section 2.1. Given the aforementioned reasons, and recalling our definition as "the degree to which specified users can learn efficiently and effectively while achieving specified goals in a specified context of use", we argue that *learnability in use* should be added to the *actual usability* characteristic as a sub-characteristic.

Moreover, using the instantiated EQ and QinU models of 2Q2U v1.0 on the JIRA case study [16], we have determined –as a first evidence- how improved attributes of *learnability* and *information suitability* on the (JIRA) WebApp *influenced* positively on *learnability in use* attributes of the evaluated WebApp-in-use task.

2.2.3 Actual Usability and User Experience

As mentioned in sub-section 2.1, these two high-level calculable concepts are mainly derivatives through the [3, 4, 8] works. In Bevan´s work relating and explaining factors contributing to system *usability* and *UX* he defines 4 characteristics of *usability in use*: i) *Effectiveness and productivity in use* ii) *Learnability in use* iii) *Accessibility in use* and iv) *Safety in use*. He also matches *usability* to performance-in-use measures equivalent to those characteristics related to the pragmatic 'to-do' goals of the end user.

Measures of *UX* are noted by Bevan as being composed of *satisfaction in use* as equivalent to achieving pragmatic and hedonic goals, with its sub-characteristics as specified by ISO 25010 including *usefulness*, *trust*, *pleasure* and *comfort*. Hassenzahl further elaborates on hedonic goals as: "fulfilling the human needs for autonomy, competency, stimulation (self-oriented), relatedness, and popularity (others-oriented) through interacting with the product or service". He further states that pragmatic quality facilitates satisfaction of be-goals. That is, be-goals are not dependent on, but facilitated by do-goals; i.e. a user could be satisfied even if do-goals are not satisfied. For example, if a user cannot buy a product online efficiently (slowly with mistakes), but ends up buying what they like, then they may achieve their be-goals and be very satisfied.

Thus, achieving *UX* is influenced through satisfaction of both *usability (in use)* –pragmatic goals- and *satisfaction in use* –hedonic goals. Ultimately, the *actual usability* and *UX* definitions given in the beginning of sub-section 2.2 are based on this rationale.

2.3 Other Quality Models for Newer Generations WebApps

As introduced in Section 1, in order to evaluate quality for newer generation WebApps as social networks, composition-oriented applications such as mashup WebApps, and service- information-

oriented applications such as portals, among others, first requires comprehending in detail how they are different from older generations or from conventional software applications as user requirements, expectations, and behavior can be somewhat different. Note that in [20] a road map for Web X.0 generations and their features is made. However, very few proposals for modeling quality of newer generation WebApps are performed.

For example, mashups –as composition- are WebApps, which are created by combining and processing on-line third party resources that contribute with functionality, presentation/user interface (UI), and/or data. Capiello *et al* [5] define mashups as "Web applications that integrate inside one Web page *two or more* heterogeneous resources at different levels of the application stack, i.e., at the data, application logic, and UI level, *possibly putting them into communication among each other*". And they add that the primary reason for this refined definition "…is that we specifically want to focus on mashups that have their own UI (to distinguish them, for example, from so-called data mashups as the one created with Yahoo! Pipes) and that aim to provide added value by *integrating* a set of existing services or components, rather than coding something from scratch. That is, we want to emphasize the typical *component-based* nature of mashups and the resulting development paradigm of *composition*" (cf. pg. 139).

Starting from this mashup WebApp characterization they propose in [5] to model quality requirements classifying quality dimensions into three categories, namely, *data quality*, *presentation quality*, and *composition quality*. Authors pointed out that the *composition quality* category "is very peculiar for mashups since it focuses on the way components interact and measures the utility of the composition" (cf. pg. 146). The composition quality dimension is in turn decomposed into characteristics or attributes such as *added value*, *component suitability*, *component usage*, *consistency*, and mashup *availability*. Note that we will discuss thoroughly these features in the framework of our proposed *functional quality* EQ characteristic of 2Q2U v2.0, in Section 3.

In addition, for the *data quality* dimension authors propose characteristics or attributes such as *accuracy*, *timeliness*, *completeness*, *availability*, and *consistency*. In a previous work [6], many of the same authors specified a requirements model for *information* (data) *quality* targeted specifically for mashup WebApps, inspired also in our previous *information quality* proposal ([23] as per their citation) that are specific to modern Web 2.0 applications. Lastly, for the *presentation quality* dimension authors propose in [5] characteristics such as *usability*, and *accessibility*.

In summary, [5, 6] are important related works that we utilized while updating 2Q2U v1.0. However, we argue that more general EQ and QinU models can be specified while still providing the capability for tailoring and instantiating quality models for specific purposes and particular entity categories and entities. For instance, the *presentation quality* (sub-)characteristics, i.e. *usability*, and *accessibility* are already considered in both ISO 25010 and 2Q2U models, while the *information* (data) *quality* characteristic is also considered in both versions of the 2Q2U EQ model. Also many of the *composition quality* (sub-)characteristics can be part of the proposed *functional quality* EQ characteristic in 2Q2U v2.0, while mashup *availability* can fall within the *reliability* sub-characteristic also named *availability* (see Fig. 2). On the other hand, a weak point in [5] is a lack of explicit separation of concerns between EQ and QinU views.

In [9] authors propose a QinU model for Web portals, which is based on the quoted ISO 25010 draft standard QinU model. They selected from this standard *usability* (in use), *safety* and *flexibility* as the main QinU characteristics. While they adapted the included sub-characteristics for these three characteristics to the context of portal WebApps, other sub-characteristics were eliminated because they considered them not being sufficiently relevant for Web portal usage. In their proposed QinU model *usability* (in use) includes *effectiveness*, *efficiency* and *satisfaction*. While *satisfaction*, in turn, includes *ease of use*, *experience*, *perceived interaction quality*, and *sense of community*. As the weakness highlighted above, a general QinU model can be tailored and instantiated for specific entity categories, such as Web portals. Hence, the elimination of sub-characteristics for a given information need should not be made on the canonical quality model, but purposefully when tailoring and editing the included relevant characteristics, sub-characteristics and attributes. Besides, the semantic of user *experience*, *usability* and *satisfaction* given by authors differs to some extent from that given in 2Q2U and the last concepts differs also from the official ISO 25010 standard [10]. However, we reuse the given definition of *sense of community* [9], which we include in the *satisfaction* characteristic too, as we discuss in Section 3.

In summary, the QinU and EQ instantiated models for 2Q2U v1.0 were used for example in the JIRA case study, which was carried out in a real context of a testing company, where beginner testers were performing their daily task of editing customer-reported defects using JIRA. The research motivation of employing 2Q2U v1.0 instantiated quality models and their practical utility in the JIRA case study were discussed in [16]. In the present work we update the 2Q2U v1.0 quality models since the final ISO 25010 standard was issued in March, 2011 –i.e. after the published 2Q2U first version [17]- as well as given the emergence of other quality models or features for new generation WebApps as introduced above.

3 Updated Version (v2.0) of 2Q2U

Considering the officially issued ISO 25010 standard –discussed in sub-section 2.1- compared with the draft version [11], we have observed substantial changes both for (sub-)characteristic definitions and characteristics included in quality models, mainly for QinU. In fact, in upgrading 2Q2U, we tried to be as compliant as possible with the current ISO 25010 standard and considering also new evaluation (sub-)characteristics for new generation WebApps, while maintaining the same 2Q2U v1.0 quality modeling framework rationale. Below, first, we summarize the core inclusions/deletions giving some definitions, and then, we discuss further details mainly focusing on *functional quality*, because it is used to show in Section 5 its practical usage for mashup WebApps.

Summarizing the 2Q2U v2.0 (see Fig. 5), firstly, 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. 3) because in our approach this is represented as a *context* component linked to the non-functional requirements specification component, independently of quality models, as we will highlight in sub-section 4.2.1. Secondly, we rephrased the *functional suitability* characteristic given in the ISO product quality model (see Fig. 2) as *functional quality* and rearranged its sub-characteristics.

Thirdly, we have eliminated two sub-characteristics of the 2Q2U v1.0 *information quality* characteristic and rephrased its definition. And lastly, we also rephrased the ISO *usability* definition –and some of its sub-characteristics as *learnability*- in order to increase clarity and reduce ambiguity since basically they are defined in terms of EQ and QinU at the same time in ISO 25010, as indicated in sub-section 2.1. Note that sub-characteristics are not shown in Fig. 5 for the EQ model, so except those discussed here the others remain the same both in names and definitions as in the ISO EQ model (Fig. 2). For the QinU model in Fig. 5 only the *freedom for risk* sub-characteristics are not shown, but are the same as those included in Fig. 3. Some rephrasing for ISO *effectiveness* and *efficiency* characteristics are documented as well.

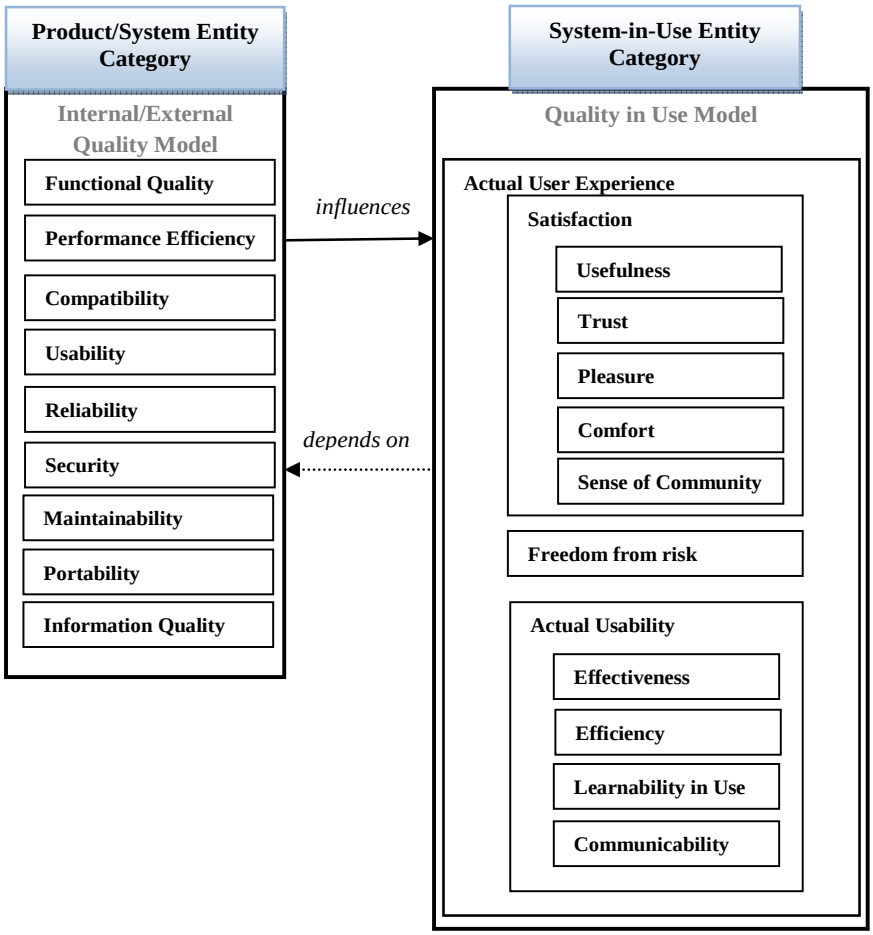


Figure 5: Quality model characteristics (with some sub-characteristics) and relationships between views in 2Q2U (v2.0).

3.1 Information Quality in 2Q2U v2.0

Information quality is defined in 2Q2U v2.0 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".

Table 1: Definition of *Information (Content) Quality* characteristics, sub-characteristics, and potential attributes (*in italic*). Note that code numbers are only intended to show hierarchical relationship dependences.

Calculable Concept/ *Attribute*	**Definition**
1 Information 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 Information 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 Added value	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.

This EQ characteristic was also included in 2Q2U v1.0 (see definition in sub-section 2.2). In 2Q2U v1.0 there were four sub-characteristics, but now there remain only two sub-characteristics viz. *information accuracy* and *information suitability* as shown in Table 1. Hence, the former *content accessibility* sub-characteristic was moved in 2Q2U v2.0 to *usability* (likewise it is in [10]); 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" (cf. [10], p.21). Note that in Table 1 there are definitions for extra sub-characteristics and potential measurable attributes. Some *information suitability* attributes were used in the recent JIRA case study [16], and specific discussions of included *information quality* sub-characteristics are in [23] as well.

3.2 Functional Quality in 2Q2U v2.0

Functional quality is defined 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 updated EQ model includes the three sub-characteristics stated in the ISO 25010 functional suitability characteristic, i.e., correctness, appropriateness and completeness (see Fig. 2). But they are re-arranged in our representation while adding increased granularity.

Table 2: Definition of *Functional Quality* characteristic, sub-characteristics, and potential attributes (*in italic*). Note that code numbers are only intended to show hierarchical relationship dependences.

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 a product or system is integrated and beneficial for the task at hand to a given user.
2.1.1 Integratedness	Degree to which a 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 (synonym: *Usefulness*)	Degree to which a 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.

As shown in Table 2, now *functional quality* has only two sub-characteristics viz. *functional accuracy* and *functional suitability* (note the parallelism of terms and categories with *information*

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* and particularly *integratedness* (defined as "degree to which a 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. *Integratedness* was absent as sub-characteristic or attribute in [5], but implicit in the authors' definitions or comments; while *added value* was explicitly considered in their model.

We consider important for the rationale of including *integratedness* to quote the following excerpt, underlying in the text the keywords, which are part of our attributes/definitions: "We can quantify the added value along a scale that ranges from the case in which a mashup gives simply the opportunity to render data coming from different sources (without any attempt to integrate them) to the case in which additional features and data are provided by an adequate integration. For example, a mashup as dailymashup.com provides low added value since its single page offers a very poorly integrated view on some selected news (taken from Yahoo!news) and, in a totally unaligned fashion, also on the top-ranked last 24 hours photos from Flickr. More added value would be offered if the mashup components were synchronized. To provide added value, mashups must offer to the users additional features or data, as for example the mashup published on www.bluehomefinder.com, where an advanced service for finding houses offers the localization of houses on a map plus other features that allow the users to perform operations of filtering and selection" (cf. [5], pg. 149).

When components are integrated, the interaction among components adds extra value because the combination (the whole) could provide more added value to users than each individual component (the part) functioning separately. The interaction comes from in part, from the synchronization among different components. Therefore, integratedness is a systems' property of being made up of parts that behave harmoniously as a whole.

Recalling the rational given in [23] for including *information accuracy*, which addresses features/attributes that deal with the very intrinsic nature of the information quality; and, *information suitability*, which addresses features/attributes that deal with the contextual nature of the information quality, i.e. it emphasizes the importance of conveying the appropriate information for user-oriented goals and tasks, a parallelism with *functional accuracy* and *suitability* can be envisioned. We defined *functional suitability* (in Table 2) as "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".

As commented above, we identified in turn for *added value* two potential attributes namely *integratedness* and *beneficialness*. *Beneficialness* (or *usefulness* as synonym) is defined as "degree to which a product or system is advantageous and contributes to make better decisions for an intended user goal". The following excerpt can be 'beneficial' for understanding its inclusion in the EQ model:

"Component usage: it may happen that even though a component is very rich from the point of

view of data and functionality, it is improperly used within a composition. For example, the Google Maps API offers several operations, such as geocoding, directions, street view, and elevation. Let us consider that a mashup developer decides to just use the street view feature. This choice is not reasonable if the user goal is to get oriented within a geographical area: the street view just offers a local and realistic view of a selected point of interest, while it does not provide a larger view of the area in which the point is located" (cf. [5], pg. 149). Note that the very rich term regarding authors in our model can be represented by the *completeness* attribute.

Appropriateness and *completeness*, in our model, are two sub-characteristics/attributes related to suitable *coverage*. Again, we use the following excerpt to support our reasoning for including *appropriateness*: "Component suitability: it refers to the appropriateness of the component features and data with respect to the goal that the mashup is supposed to support. For example, a mashup that aims at providing addresses of the nearby restaurants with respect to the current user location should be effectively built based on map-based components. In fact, a simple list of textual addresses could not appropriately support those users that are not acquainted with the geographical area" (cf. [5], pg. 149).

Regarding *functional accuracy* in Table 2, we specified two sub-characteristics viz. *correctness* (with two potential attributes: *precision* and *consistency*) and, *credibility*. Also in [5] *consistency* was considered: "… poor quality compositions can also be caused by inconsistencies at the orchestration level. In fact, the composition of two components is feasible if the two linked operations are compatible from only a syntactic perspective even if they are incompatible from a semantic perspective. In this way, the composition can produce inaccurate results" (cf. pgs. 149-150).

We argue the updated 2Q2U v2.0 *functional quality* characteristic has its own 'added value' for instantiating and evaluating *functional quality* requirements not only for older but also for newer generation WebApps. Particularly, we have just discussed the mashup WebApp entity category and the quotations of the rich discussions made in [5], to motivate the reader for a better comprehension of the mashup evaluation example that we elaborate in Section 5, from the *functional quality* point of view.

3.3 Usability, Communicability and Sense of Community

On the other hand, we have a clear separation of concerns in 2Q2U regarding the specification of *usability* from the EQ model view, and *actual usability* (or *usability in use*) from the QinU model view, as depicted in Fig. 5.

Usability is rephrased in the 2Q2U v2.0 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". While in the current ISO 25010 product quality model it 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". This statement is closer to the QinU semantic. In 2Q2U v2.0 *actual usability* is rephrased –compared to the 2Q2U v1.0 definition given in sub-section 2.2- as the "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". Table 3 shows renamed sub-characteristics or revised definitions for 2Q2U v2.0 *usability* compared with ISO 25010.

Finally, regarding the added QinU (sub-)characteristics in 2Q2U v2.0, the *actual usability* definition includes the new *communicability* term as well as the inherited *learnability in use* sub-characteristic. See these definitions, or rephrased ISO sub-characteristics, in Table 4.

Table 3: Names and definitions of *Usability* sub-characteristics as per ISO 25010 and 2Q2U v2.0 product/system model.

ISO *Usability* sub-characteristic / Definition	**2Q2U v2.0 *Usability* sub-characteristic / Definition**
Appropriateness recognizability / Degree to which users can recognize whether a product or system is appropriate for their needs.	*Appropriateness recognizability* (synonym: *understandability*) / Note: Same definition.
Learnability / Degree to which a product or system can be used by specified users to achieve specified goals of learning to use the product or system with effectiveness, efficiency, freedom from risk and satisfaction in a specified context of use.	*Learnability* / Degree to which the product or system enables users to learn its application.
Operability / Degree to which a product or system has attributes that make it easy to operate and control.	*Operability* / Note: Same definition.
User error protection / Degree to which a system protects users against making errors.	*User error protection* / Degree to which a product or system protects users against making errors and provides support to error tolerance.
User interface aesthetics / Degree to which a user interface enables pleasing and satisfying interaction for the user.	*Attractiveness* (synonym: *UI aesthetics*)/ Degree to which the product or system is attractive to the user.
Accessibility / Degree to which a product or system can be used by people with the widest range of characteristics and capabilities to achieve a specified goal in a specified context of use.	*Accessibility* / Degree to which a product or system can be used by people with the widest range of characteristics and capabilities.

Communicability [26] as part of *actual usability* evaluates pragmatic or do-goals [16] as well as *effectiveness*, *efficiency* and *learnability in use*. It can help to find the causes of problems looking at communicative breakdowns in the user/system-in-use interactions. Communicative breakdowns between users and system-in use interactions for given tasks can be in turn tagged [26], which facilitates to collect observationally (usually by intrusive means) interaction problems associated to breakdown categories. In the JIRA case study [16], we collected non-intrusively QinU problems related to *effectiveness*, *efficiency* and *learnability in use* attributes for a given task, to derive EQ weak attributes for the (JIRA) WebApp with the aim to perform further improvements. In the QinU-EQ derivation process *communicability* measures in conjunction with *effectiveness*, *efficiency* and *learnability in use* measures can help performing better mappings between QinU problems and weak system attributes. For brevity reasons, this will be discussed in a separate manuscript.

Lastly, we also added *sense of community* as part of the *satisfaction* characteristic, which evaluates hedonic or be-goals, as discussed in sub-section 2.2.3. We re-use totally the ISO *satisfaction* sub-characteristics, which includes *usefulness*, *trust*, *pleasure* and *comfort*. The meaning of these sub-characteristics does not embrace the semantic of *sense of community* i.e. how satisfied a

user is when meeting, collaborating and communicating with other users with similar interest and needs. The *sense of community* concept is also implicit in the Hassenzahl statement, when he elaborates on hedonic goals as: "fulfilling the human needs for autonomy, competency, stimulation (self-oriented), relatedness, and popularity (others-oriented) through interacting with the product or service"; or when he says that rather the real goal of a user is "to fulfill be-goals such as being autonomous, competent, related to others, stimulated, and popular through technology use".

Hence, the *sense of community* sub-characteristic can be especially useful for evaluating aspects of the *satisfaction* –and ultimately *user experience*- characteristic for social network and collaborative WebApps. We made a first study using *communicability* and *sense of community* sub-characteristics for evaluating the QinU of two social WebApps [15]; specifically, Douban FM (http://douban.fm/) and Xiami Radio (http://www.xiami.com/radio/) are two leading online automated music recommendation services in China.

Table 4: Just names and definitions in 2Q2U v2.0 QinU (sub-)characteristics that are absent [10], or were rephrased.

2Q2U v2.0 QinU (sub-)characteristic / Definition	**Related ISO QinU concept / Definition**
Actual User Experience / Degree to which specified users can achieve actual usability, freedom from risk, and satisfaction in a specified context of use	Note: Absent calculable concept
*Actual Usability (*synonym: *usability in use)* / 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	Note: Absent calculable concept, but similar concept (i.e. *usability in use*) was in the ISO 25010 draft [11].
Effectiveness / Degree to which specified users can achieve specified goals with accuracy and completeness in a specified context of use.	*Effectiveness* / Accuracy and completeness with which users achieve specified goals.
Efficiency / Degree to which specified users expend appropriate amounts of resources in relation to the effectiveness achieved in a specified context of use.	*Efficiency* / Resources expended in relation to the accuracy and completeness with which users achieve goals.
Learnability in use / Degree to which specified users can learn efficiently and effectively while achieving specified goals in a specified context of use.	Note: Absent calculable concept
Communicability / Degree to which specified users can achieve specified goals without communicative breakdowns in the interaction in a specified context of use.	Note: Absent calculable concept
Sense of Community / Degree to which a user is satisfied when meeting, collaborating and communicating with other users with similar interest and needs	Note: Absent calculable concept

4 A Generic M&E Approach: An Overview

In the previous two sections we have concentrated on quality views, quality focuses and models (e.g. EQ, and QinU). Particularly, for quality models we have discussed and specified the included characteristics and sub-characteristics for 2Q2U v2.0, giving for illustration purpose more details –at the attribute level- for *information* and *functional quality* characteristics. However, the quality

models above discussed were neither instantiated for a given purpose and information need, nor evaluated for a concrete entity. Moreover, quality models can be related to each other.

Fig. 6 shows that the *influences* relationship exists between EQ and QinU views, and *depends on* between QinU and EQ. Therefore, quality focuses/views –and their relationships-, and also their associated entity categories can be grouped in a mechanism that we call a *quality modeling framework*. So, not only quality models can be tailored and instantiated but also relationships between quality views. In order to do this, and to drive the specific measurement and evaluation process accordingly, a specific *M&E strategy* should be used.

This section gives an overview of the proposed generic *evaluation approach*, which is made up of a basic *quality modeling framework* and *M&E strategies*, where a concrete strategy should be selected for purposefully instantiating models and performing evaluations in a given M&E project. Further details of this evaluation approach using the SIQinU (*Strategy for understanding and Improving Quality in Use*) strategy for the JIRA case study, are in [16]. In sub-section 4.1 we summarize a basic *quality modeling framework* and in sub-section 4.2 we overview the three pillars of any M&E strategy, stressing this on GOCAME (*Goal-Oriented Context-Aware Measurement and Evaluation*) strategy, which is used later in Section 5.

4.1 A View of the Quality Modeling Framework

One of the first steps in a concrete M&E project is to define non-functional requirements, basically, using as input a quality model. A quality model, which is targeted for a quality focus and entity category, provides the basis for 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, in any given M&E 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) can be necessary as well. A quality modeling framework can be used to develop these relationships.

Fig. 6, as an extension of Fig 1, shows the schema for a basic quality modeling framework, which is slightly adapted from [25]. Note that target entity categories of quality models and their relationships are also considered in the recent ISO 25010 standard (cf. [10], pg. 28, Fig. C.3). Likewise in Fig. 1, the first column in Fig. 6 represents the super-category of the above mentioned entity categories; the second column, specifies the quality focus respectively, i.e., the root characteristic of a quality model; and the third column illustrates examples of entity categories. In addition, the quoted *influences*/*depends on* relationships between quality models are depicted.

Usually, 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 addition, 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 a specific entity category; for instance, GOCAME is the name of a specific M&E strategy.

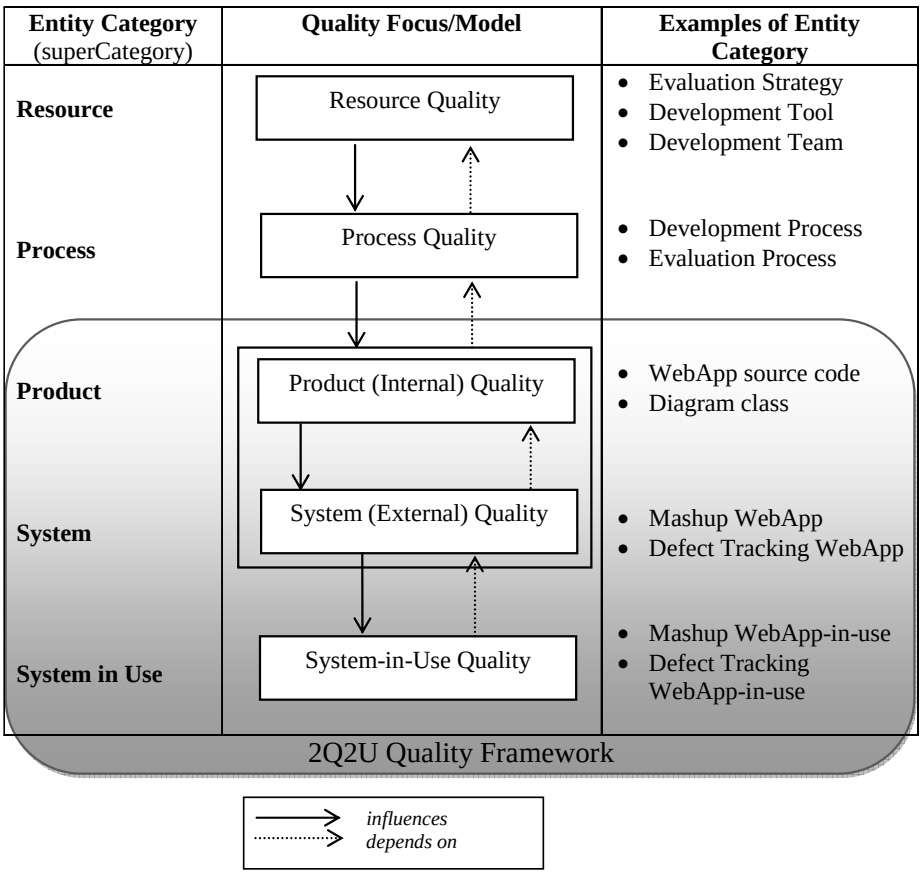


Figure 6: Basic quality modeling framework regarding target entity (super) categories and quality focuses/models. In the bottom the 2Q2U quality modeling framework is highlighted.

Besides, more specific entity categories for system such as "Mashup WebApp" or "Defect Tracking WebApp" can be identified. Particularly, the woozor (woozor.com/) mashup WebApp is a concrete entity for the former entity category, and JIRA (www.atlassian.com) for the latter entity category. In Section 5, we will illustrate the instantiated EQ model for the woozor mashup WebApp. Besides in [16], a study for both entity categories viz. "Defect Tracking WebApp" and "Defect Tracking WebApp-in-use", exploring also the relationships between EQ and QinU models, was conducted.

Finally, in Fig. 6, we depict a shadowed rectangle labeled *2Q2U quality framework*, which is a sub-set of the shown quality modeling framework. We can conclude that a quality modeling framework, conveys more information than a simple quality model, and it can be reused and tailored

purposefully by M&E strategies in an integrated yet flexible manner.

4.2 Measurement and Evaluation Strategies: An Overview

As above mentioned the proposed *evaluation approach* relies on two pillars, namely: i) a *quality modeling framework* –where 2Q2U is a subset; and ii) *measurement an evaluation strategies*, which in turn are grounded on three principles viz. a *M&E conceptual framework*, a *well-established M&E process*, and *methods and tools*.

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 [24], 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 it 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". 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.

Briefly outlined, the GOCAME strategy follows a goal-oriented approach in which all the activities are guided by a stated and specific information need; this information need is intended to satisfy particular non-functional requirements of some entity category (and in the end a concrete entity) for a particular purpose and stakeholder's viewpoint. The non-functional requirements are represented by concept models including high-level calculable concepts (e.g. EQ, or functional quality) as in 2Q2U's quality models, which in turn combine measurable attributes of the entity under analysis. 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, which can involve improvement or other change objectives.

GOCAME –likewise SIQinU- 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 M&E project/program requires a sound *M&E 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 [19, 24] based on our metrics and indicators ontology. Note that GOCAME re-uses totally C-INCAMI conceptual framework and its 6 components (outlined in sub-section 4.2.1).

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. We outline a few activities, mainly those devoted to illustrate M&E design and quality model instantiation in sub-section 4.2.2.

Finally, GOCAME's third principle is *methods and tools*, which can be instantiated from both the conceptual framework and process. Used methods and tools were illustrated elsewhere [16, 24].

4.2.1 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: *M&E project definition*; *Nonfunctional requirements specification*; *Context specification*; *Measurement design and implementation*; *Evaluation design and implementation*; and *Analysis and recommendation specification*.

Of particular interest to illustrate in this article for instantiating quality models are C-INCAMI's Nonfunctional Requirements Specification (NFRS), and Context Specification (CS) components, both shown in Fig. 7. For conciseness reasons we describe only these two, while the others can be found in [24]. Note that key words in the figure are highlighted in italic below.

In a broad sense, the NFRS component (labeled *requirements* in Fig. 7) specifies the *Information Need* of any M&E project; that is, the *purpose* (e.g. "understand", "predict", "improve" etc.) and the *userViewpoint* (e.g. "developer", "traveler user", etc). In turn, it *focuses* on a *CalculableConcept* (whose *name* is for instance "EQ") and *specifies* the *EntityCategory* to evaluate –e.g. a system, system-in-use *superCategory*-, by means of a concrete *Entity* –e.g., the "woozor.com WebApp".

On the other hand, the selected calculable concept and its *subconcepts* can be *represented by* a *Concept Model* (e.g. in "EQ 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 "mashup WebApp").

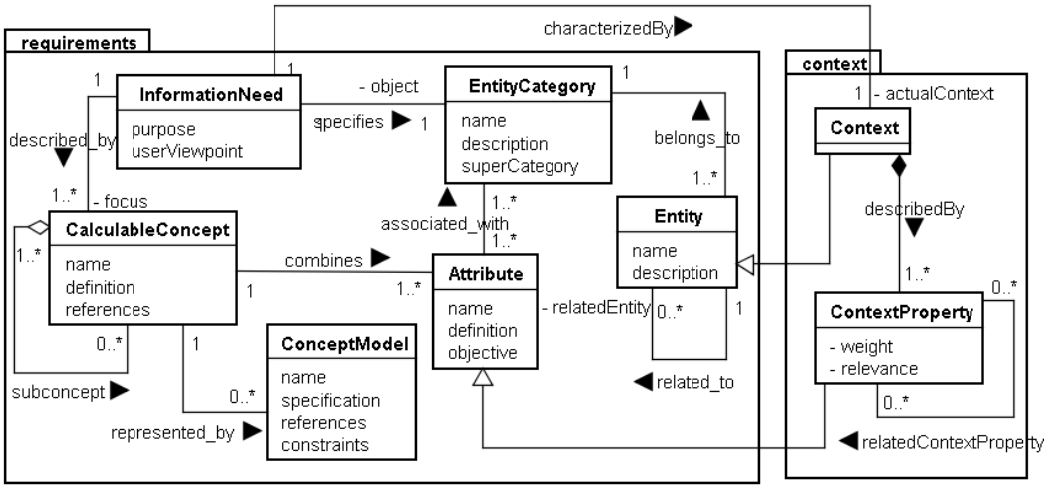


Figure 7: C-INCAMI *Nonfunctional Requirements* and *Context Specification* components.

The context specification component (labeled *context* in Fig. 7) 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, user tasks, etc.-, the project itself, the system domain characterization, as the pattern type of the mashup composition, e.g. master-master, master-slave [6], etc.

To describe the context, properties of the relevant entities, which are also attributes called *ContextProperties* are used. Context is particularly important regarding QinU requirements as instantiation of requirements must be done consistently in the same context so that evaluations and improvements be accurately measured and compared. But also context in a given M&E project can be important regarding EQ requirements, as we see in sub-section 5.1.2. Recall that in 2Q2U v2.0 we propose to delete the ISO 25010 *context coverage* characteristic in Fig. 3, since as shown here the context specification can be represented independently 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.

4.2.2 The GOCAME Process

In order to instantiate models for M&E in a consistent, repeatable and purpose-oriented way, in our evaluation approach we utilize the GOCAME's process and C-INCAMI M&E framework capabilities. Considering process views, the functional process perspective for GOCAME represents

what activities and tasks should be specified, what hierarchical activities structure there exists, what conditions (pre- and post-conditions) should be accomplished, and what inputs and outputs (artifacts) will be required.

Taking into account the terminology and components used in the C-INCAMI framework, the GOCAME process embraces the following core activities (see Fig. 8): A1) *Define Non-Functional Requirements*; A2) *Design the Measurement*; A3) *Design the Evaluation*; A4) *Implement the Measurement*; A5) *Implement the Evaluation*; and A6) *Analyze and Recommend* as shown.

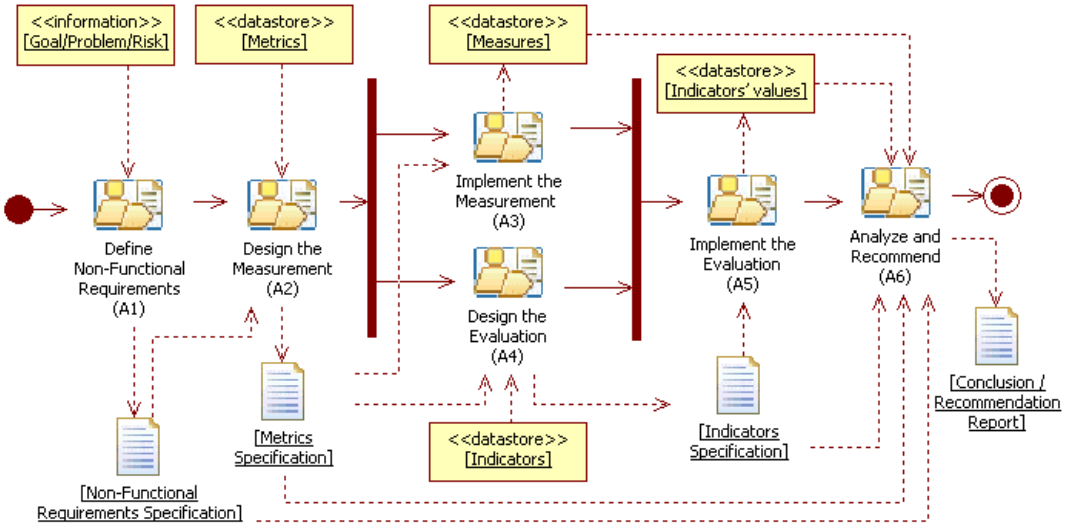


Figure 8: Overview of the *GOCAME Measurement and Evaluation* Process.

Of particular use in instantiating quality models for different purposes are C-INCAMI's NFRS and CS components (recall Fig. 7), and the following three related generic processes represented in Fig. 9, which are sub-activities of A1, namely: i) *Establish Information Need*; ii) *Specify Context*; and iii) *Select a Concept Model*. We call them generic activities for the non-functional requirements specification process, because these GOCAME activities are accordingly specialized in the SIQinU strategy for both QinU and EQ [22]. Fig. 9 shows also that a given quality model can be selected from the *Concept Models* <<datastore>> which is linked to a *quality modeling framework* – introduced in the sub-section 4.1.

Besides, in Table 5 an activity template in which the objective and definition, input and output artifacts, roles, sub-activities with depicted interdependencies, conditions are documented for the *Select a Concept Model* activity of Fig. 9.

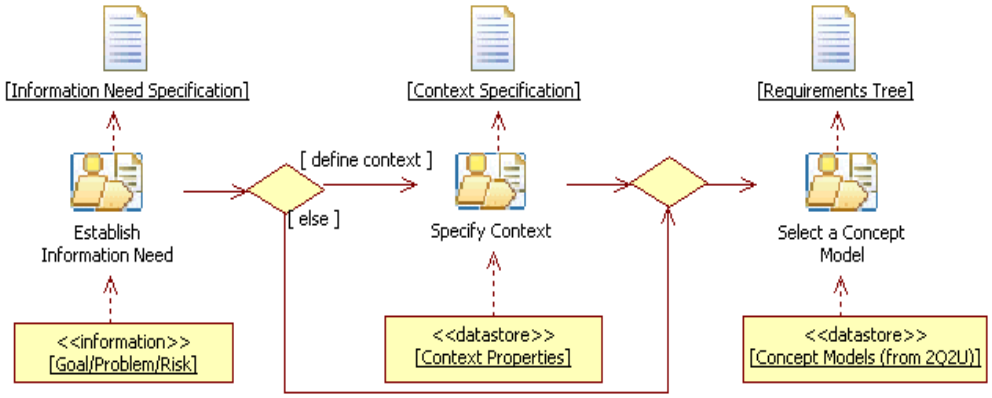
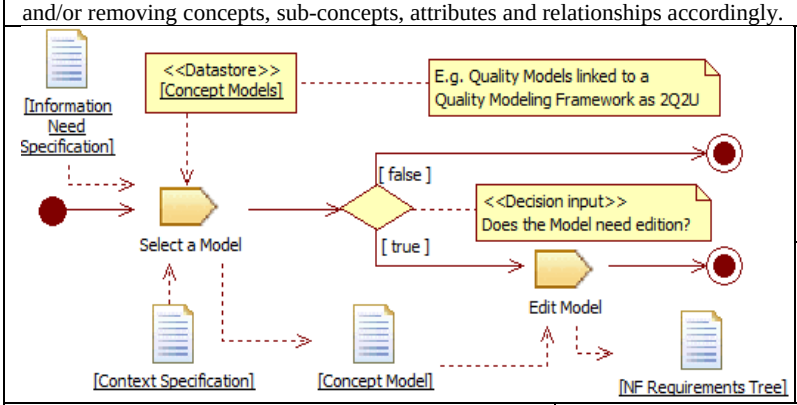


Figure 9: Sub-activities for the *Define Non-Functional Requirements* (A1) activity.

Table 5: Process template in which information is 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 colspan="2">Input Artifacts:
• Concept Models repository
• Information Need Specification
• Context Specification</td><td>Output Artifacts:
• NF Requirements Tree (i.e. the selected/edited characteristics, sub-characteristics and attributes)</td></tr>
<tr><td colspan="2">Pre-conditions: there is a repository with concept models.</td><td>Post-conditions: the instantiated requirements tree.</td></tr>
</table>

Examining the activity diagram within Table 5, we can say that the *Information Need* and *Context* specification documents are inputs to the *Select a Model* sub-activity. Also, there is another input to select a model from the *Concept Models* <<datastore>>. This is to say, knowing beforehand the

evaluation *focus*, *purpose*, *user viewpoint*, and *entity category*, for example, a "EQ model" can be chosen from the 2Q2U quality modeling framework (e.g. from 2Q2U v2.0 or from 2Q2U v1.0). In addition, the *Edit Model* sub-activity may be necessary to remove or add *(sub-)concepts*, and *attributes* according to the information need and context.

Lastly, note the correspondence and consistency of terms between the C-INCAMI conceptual framework and the GOCAME process. For instance, labels of activities and artifacts in figures 8 and 9 come from the labels of terms, attributes and relationships of the non-functional requirements and context specification components shown in Fig. 7.

5 Instantiated External Quality Model: A Mashup WebApp Example

This section presents an instantiation of the 2Q2U EQ model for a mashup WebApp, specifically for the *functional quality* characteristic shown in Fig. 5 –and detailed in Table 2. Additionally, an evaluation of a mashup WebApp –not as an individual component but rather as a composition- is performed. For this purpose, taking into account the introduced *GOCAME strategy* and *quality modeling framework*, we emphasize the definition of non-functional requirements in sub-section 5.1 going from establishing the information need to EQ model selection and editing activities. Then, in sub-section 5.2 design and implementation issues for measurement and evaluation are described as well. Finally, analysis of results is made in sub-section 5.3. Note that to show practical usage of the 2Q2U EQ model with sub-characteristics and measurable attributes to evaluate *functional quality* for a mashup WebApp, other relevant characteristics as *usability* and *information quality* were omitted for conciseness reasons.

5.1 Defining Non-functional Requirements

As depicted in Fig. 8, the *Define Non-Functional Requirements* activity (coded A1) has a specific goal or problem as input and a nonfunctional specification document as output. It consists of: *Establish Information Need*, *Specify Context*, and *Select a Concept Model* sub-activities as shown in Fig. 9, and described below from sub-section 5.1.1 to 5.1.3 respectively.

5.1.1 Establishing Information Need

The *Establish Information Need* activity is aimed at indicating the purpose, the user viewpoint, the quality focus, entity and entity category for the created M&E project. Particularly, this activity –reusing the terms of the C-INCAMI requirements component (Fig. 7)-, involves *Define purpose* and *user viewpoint*, *Establish object* and *entity* under study, and *Identify focus* of the evaluation as represented in Fig. 10. In the mashup study, the purpose for performing the evaluation is to "understand" the WebApp being used from the "traveler user" viewpoint with focus on the "functional quality" calculable concept. In lay terms, from a traveler user point of view, we want to understand the functional quality of this WebApp.

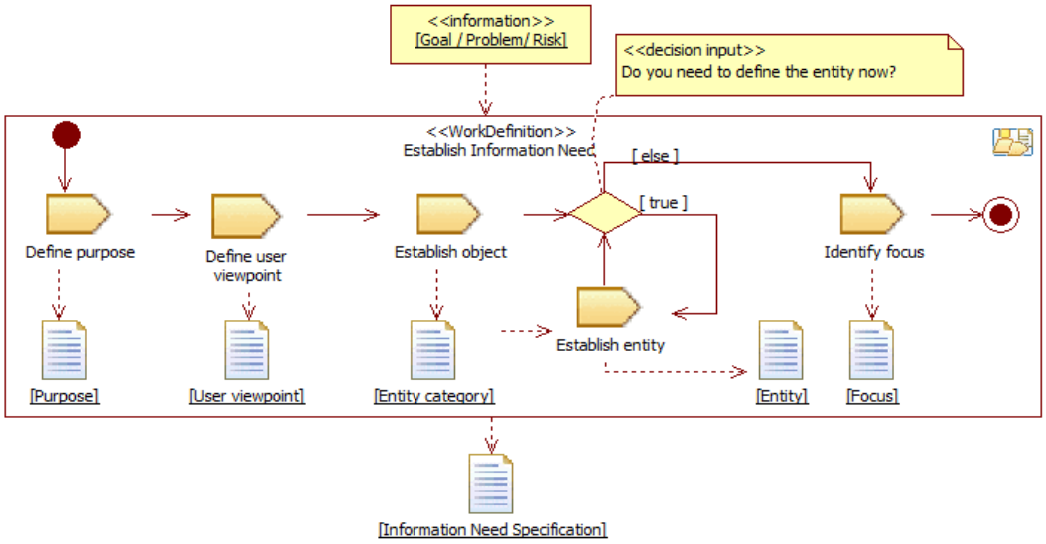


Figure 10: Activities to *Establish Information Need*.

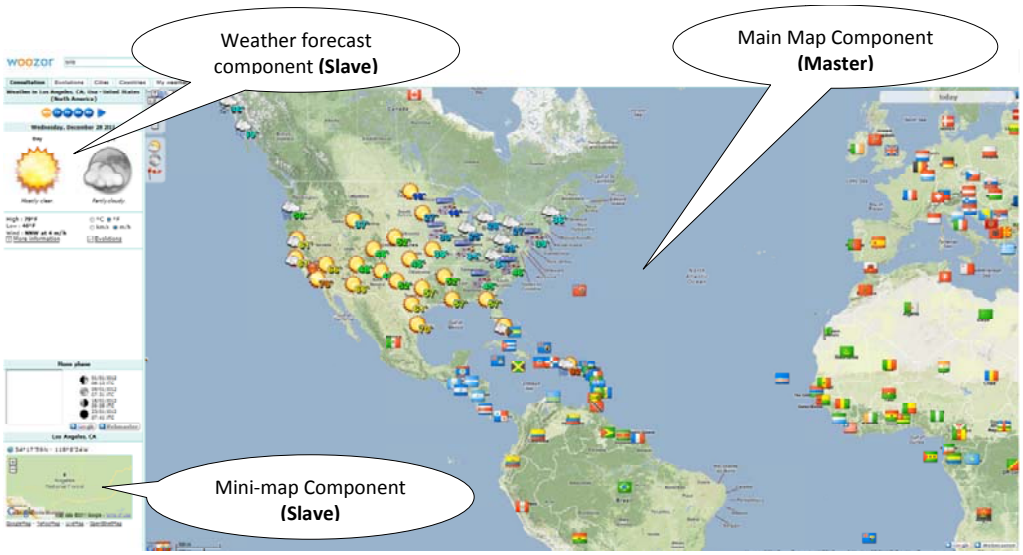


Figure 11: Screenshot of the woozor.com WebApp highlighting the three component views.

The entity category (i.e. the object) assessed is a “mashup WebApp”, while the concrete entity being studied is “woozor.com”. Recall that in the quality modeling framework of Fig. 6, a mashup WebApp is a “system”, from the assessed entity super-category point of view. Woozor is a WebApp

basically made up of three mashed up components intended to provide information about current and forecasted weather for a user, covering cities and their locations all around the world.

Fig. 11 shows a screenshot of the woozor WebApp highlighting the three rendered components.

As output for the *Establish Information Need* activity, an information need specification document is yielded, which is an input to the *Specify Context* activity (Fig. 9).

5.1.2 Specifying Context

This activity deals with the selection of *context properties* such as "pattern type of mashup composition", "number of visible components", among others, taking into account that *context* is a special kind of *entity* (recall Fig. 7) related to the mashup WebApp entity to be evaluated. In order to understand the rationale for context properties and the importance to store them in a M&E project separately from the instantiated quality model and its measurable attributes, a brief characterization of the mashup domain should be made. A mashup component can be hidden or visible. The latter plays different roles that can affect the user's quality perception of the final integration. In [6] authors identified three typical mashup roles, namely: i) *master*, in which even if a mashup WebApp integrates many components in a single page, in most cases, one component is more important than the others. This master component is the one users interact with most, being usually the starting point for user interaction that causes the other components to synchronize and react accordingly; ii) *slave*, in which the behavior of a slave component depends on another component. In addition, many mash-up WebApps allow users to interact with slave components, but the content or functionality that slave components display are selected via the user's interaction with the master component and automatically synchronized; iii) *filter*, which let users specifying conditions over the content the other components can show.

Based on these roles, the cited authors further identify three basic patterns that characterize most mashup WebApps. One is the *master-slave* functional pattern, which is the most widely used among today's mashup WebApps; in the authors' words: "it features all three component roles. A filter component lets users restrict the data all the other components simultaneously show. Users employ the master component to perform the main interactions with the application, such as selecting interesting data items. The slave component is automatically synchronized according to the selections performed on the master component, thereby visualizing the selected elements' details" (cf. [6], pg. 18). The other mashup patterns are *master-master* and *slave-slave*.

Woozor WebApp can be evaluated basically considering the *master-slave* functional (behavioral) pattern. Fig. 11 shows its three views highlighting the master and slave components. For consistency and repeatability reasons in ulterior evaluation and analysis, we record the following six context attributes (and their values):

1) *mashup composition pattern type* ={"master-slave"}; 2) *mashup composition source type*={"functional-oriented"} –note that other values can be "data-oriented", "UI-oriented"-; 3) *number of visible components*= {3}; 4) *number of hidden components*= {0}; 5) *number of slave components*= {2}; and 6) *number of master components*= {1}.

These context properties are not part of the instantiated EQ model for evaluating the woozor entity, but rather characterize the situation of the system at hand. For the above reasons we argue *context* can possibly be represented out of the scope of quality (concept) models.

5.1.3 Selecting a Concept Model

Fig. 9 shows that *Select a Concept Model* is the last sub-activity to be enacted in order to *Define Non-Functional Requirements*. Additionally, the activity template in Table 5 fully documents the *Select a Concept Model* sub-activity.

Examining the activity diagram within this table, we can say that the *Information Need* and *Context Specification* artifacts 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 quality focus, purpose, user viewpoint, entity category, etc. an "EQ model" can be chosen, for example, from the 2Q2U v2.0 *quality modeling framework*. In addition, editing the model may be necessary to remove (sub-)characteristics, and add attributes accordingly.

Table 6*: EQ Requirements Tree Specification* for a mashup WebApp. In the 1st column represents (sub-)characteristics and attributes -in italic; the 2nd column represents elementary, partial and global indicators' values (in %).

1. **Functional Quality**	**32.26**
1.1. **Functional Accuracy**	**22.02**
1.1.1. **Correctness**	**3.30**
1.1.1.1. *Consistency of forecasted temperature against its source*	3.3
1.1.2. **Credibility**	**40.75**
1.1.2.1. **Reputability**	**47.50**
1.1.2.1.1. *Composition Reputability*	25
1.1.2.1.2. *Component's Source Reputability*	70
1.1.2.2. *Verifiability*	25
1.2. **Functional Suitability**	**42.50**
1.2.1. **Added value**	**68.75**
1.2.1.1. **Integratedness**	**100**
1.2.1.1.1. *Component Integratedness on clicking*	100
1.2.1.1.2. *Component Integratedness on searching*	100
1.2.1.2. **Beneficialness**	**37.50**
1.2.1.2.1. *Current Weather Capability Beneficialness*	0
1.2.1.2.2. *Weather Forecast Capability Beneficialness*	75
1.2.2. **Coverage**	**16.25**
1.2.2.1. **Appropriateness**	**32.50**
1.2.2.1.1. *Search Capability Appropriateness*	50
1.2.2.1.2. *Weather Forecast Capability Appropriateness*	50
1.2.2.1.3. *Current Temperature Capability Appropriateness*	0
1.2.2.2. **Completeness**	**0**
1.2.2.2.1. *Current Weather Itinerary Trip Advisor Completeness*	0
1.2.2.2.2. *Weather Forecast Itinerary Trip Advisor Completeness*	0

As output a *non-functional requirements tree* is produced. This requirements tree specification for evaluating the mashup WebApp is represented in the 1st column of Table 6. Additionally, Table 7 defines most of the attributes shown in Table 6 –recalling that *functional quality* (sub-)characteristics were defined in Table 2.

Note that we could have selected other characteristics for EQ, but we purposely instantiated these (sub-)characteristics from the 2Q2U modeling framework as per our information need via the GOCAME's process and C-INCAMI framework capabilities.

5.2 Designing and Implementing the Measurement and Evaluation

This sub-section summarizes results of activities that are performed from designing the measurement and evaluation (A2 and A4 in Fig. 8) through implementing the measurement and evaluation (A3 and A4 activities).

The above requirements tree is 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 quality models.

Table 7: Definition of many attributes that are in the *Requirements Tree Specification* in Table 6.

Attribute	Definition
Consistency of forecasted temperature against its source (1.1.1.1)	Degree to which the forecasted temperature of a city shown by the WebApp component matches the temperature shown by the component's source.
Composition Reputability (1.1.2.1.1)	Degree to which the owner of the composition is trustworthy.
Component's Source Reputability(1.1.2.1.2)	Degree to which the owner of a component source is trustworthy.
Verifiability (1.1.2.2.)	Degree to which the enterprise/developer/version/date of a component source can be verified.
Component Integratedness on clicking (1.2.1.1.1)	Synchronized and harmonious behavior among components (slaves and masters) after clicking on weather icons on one master component.
Component Integratedness on searching (1.2.1.1.2)	Synchronized and harmonious behavior among components (slaves and masters) after searching for a city by means of a filter mechanism.
Current Weather Capability Beneficialness (1.2.1.2.1)	Degree to which the current weather capability contributes to be perceived as useful for a given user.
Search Capability Appropriateness(1.2.2.1.1)	Degree to which the searching mechanism (as filter) fits an intended user goal.
Current Weather Itinerary Trip Advisor Completeness (1.2.2.2.1)	Degree to which the current weather capability shows a complete weather-iconed itinerary between two cities regarding the intended user goal

Table 8: Specification of all metrics involved in *Components Integratedness on Clicking Ratio* indirect metric.

Attribute Name: *1.2.1.1.1 Component Integratedness on clicking*
Indirect Metric Name: Component Integratedness on Clicking Ratio (CICR)
Objective: Determine the degree of integratedness that slave components keep after clicking on weather icons over the master component.
Calculation Method:

Function Specification: CICR =(($\sum_{i=0}^{n} CICi$) / (#CCW*MAX_SV))*100,
where MAX_SV is a parameter that represents the maximum scale value, which is 2 regarding CIC; and n = #CCW
Numerical Scale:
Representation: Continuous ValueType: Real ScaleType: Proportion
Unit: Name: Percentage Acronym: %
Related metrics:
CIC (Component Integratedness after clicking);
#CCW (Total Number of Cities with Checked Weather)

Attribute Name: Component Integratedness after clicking on a specific weather icon
Direct Metric Name: Component Integratedness after clicking (CIC)
Objective: Determine if slave components keep harmony and synchronization after clicking on a specific weather icon (from #CCW) on the master component.
Measurement Method:
Type: Objective
Specification: Given a representative sample of weather-iconed cities (as per #CCW metric specification), check if after clicking on a weather icon on the master component: 1) the mini-map slave component shows the area of the world that corresponds with the clicked weather icon of the city; and 2) the weather forecast slave component shows information about the clicked weather icon of the city. Then:
CIC = 0 if neither mini-map slave component nor weather forecast slave component are integrated with respect to the master; CIC = 1 if just mini-map slave component or weather forecast slave component is integrated with respect to the master; CIC = 2 if all components are integrated.
Numerical Scale:
Representation: Discrete ValueType: Integer ScaleType: Proportion

Attribute Name: Amount of Cities with Checked Weather
Direct Metric Name: Number of Cities with Checked Weather (#CCW)
Objective: The total number of cities of a given sample or population taken from the master component, in which each city has the weather icon.
Measurement Method:
Type: Objective
Specification: Choose a number of cities that has a weather icon over them.
Numerical Scale:
Representation: Discrete ValueType: Integer ScaleType: Absolute

In a given M&E project, for each attribute of the requirements tree a metric should be selected. Table 8 shows the metric specification for the *Component Integratedness on clicking* (coded *1.2.1.1.1* in Table 6) attribute, which is defined as "synchronized and harmonious behavior among components (slaves and masters) after clicking on weather icons on one master component". The used metric to quantify *1.2.1.1.1* is indirect, which includes two related direct metrics. The objective of the CICR indirect metric is to "determine the degree of integratedness that slave components keep after clicking on weather icons over the master component". Data collection for direct metrics was

made by several expert evaluators in the last week of Dec., 2011. Note EQ measurement was made by inspection, as opposed to user-centered observation-based evaluation commonly used for usability studies where the goal is to increase study participants to increase statistical validity.

From the evaluation design standpoint, for the above *1.2.1.1.1* attribute, the elementary indicator named *Performance Level of the Component Integratedness on clicking* interprets the metric's value of it. Note that each elementary indicator interprets the level of satisfaction for each attribute regarded as an elementary non-functional requirement. Therefore, a new scale transformation and decision criteria (in terms of acceptability ranges) should be defined. In our study, we used three acceptability ranges in a percentage scale: a value within 40-70 (a marginal –yellow- range) indicates a need for improvement actions; a value within 0-40 (an unsatisfactory –red- range) means changes must take place with high priority; and a score within 70-100 indicates a satisfactory level –green- for the analyzed attribute.

Details of elementary and global evaluation, as well as the weighted additive aggregation model –as used in this study- to calculate indicators, can be referred elsewhere [24]. Table 6 depicts the elementary, partial and global indicators' values in its 2nd column, as well as the acceptability levels achieved.

5.3 Analysis of Results

Data and information coming from measurements and evaluations can be used for analysis, recommendation, comparison, possibly selection activities and, ultimately, for decision making. A particular M&E project records all data, metadata, and information coming from metrics and indicators as well as non-functional requirements and context specifications and values.

As indicated in sub-section 5.1.1, the established purpose to perform the evaluation –as proof of concept- is to *understand* the current level of *functional quality* for a concrete *mashup WebApp* from the *traveler* user viewpoint. The underlying hypothesis is that at the higher level of (sub-)characteristics they have to accomplish the satisfactory acceptability range –at least its lower threshold values which are >70.

Based on the results shown in the 2nd column of Table 6, we can observe that the *Functional Quality* characteristic in woozor reached an unsatisfactory acceptability level (32.26%), which means improvements must be made. Taking into account its two sub-characteristics of higher level, *Functional Accuracy* (1.1) met 22.02, and *Functional Suitability* (1.2) reached a marginal acceptability level with an indicator value of 42.50%. In turn *Added value* (1.2.1) ranked 68.75 % because the *Integratedness* sub-characteristic and particularly its two attributes viz. *Component Integratedness on clicking (1.2.1.1.1)* and *Component Integratedness on searching (1.2.1.1.2)* met both 100%, i.e. they totally satisfied the elementary requirements –recall the *1.2.1.1.1* metric specification in Table 8. However, *Added value* is influenced also for *Beneficialness* (1.2.1.2), where *Current Weather Capability Beneficialness (1.2.1.2.1)* is perceived useless (ranked 0%) because woozor does not support so far current (right now) updated temperatures in cities, only the maximum temperatures in the current day and night into the master component, and maximum and minimum temperatures in the weather forecast component –extending the weather forecast for 5 days.

Therefore, *Added value* was downgraded in its score for this last weak point. Also, it is important to stress that even if components are totally integrated (100%), *Weather Forecast Capability Beneficialness (1.2.1.2.2*) scored 75%.

Regarding *Coverage (1.2.2)* and, particularly, the *Appropriateness (1.2.2.1)* attribute named *Weather Forecast Capability Appropriateness (1.2.2.1.2)* reached a marginal acceptability level where this functionality is partially appropriate (50%) because the 5-days forecast does not have the option to show all days at once, but just day by day. On the other hand, the *Completeness (1.2.2.2)* sub-characteristic scored 0% since, for example, the attribute named *Weather Forecast Itinerary Trip Advisor Completeness (1.2.2.2.1*) is a missing capability, which is intended to show weather-iconed itinerary between two cities regarding the traveler user.

Lastly, note that the metric specification and the measured value in conjunction with the indicator specification and value help evaluators to understand the reasons why a given elementary indicator achieved for instance an unsatisfactorily acceptability level, then facilitating recommendations for further improvement. Additionally, note that many metrics can be designed for the same attribute, but only one must be selected for actual measurement design. The selected metric should be one that is cost-effective for the evaluation purpose and available resources.

6 Conclusions and Future Work

The ultimate objective of our research and for this manuscript is to demonstrate the means –i.e. the evaluation approach- to gauge newer generation WebApps. In this pursuit, however, we found that we needed to modify existing quality models in order to conduct such evaluations to incorporate the particular characteristics of new eras of WebApps. And in doing so, past research was relied upon for modification and updating such as 2Q2U quality models. We generalized the instantiation such that the same evaluation approach can be used for current and future generation WebApps as well.

In summary, throughout the manuscript we have developed and discussed the particular contributions stated in the Introduction Section:

- A discussion in Section 2 of strengths and weaknesses of recently issued ISO 25010/25012 standards related to software/Web quality models. Additionally, a particular discussion of quality models related to some system domains, such as Web portals and mashup WebApps. At this point, in sub-section 2.3, we have argued that general EQ and QinU models can be specified while still providing the capability for tailoring and instantiating quality models for specific purposes and particular entity categories and entities.
- The rationale for including new (sub-)characteristics to the 2Q2U quality modeling framework and their added value for evaluating not only older but also newer generations WebApps –mainly in Section 3. At this point, the foundations were raised in general but giving details on the rationale for including some particular sub-characteristics for composition-oriented WebApps. Also, the modeling of context out of

the scope of quality models was finally justified in Section 4, following the principles of the research made in [19].

- An instantiation for illustration purposes (Section 5) in order to show practical usage of the 2Q2U EQ model with sub-characteristics and measurable attributes to evaluate *functional quality* for mashup WebApps. Moreover, the instantiation was systematically guided by an *evaluation approach* which is based on: i) a general *quality modeling framework* –where 2Q2U is a subset; and ii) *measurement and evaluation strategies*. We have overviewed this approach in Section 4 and applied in Section 5 by using the GOCAME strategy.

Using 2Q2U v2.0 we have recently performed a case study [15] showing the evaluation of *communicability* and *sense of community*, two important QinU characteristics for new generation social WebApps, in addition to *effectiveness*, *efficiency*, *learnability in use*, *trust*, *pleasure* and *comfort* sub-characteristics. Note also that 2Q2U (sub-)characteristics such as *actual usability*, *learnability in use*, *effectiveness*, *efficiency* for QinU, and *operability*, *information suitability* and *learnability* for EQ as well as *influences* and *depends on* relationships between quality models were documented elsewhere [16, 22].

We are aware that for assuring the usefulness of 2Q2U enhanced quality models, particularly for the correctness and the completeness of the new set of proposed (sub-)characteristics more representative sets of selected WebApps are needed. So as future work, we plan to perform new case studies regarding this.

Acknowledgments

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. Abran A., Surya W., Khelifi A., Rilling J., Seffah A., Robert F.: Consolidating the ISO Usability Models. 11th annual Int'l Software Quality Management Conference, 2003.
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.: Classifying and selecting UX and usability measures, In Proc. of the 5th COST294-MAUSE Workshop on Meaningful Measures: Valid Useful User Experience Measurement, 2008.
5. 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.
6. Cappiello C., Daniel F., Matera M., Pautasso C.: Information quality in mashups. IEEE Internet Computing, 14(4), pp. 14-22, 2010.
7. Grossman T, Fitzmaurice G, Attar R.: A survey of software learnability: metrics, methodologies and guidelines, Proceedings of the 27th Int'l Conference on Human factors in computing systems, pp. 649-658, 2009.
8. 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.
9. Herrera M., Moraga M.A, Caballero I., Calero C.: Quality in Use Model for Web Portals (QiUWeP). ICWE'10 Proceedings of the 10th international conference on Current trends in Web Engineering, Springer, LNCS 6385 pp. 91-101, 2010.
10. ISO/IEC 25010: Systems and software engineering. Systems and software Quality Requirements and Evaluation (SQuaRE). System and software quality models, 2011.
11. ISO/IEC CD 25010.3: Systems and software engineering. Systems and software Quality Requirements and Evaluation (SQuaRE). System and software quality models, 2009
12. ISO/IEC 25012: Software engineering -- Software product Quality Requirements and Evaluation (SQuaRE) -- Data quality model, 2008
13. ISO/IEC 9126-1: International Standard, Software Engineering - Product Quality - Part 1: Quality Model, 2001.
14. ISO 9241-110: Ergonomics of human-system interaction, Part 110: Dialogue principles, 2006.
15. Lew P., Qanber Abbasi M., Rafique I., Wang X., Olsina L.: Using Web Quality Models and Questionnaires for Web Applications Evaluation. To appear in IEEE proceedings of QUATIC, Lisbon, Portugal, 2012.
16. 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.
17. 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.
18. 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.
19. 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.
20. Murugesan, S.: Web X.0: A Road Map, Handbook of Research on Web 2.0, 3.0, and X.0: Technologies, Business, and Social Applications, Murugesan Ed., IGI Global, pp. 1-11, 2010.
21. Nielsen J: Mobile Usability Update, Jakob Nielsen's Alertbox, Sep 26, 2011, Available at http://www.useit.com/alertbox/mobile-usability.html, accessed by Dec, 2011
22. Olsina L., Lew P., Dieser A., Rivera B.: Using Web Quality Models and a Strategy for Purpose-Oriented Evaluations, Journal of Web Engineering, Rinton Press, US, 10 (4), pp. 316-352, 2011.
23. 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.
24. Olsina L., Papa F., Molina H.: How to Measure and Evaluate Web Applications in a Consistent Way. Ch. 13 in: Modelling and Implementing Web Applications, Rossi G., Pastor O., Schwabe

D. & Olsina L. (Eds), Springer-Verlag HCIS, London, pp. 385–420, 2008.
25. 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, 2002.
26. Prates R., de Souza C., Barbosa S.: A method for communicability evaluation of user interfaces. Interactions, ACM, 7(1), pp. 31-3, 2000.
27. Rossi G., Urbieta M., Ginzburg J., Distante D., Garrido A.: Refactoring to Rich Internet Applications. A Model-Driven Approach. In: IEEE proceedings of 8th Int'l Congress on Web Engineering (ICWE'08), NY, USA, pp. 1-12, 2008
28. Santos P. J., Badre A. N.: Discount Learnability Evaluation. Graphics, Visualization & Usability Center, Georgia Institute of Technology, 1995, available at http://smartech.gatech.edu/bitstream/1853/3574/1/95-30.pdf, accessed by 2011/12/01