FACULTEIT ECONOMIE
EN BEDRIJFSKUNDE

TWEEKERKENSTRAAT 2
B-9000 GENT
Tel. : 32 - (0)9 – 264.34.61
Fax. : 32 - (0)9 – 264.35.92

WORKING PAPER

Towards a Quality-Aware Engineering Process for the Development of Web Applications

Cristina Cachero [1]

Geert Poels [2]

Coral Calero [3]

Yolanda Marhuenda [4]

May 2007

2007/462

[1] IWAD Research Group, University of Alicante, Spain
[2] Dept. Management Information, Operations Management and Technology Policy, Ghent University, Belgium
[3] ALARCOS Research Group, University of Castilla-La Mancha, Spain
[4] University of Elche, Spain

D/2007/7012/33

Towards a Quality-Aware Engineering Process for the development of Web Applications

Cristina Cachero[1], Geert Poels[2], Coral Calero[3], Yolanda Marhuenda[4]

[1] IWAD Research Group, University of Alicante. Spain

ccachero@dlsi.ua.es

[2] Faculty of Economics and Business Administration, Ghent University. Belgium

geert.poels@ugent.be

[3] ALARCOS Research Group. University of Castilla-La Mancha. Spain

Coral.Calero@uclm.es

University of Elche, Spain

yolanda.marhuenda@umh.es

Abstract

It is widely recognized that Web Engineering (WE) practices lack an impact on industry. One of the reasons for this fact is that Web applications developed with WE methodologies have not proven a better quality than those developed with creative practices. In this technical report we claim that one way to change such perception is including specific quality management activities as part of the WE process. In order to perform this inclusion in a sensible way, in this technical report we explore principles and achievements that, uncovered in different Web quality lines of research, provide insights into how to deal with quality in each of the different workflows that a typical WE process defines, from requirements to implementation. Also, in order to preserve the (semi-)automatic nature of WE processes, we propose the definition of measurable concepts, measures and decision criteria in a machine-readable way that allows for the automation of the quality evaluation process, thus preserving the MDE nature of WE processes. In this way we are providing the user of a WE methodology with the advantages associated with managing quality from the early stages of development with little extra development costs.

Keywords

Web Quality assessment, Web Engineering, MDA, Empirical Validation

1. Introduction

It is an avowed fact that WE practices lack an impact on industry [32]. We can name at least three possible reasons for this fact:

- It has been empirically assessed that, generally speaking, Web stakeholders' interest is focused, besides cost and time-to-market, on usability and visual appearance [11] which just pay a secondary role in Web methodologies.
- From the point of view of Web developers, it is too hazardous to decide on the use of a tool that systematizes the Web application construction, due to the small amount of reliable information available on methodologies, technologies and tools. In fact, just around 5% of the claims of the WE field about improved time-to-market and reduced development costs are based on actual facts, even if it is well-known that conventional wisdom, intuition, conjecture and proofs of concepts are known not to be reliable sources of credible knowledge [36].
- Finally, and from the point of view of the final user of the application, the use of a WE methodology does not guarantee any kind of improvement on the quality in use of the deployed applications. Again, this is partly due to lack of reliable data that empirically supports the WE claim of better quality in use of the developed applications. Actually, WE development processes do not usually include specific support for quality requirements. As far as we know, only WebSA, which tackles architectural issues that may influence some aspects related to the final quality in use of the application, is an exception in this sense.

We believe that this situation is a clear sign of immatureness of the field, situation that should be reversed if we aim at increasing the confidence of industry on our methodologies. We therefore support Kitchenham et al. claim [29,30] that Evidence Based Software Engineering (which includes Evidence Based Web Engineering EBWE) is necessary in order to (1) help industry practitioners to make rational decisions about technology adoption and (2) increase the acceptability of software methodologies.

Unfortunately, the WE community is not yet familiar with either systematic quality evaluation issues or empirical research, and therefore tools and guidelines to ease this shift are necessary. Concretely, for EBWE to stop being a utopia, a general framework is needed in WE to:

- Guide the way in which WE methodologies can provide empirical evidence of the quality of their proposed development process (i.e. **process quality**), and the advantages it provides to **analysts, designers, developers and maintainers** compared to ‘creative’ approaches - for instance the extent to which it improves developers’ productivity-. Given the fact that WE processes are commonly based on the MDA paradigm, this process quality involves assessing the quality of the (semi-) automated transformations defined.
- Guide the way in which WE methodologies can provide empirical evidence of the quality of the WE intermediate artefacts (i.e. **internal product quality**), which correspond with the intermediate models that are generated as part of the process, and how they help indeed to better manage the complexity of Web development.
- Guide the way in which WE methodologies can provide empirical evidence of the quality of the application delivered using such methodologies under testing conditions (i.e. **external product quality).**
- Guide the way in which WE methodologies can provide empirical evidence of how assuring a certain degree of internal and external product quality offers advantages to acquirers and end-users by affecting the user perception under real conditions of use of the quality of the deployed application (i.e. **quality in use**)
- Guide the way in which WE methodologies can incorporate quality evaluation issues at every level of abstraction into their development processes without hampering the cost and/or time to market of the delivered application.

Even if all these aspects are important, we have already mentioned that Web stakeholders’ main interests lie in usability, cost and time to market of the deployed application, which correspond to the last two aspects of our enumeration. Hence, in this report we will centre on these two aspects, thus leaving the provision of frameworks to empirically backing the remaining assumptions as open lines of research.

For the demonstration of how the quality of intermediate models affects the end-user quality perception, in this paper we propose the use of a WE quality evaluation process that is based on empirically-validated WE quality models (WE-QM), one for each intermediate WE artefact. We also provide insights about how the relationship between the quality of the different WE models (internal product quality) and the quality in use of the application generated from those models (end-user quality perception of the application delivered based on the WE models) can be correlated in an empirical way.

In order to construct our guiding framework, we will start by clarifying the most important quality concepts that we will use in the remaining of the paper.

1.1. Quality Concepts

All along the years, many different definitions of quality have come up. Garvin[18] proposes five different **perspectives of quality**: transcendental view, user view, manufacturing view, product view and value-based view. From them, two are especially relevant from the point of view of WE [46]:

- **Conformance to specification (Manufacturing view)**. Quality defined as a matter of products whose measurable characteristics satisfy a fixed, in beforehand defined specification
- **Meeting customer needs (User view)**: quality defined, independent of any measurable characteristic, as the product capability to meet customer expectations, be them explicit or not

Let's illustrate how to deal with these two perspectives with an example. A Web application with, let's say, six navigation steps from the home page to a given target page hasn't necessarily a low level of quality in use *per se*, but it turns into a low-quality-in-use application if it causes the user to be less effective, less efficient or even to feel less comfortable with the application. If this is the case, when asked about the problem, she wouldn't probably be able to identify its exact nature. All she would be able to say would be that she is not comfortable using the system or, at most, that the way to achieve her goals is too cumbersome. However, actions should be taken to fix the problem. While the developer may infer from the dissatisfaction of the user that the problem is paths too long and therefore could work on shortening the path that caused the problem at implementation level, it would have been preferable if someone had translated the *customer need* (navigate fast) into a *conformance to specification* quality requirement such as 'the maximum depth of any navigational map to three levels'. Even more interesting would have been that such requirement had been handed in to the developer while he was designing the navigation model of the application, so that the error would have been prevented rather than detected.

In order to evaluate quality we need an **evaluation instrument**. One possibility is to use a certain **quality model**. A quality model is defined in ISO as the *set of characteristics and the relationships between them which provide the basis for specifying quality requirements and evaluating quality*. According to the ISO/IEC 9126 [21] and ISO/IEC 14598 [23], the overall objective of any quality evaluation process should be 'meeting customer needs'. Provided that we narrow the term 'customer' to that of 'end-user', this concept of quality from the end-users' perspective is what the ISO/IEC 9126 standard defines as 'quality in use', that is, *the efficiency, productivity, security and satisfaction with which users use the application to satisfy specific goals under specific conditions*. From this definition it is possible to extract the four characteristics (efficiency, productivity, security and satisfaction) that, according to the ISO/IEC 9126, make up a 'quality in use' product quality model. On the other hand, the Web application, as any other software product, presents certain characteristics that can be evaluated before it has been deployed, namely (again according to ISO/IEC 9126) usability, functionality, reliability and efficiency[1]. All these characteristics, together with the elements (measures, decision criteria and so on) that permit to evaluate them, make up the ISO/IEC 9126 internal/external **product quality model**.

Quality has been a continuous concern for Web developers, due to the necessity for most of these applications to keep the audience coming back to the site [16]. Even when the use of the Web system is mandatory (e.g. when we deal with traditional business applications accessed through a Web interface) the fact that such Web system presents a high quality in use has been traditionally associated with increased gains in terms of user effectiveness, efficiency and satisfaction. Talking in terms of the OMG Standard Metapyramid [22] (see main subdivisions in Figure 1), the Web quality evaluation effort has been traditionally centred on the M1-Implementation level (measures over the application code, without running it) and M0-test level of abstraction (code running under testing conditions). . These two levels are reflected in the myriad of design guidelines [40] and automated measures [24] that have been gathered in literature as relevant for Web development. While guidelines are, for the most part, ambiguous and

[1] We have intentionally left out of this list the ISO/IEC 9126 characteristics of maintainability and portability, which we consider relevant for other stakeholders different from the final user (analysts, designers, developers, maintainers and so on.)

hard to follow [26], Web measures over the implementation have showed themselves as a systematic and accurate way of evaluating products. In fact quality assessment of Web interfaces with the help of measures matches in some cases up to 80% of the results based on expert evaluation of the same Web pages [25]. Examples of traditional Web measures at M1-Implementation level include fonts, colours, position of menus and so on. Examples of measures that can be applied at the M0 level of abstraction include use of network resources, page load time, etc.

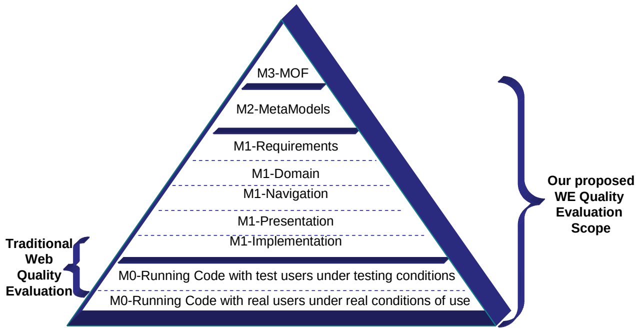


Figure 1: The OMG Standard Metapyramid with additional WE subdivisions to distinguish among different levels of abstraction at M1 level (adapted from ISO 10027)

However, performing improvements at such a late stage of development is avowed to have a negative impact on the final product cost and quality [6]. In fact, the cost associated with removing a defect during design is on average 3.5 times greater than during requirements; at implementation stage the effort associated with removing the same defect can be up to 50 times greater, and up to 170 times greater after delivery. Other empirical studies have shown that moving quality evaluation effort up to the early phases of development can be 33 times more cost effective than testing done at the end of the development [38]. Also, such measures have been traditionally devoted to assessing product usability issues, which is just one of the characteristics of the internal/external quality model that may affect the quality in use of the resulting application.

But how can we council an early evaluation of the main internal product characteristics (that implies a 'conformance to specification' perspective) with the ultimate goal of 'meeting customer needs' that we have aforementioned? Fortunately, the ISO set of quality standards establishes an interesting relationship between these two quality perspectives; it sets that the 'conformance to specifications' degree of a given software product (which includes not only the code but also the intermediate artefacts generated as part of a WE process), which can be evaluated through an internal/external quality/model, may be a valid predictor of the ability of the product to meeting user needs (quality in use), even if the exact accuracy of such prediction is an open issue that depends on variables such as type of application or context of use. Additionally, departing from an internal/external quality model guarantees that every quality characteristics, and not only usability, are considered. Otherwise stated, improving the internal/external quality of a Web application through the use of an internal/external product quality model may positively influence the quality in use of such product.

This assumption means that Web quality in use can be worked on from the early stages of Web development. Analysts and designers work on tangible specifications, and not on intangible needs. Therefore, before starting the development process 'customers' needs' (called external quality requirements in the ISO/IEC 9126) should be translated into 'specifications' (called internal quality requirements in the ISO/IEC 9126) that the analysts/designers can systematically check. Then, and provided that the relationship between internal and external quality requirements has been empirically established, we can be confident that the focus all along the development process is still kept on the final goal of meeting customer needs. Again in terms of the OMG Standard Metapyramid, with the set of models and meta-models provided by WE, the set of measures at M1 level of abstraction can be broadened to include new measures on requirements models [45], domain models [19], navigational models [1,3] and presentation models. Some examples of measures at these new levels of abstraction are cohesion of requirements, number of domain classes, complexity of domain relationships, number of navigational classes, density of the navigational map, number of widgets included in the presentation model, coherence in the use of widgets, colours and fonts, and so on. Also, measures on implementation models and at M0 levels of abstraction are still valid. Going one level of abstraction further, we could even define measures at M2 level, e.g. number of meta-model concepts involved in the M1 models that support the M0 running Web application.

All these possibilities are however hampered by the aforementioned fact that Web development is still commonly based on 'creative' approaches, where early artefacts are scarce and lack the necessary rigor to perform measurements on them. This fact in turn causes that the main body of existing measures [11, 24] is still centred on implementation issues. Therefore in this paper we claim that, in order to manage Web quality from early stages of development, WE methodologies must increase their presence in industry. Only with WE practices would it become possible to build up the characteristics that make up the quality in use of the Web application during the whole Web development process, and not only once the deployment phase has been attained. This fact is implicit in the definition of WE, which involves *the application of systematic, disciplined and* ***quantifiable*** *approaches to the cost-effective development and evolution of high-quality applications in the World Wide Web* [20].

Building up a (from the end-user perspective) high-quality Web application all along the WE process means that quality issues should be taken into account while developing each outgoing artefact, from the requirements model to the final application to be delivered. This approach implies a shift from the traditional WE quality assessment perspective to a WE Total Quality Management (TQM) approach [49]. Briefly speaking, adopting a Total Quality Management perspective means setting the focus on preventing rather than detecting errors, with the ultimate aim of reducing the reliance on code inspections as a way of achieving quality [37]. Our assumption is that providing practitioners with WE methodologies that assure a certain degree of quality of the application delivered is likely not only to support some of the WE traditional claims, but also to increase acceptance rate of the WE technology in industry. This in turn would provide the WE community with more data on which to refine their knowledge about when and how to use each WE methodology (see Figure 2).

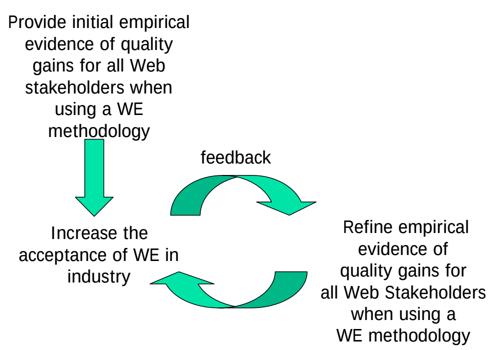


Figure 2. Empirical studies as the basis for a broader industry acceptance of WE methodologies

In order to perform such inclusion of quality concerns in existing WE methodologies in a sensible a consistent way, we have based our proposal on principles and achievements that, uncovered in different quality lines of research, provide insights into how to deal with quality in each of the different workflows that a typical WE process defines, from requirements to implementation. Next we briefly present our main findings.

1.2. Related Work

Quality models for software products are far from scarce. Well known pioneer models include McCall [35], Boehm[5], Dromey [13] and ISO/IEC 9126 [21]. All of them centre on measurable elements over the implementation of the software product on one hand, and on the (abstract) quality characteristics on the other hand, and try to establish relationships among both dimensions. The General Electrics Model of McCall distinguishes among three major perspectives for defining the quality of a software product, which correspond with different stakeholders working at different levels of abstraction in the development lifecycle. Among them, the product operations perspective refers to the end user perspective during the usage of the application. Boehm's Quality Model also defines three perspectives, from which 'as-is-utility' refers to the end-user perception of the software during its usage (at execution time). Dromey's Quality Model presents a product based quality model that recognizes that quality evaluation may differ for each product. An important addition of Dromey's to the quality model field is that he defines a quality model building process that involves five steps, from choosing the quality characteristics relevant for the evaluation to evaluating the product and identifying weaknesses[14]. Last, the ISO/IEC 9126 standard includes a general framework with characteristics, sub-characteristics and measures that can be used to evaluate a software product. Due to the spread use of the ISO family of standards, many proposals have aimed at tailoring/refining/improving the ISO quality models. For example, Quint2[42] is an example of a quality model that regards the ISO/IEC 9126 as a valid but incomplete quality model, and therefore tries to complete it with additional features.

There are various proposals of specific Web quality models, most of them tackling the Web idiosyncrasy from the 'meet the user needs' perspective [41][11][39][12][2]. From them, only [12] and [2] promote considering other artefacts (apart from code) that may take part in the WE development cycle, and none of them provide independent quality models for each level of abstraction. These approaches can however be refined and complemented by research in conceptual modelling quality (e.g. Lindland et a. framework [34], Krogstie et al. framework [31] and Moody and Shanks framework

[37], which provides further insight into how the quality concept can be dealt with at higher levels of abstraction.

Last but not least, Web quality evaluation needs to be performed following a well defined quality evaluation process. Some well known Web quality evaluation processes are WebQUEM [41] and WebTango [24]. The main drawback of these processes is that they assume that Web quality evaluation is performed on the deployed application. Only [37] and [2] present a broader perspective and try to conciliate Web quality evaluation with a general WE development process.

Next, we present the challenges all these fields pose, and how we propose to integrate them in a single, consolidated proposal in the context of WE.

1.3. Research Issues

When trying to operationalize all the myriad of different quality models and quality evaluation processes that have been proposed in literature, several theoretical and practical issues arise [38, 44]:

- **P1: Terminology inconsistencies.** Most approaches (the exception being those based on theoretical grounds) lack a definition for quality concepts that is precise and concise. For instance, while in the ISO/IEC 9241-11 usability refers to the end-user perception as a whole (and therefore encompasses efficiency effectiveness and satisfaction), in the ISO/IEC 9126 end-user perception is referred to as 'quality in use', and usability is only one of the internal characteristics that may affect such quality in use.
- **P2: Incomplete definition.** Most quality models are outlined but not fully developed. All define measurable concepts, some of them also attributes, few of them include (most often partial) measures and scarcely any defines decision criteria or indicators. Therefore intensive work is necessary by the people using them to get them operational. An example of a quality model suffering from this problem is the ISO/IEC 9126 itself.
- **P3**: **Lack of focus.** Most quality models provide an extensive (and mostly tangled) coverage of stakeholders and levels of abstraction. An example of such assertion is the QUIM model [44], which aims at being a consolidated usability model that integrates all possible perspectives. As another example, WQM [11] covers 10

factors, 26 subfactors and 127 measures that may be related to any WE artefact, from analysis to implementation.

- **P4: Lack of simplification and validation.** Quality models that include measures usually pay little attention to the theoretical/empirical validation of the included measures. Furthermore, although empirical research has shown that a few measures (three in [37]) most times suffice to obtain significant gains in quality, quality models usually include an extensive, even redundant set of measures. Such verboseness unnecessarily increases the complexity and therefore hampers the potential usefulness of the quality models.
- **P5: Interdependencies and measure interpretations not clear.** In most quality models (again the notable exception being those that are based on theory), the degree of influence of individual internal quality factors on the quality in use of the application, as well as their interdependencies, are not well established. For example, the role of learnability versus understandability in the usability model presented in [2] is an open issue. Also, little information is provided on how to interpret measurement results.
- **P6: Lack of integration with current practices.** Quality management is not integrated into current WE practices
- **P7: Disregard for standard process quality frameworks.** Most quality models define criteria and, in some cases, measures for evaluating products (error detection), but not how to develop products in a way that assures a certain level of quality (error prevention).
- **P8: Lack of guidelines for improvements.** Even in the case of being able to evaluate a certain Web characteristic, to our knowledge extent no quality model provides a clue about how (by means of which changes in the artifacts) such evaluation could be improved, let alone to which extent such changes may affect the evaluation of other characteristic included in the quality model.
- **P9: Lack of tool support.** Although most Web measures are automated, tool support for the definition of quality models and, even more important, for the automation of the measurement process on a given Web application is still an open issue.

In order to overcome these problems, certain requirements should be preserved when defining WE quality models and integrating them with WE development processes:

Requirement 1. WE quality models should be expressed using a set of clear concepts with clear semantics and relationships, in order to ease their understanding and assure a structural coherence. This palliates problems P1, P2 and P3

Requirement 2. WE quality models should be defined taking into account a specific stakeholder and a specific software artifact. This palliates problem P3.

Requirement 3. WE quality models should be empirically validated before being included in the WE process. This palliates problems P4 and P5

Requirement 4. WE quality models should be accompanied by a WE quality evaluation process. Such process must be defined and seamlessly integrated with the WE development process. This means following an MDE approach. This contributes to overcome problems P6 and P7

Requirement 5. For the definition of the WE quality evaluation process, standards should be followed when possible. This alleviates problem P7

Requirement 6. Guidelines should be provided when possible to improve WE artifacts according to the WE quality artifact under consideration. Such guidelines should also if possible preserve the semi-automatic nature of the WE process. This contributes to solving problem P8.

Requirement 7. The integration of WE quality models in the WE process should always be accompanied by tool support. Basing such integration on standards (for which tool support is provided by third parties) simplifies the task of finding such tool support, therefore contributing to alleviating problem P9

In the remaining of this report we will present how our proposal fulfils all these requirements.

1.4. Outline of the paper

In this paper we propose the inclusion of specific quality evaluation activities as part of the WE process that is defined in the main WE methodologies. This inclusion means, as we have outlined before, a shift towards a TQM WE discipline, and aims at assuring that Web applications developed with WE methodologies are of better quality than those developed with creative practices. In order to produce such shift without incurring in the problems presented in Section 1.3, in Section 2 we propose to operationalize WE quality models as WE measurement models (WE-MM) that are based on an ontology-based measurement meta-model. This ontology support contributes to avoiding terminology inconsistencies (Requirement 1), while the use of a meta-model assures the syntactic correctness of the WE measurement model (including completeness restrictions and focus control on specific stakeholders and specific WE artefacts, Requirement 2). Section 3 justifies why, before including any WE quality model as part of this WE quality-aware development process, such quality model should be empirically validated to assure that it is minimal yet complete, and that all the included measures are also valid (Requirement 3). For this purpose, it presents how such empirical validation could be performed. Section 4 integrates the different WE quality models and corresponding WE measurement models into a WE quality evaluation process that can be seamlessly integrated with a generic WE development process (Requirement 4). Section 5 presents how such WE quality evaluation process is compliant with the ISO 14598 (Requirement 5). Section 6 briefly explains how the use of WE measurement models and standard QVT-based transformation rules permit the definition of both evaluation and evolution actions over WE models (Requirement 6). Also, they make possible that the resulting WE quality-aware development process still preserves the (semi-)automatic nature of the WE traditional processes (Requirement 7). Last, Section 7 presents conclusions and further lines of research.

Next, we present the ontology and the meta-model that we propose to use as a basis to instantiate WE measurement models that reflect the underlying WE quality model.

2. Definition of WE measurement models following an ontology and a meta-model

As we presented in Section 1.3, one of the problems that existing quality models face is terminology inconsistencies (P1). In order to overcome such problem we need a common vocabulary both to express WE concepts and to express quality concepts. Such common vocabulary usually comes in ontology form.

Ontologies, defined as explicit, formal and shared specifications of a conceptualization, have been widely used in Software Engineering [43]. Ontologies are descriptive in nature. They try to identify all the elements that are relevant in a given domain, and provide an exact definition of each of them. Ontologies also identify the relationships among the elements and what these relationships mean. Such a common vocabulary is necessary for several reasons, such as [36]:

- To allow researchers and practitioners to understand and cooperate with each other
- To provide the basis for gathering, validating and analyzing trustworthy data
- To allow for the summary of findings from several empirical studies
- To improve the research and reporting process

The use of an ontology not only avoids vocabulary conflicts and inconsistencies but also establishes the adequate level of detail for the definition of each concept. While the definition of a WE ontology is in its first stages of development and remains out of the scope of this technical report, the greater maturity of the measurement field causes a proposal for a Software Measurement Ontology (SMO) to be already available.

.

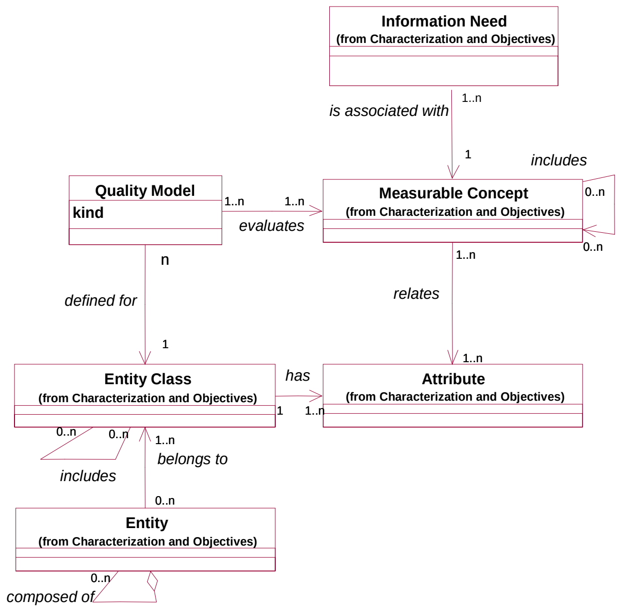


Figure 3: UML Diagram of the 'Software Measurement Characterization and Objectives' Sub-Ontology

The SMO was first presented in [17] and has since then been used to define some quality models [39]. Our reason for choosing this ontology has been twofold. The first reason is that this ontology comes together with a Software Measurement Meta-model (SMM) [15]. Metamodels, unlike ontologies, are prescriptive in nature, and aim at identifying how a given domain must be built, explaining the kind of entities and how they are interconnected in a given context. The SMO provides to the SMM the degree of ‘completeness’ and ‘shareness of concept’ that common meta-models, defined in the context of a particular organization, lack. This ‘shareness of concept’ is bound to simplify and homogenize the way in which such meta-model is instantiated to define machine-readable measurement models. The second reason for choosing the SMO is that it is, to our knowledge extent, the most complete one that explicitly characterizes the relationships between abstract quality concepts on one side and concrete software measurement strategies on the other.

The SMO ontology is structured around four packages, namely:

- Software Measurement Characterization and Objectives, which includes the concepts required to establish the scope and objectives of the software measurement process
- Software Measures, which aims at establishing and clarifying the key elements in the definition of a software measure
- Measurement Approaches, which introduces the concepts necessary for reflecting measurement results
- Measurement, which establishes the terminology related to the act of measuring software

In Figure 3, Figure 4, Figure 5 and Figure 6 the UML diagrams of the different sub-ontologies are presented, while in Table **1**, **Table 2**, **Table 3** and Table 4 the concepts defined in these sub-ontologies are shown.

Table 1: Concepts of the 'Characterization and Objectives' Sub-Ontology

Term	**Definition**
Information Need	Insight necessary to manage objectives, goals, risks, and problems
Measurable Concept	Abstract relationship between attributes of entities and information needs
Entity	Object that is to be characterized by measuring its attributes
Entity Class	The collection of all entities that satisfy a given predicate
Attribute	A measurable physical or abstract property of an entity, that is shared by all the entities of an entity class

Quality Model	The set of measurable concepts and the relationships between them which provide the basis for specifying quality requirements and evaluating the quality of the entities of a given entity class

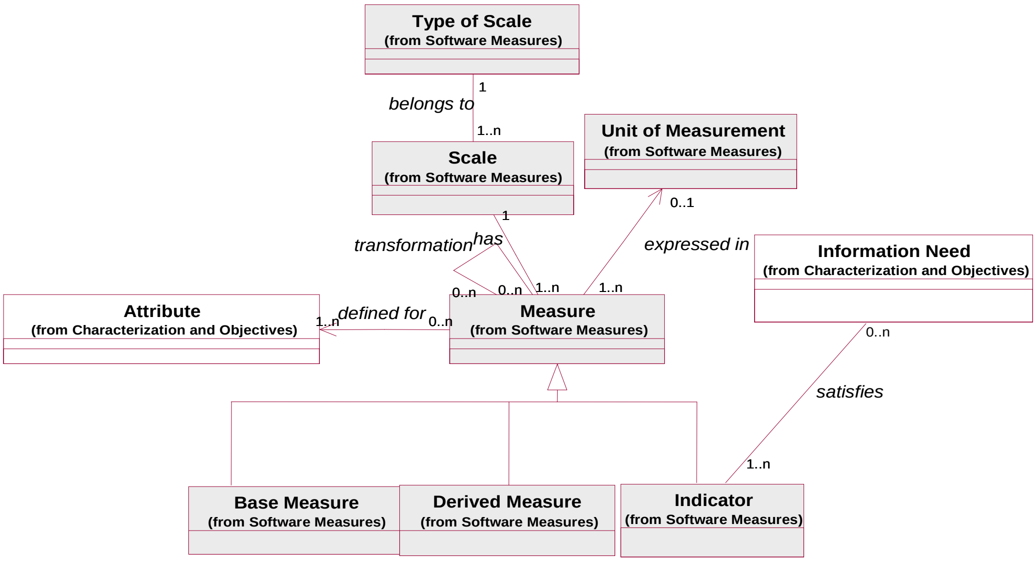


Figure 4: UML Diagram of the 'Software Measures' Sub-Ontology

Table 2: Concepts of the 'Software Measures' Sub-Ontology

Term	Definition
Measure	The defined measurement approach and the measurement scale. (A measurement approach is either a measurement method, a measurement function or an analysis model)
Scale	A set of values with defined properties
Type of Scale	The nature of the relationship between values on the scale
Unit of Measurement	Particular quantity, defined and adopted by convention, with which other quantities of the same kind are compared in order to express their magnitude relative to that quantity
Base Measure	A measure of an attribute that does not depend upon any other measure, and whose measurement approach is a measurement method
Derived Measure	A measure that is derived from other base or derived measures, using a measurement function as measurement approach
Indicator	A measure that is derived from other measures using an analysis model as measurement approach

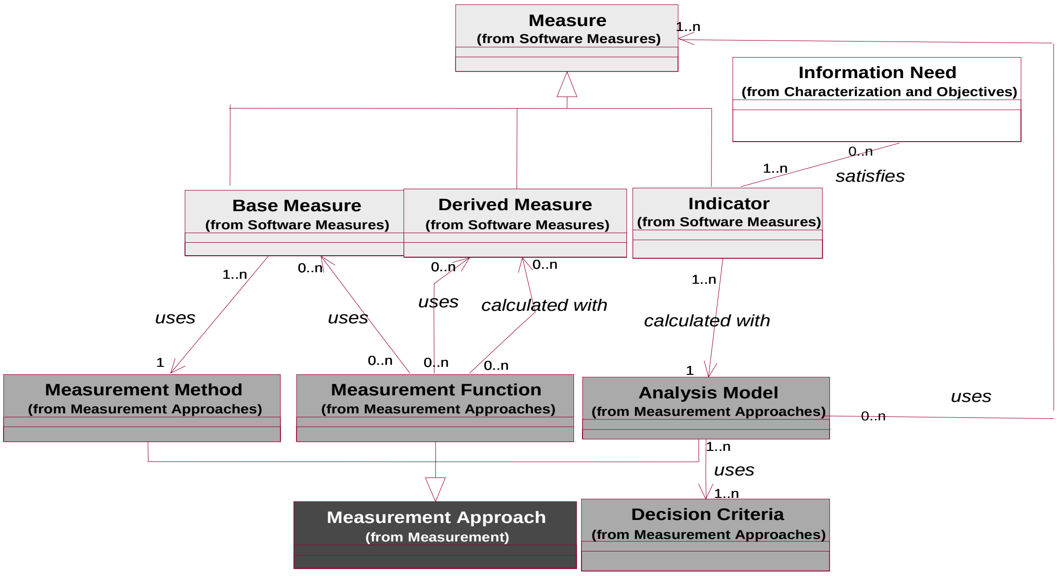


Figure 5: UML Diagram of the 'Measurement Approaches' Sub-Ontology

Table 3: Concepts of the 'Measurement Approaches' Sub-Ontology

Term	Definition
Measurement Method	Logical sequence of operations, described generically, used in quantifying an attribute with respect to a specified scale. (A measurement method is the measurement approach that defines a base measure)
Measurement Function	An algorithm or calculation performed to combine two or more base or derived measures. (A measurement function is the measurement approach that defines a derived measure)
Analysis Model	Algorithm or calculation combining one or more measures with associated decision criteria. (An analysis model is the measurement approach that defines an indicator)
Decision Criteria	Thresholds, targets, or patterns used to determine the need for action or further investigation, or to describe the level of confidence in a given result

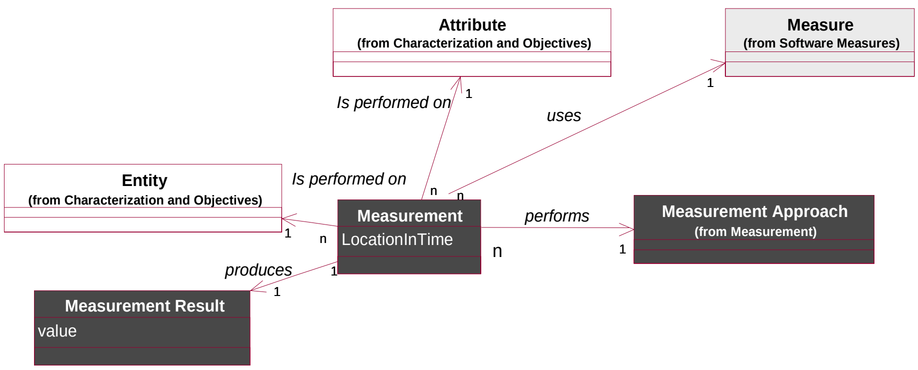


Figure 6: UML Diagram of the 'Measurement' Sub-Ontology

Table 4: Concepts of the 'Measurement' Sub-Ontology

Term	Definition
Measurement Approach	Sequence of operations aimed at determining the value of a measurement result. (A measurement approach is either a measurement method, a measurement function or an analysis model)
Measurement	A set of operations having the object of determining the value of a measurement result, for a given attribute of an entity, using a measurement approach
Measurement Result	The number or category assigned to an attribute of an entity by making a measurement

This ontology is the basis on which a namesake (and structure-equivalent) meta-model has been defined [15]. Next, we present how we have adapted such meta-model to meet our detected needs.

2.1. The WE Software Measurement Meta-Model (SMM)

The SMM presented in [15] is a mirror of the underlying Software Measurement Ontology, and may be instantiated to define in a systematic and non-ambiguous way a measurement model that includes all the necessary concepts for the operationalization of a given quality model. The main advantage of using meta-models instead of ontologies in the context of a software development process stems in their prescriptive rather than descriptive nature, what permits the designer to make assumptions on the measurement models that are not possible with ontologies. Also, meta-models can be tailored for specific contexts.

Although the SMM does not guide the selection of the concrete measurable concepts and attributes that must be included in a certain measurement model (this needs to be done by extensive research on existing models, theories, experience and/or empirically proven assumptions, whose result is a given quality model), it provides (by means of the underlying ontology) a clear definition of such concepts. As an example, measurable concepts are defined as abstract concepts that relate to an information need and that cannot be directly measured. On the contrary, attributes are concrete concepts that relate to a given entity class and that have a set of concrete (base or derived) measures associated. Also, the SMM provides the grounds to define the context of the measurement model we want to define by means of two concepts (see Figure 3):

Information Need (what is the purpose of each measurable concept in the quality model) and Entity Class (artefact on which measures contained in the quality model are to be applied). Last but not least, the SMM establishes that in order for a quality model to be complete we need not only information needs, measurable concepts, entity classes and attributes, but also measures with a scale, a type of scale and a unit, indicators that combine measures to give a combined value, decision criteria to translate such value into an answer to an information need, and so on.

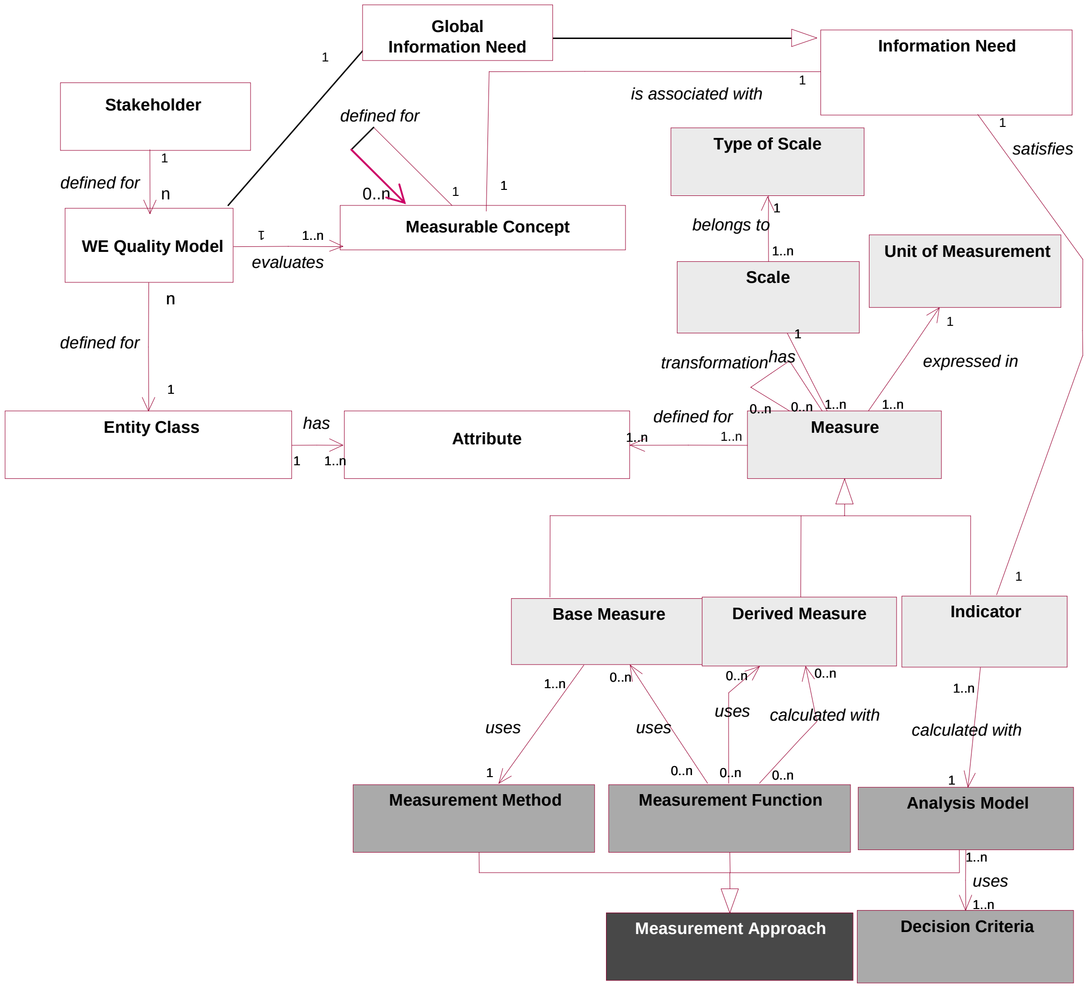


Figure 7. WE-Measurement Meta-model

However, the fact that the SMM presented in [15] is aimed at being used for such diverging purposes such as storing measuring results, managing quality models or managing divergent quality criteria among families of applications, makes it too complex. Given the fact that we aim at simplifying as much as possible the definition of WE measurement models, we have simplified and adapted the SMM to make their instantiation more intuitive for Web designers. Such simplification is presented in Figure 7. The concepts and relationships included in this meta-model force a certain structure similarity to any quality model defined based on them. In the case of WE, this similarity not only facilitates the understanding and discussion of new WE quality models for practitioners familiar with other WE quality models, but it also helps to the merging process of all these models in a global WE measurement model.

The construction of the WE-SMM presented in Figure 7 has implied the following actions over the original SMM:

- We have limited the risk for inconsistencies in the measurement model by eliminating SMM redundant relationships: the relationship *measurable concept-attribute* and the relationship *analysis model-measure*. The reason why we have chosen to eliminate the relationship measurable concept-attribute has to do with the way in which we propose to empirically construct the WE Quality Model that sustains this measurement model. In this empirical construction process, attributes are defined as associated to Entity Classes, and measures are defined for each attribute. No hypothesis is made *a priori* on which attribute serves to measure which measurable concept. Then, it is the gathered empirical evidence what serves to establish the measures that are in fact influencing each measurable concept, and these measures (which make up the indicator for the information need related to that measurable concept) indirectly indicate which attributes are involved with that measurable concept Section 6 will provide further detail on this empirical construction process.
- We have limited the set of valid Entity Classes to the outgoing artefacts of the WE development process. In this way, measurable concepts that are to be measured on different WE artefacts are forced to belong to different quality models.
- We have introduced a global Information Need that is connected with the WE-quality model as a whole to justify its definition. For the structure of this Global Information Need we propose to use the GQM template for goal definition [5].
- In order to keep the quality model simple, we have limited the connection of each Measurable Concept to a single Information Need.
- For the same reason, we have established that each Information Need be satisfied by a single Indicator, implying that the Measurable Concept connected with the Information Need is also (transitively) associated with that indicator.

- In order to assure that every Attribute is measurable, every attribute defined in a WE quality model should be associated with at least one Measure that is devoted to measuring such Attribute. This restriction makes sure that the evaluation model is operationally defined by means of Measures, that is, no reliant on subjective interpretations of concepts [38].
- In order to establish a single way of calculating Indicators, we propose that every Measure is associated with a single Analysis Model
- In order to further contextualize the WE quality model and help to keep the focus, we have added a 'Stakeholder' element to the original SMM.
- Finally, we have omitted from the WE-SMM the Measurement package, due to the fact that their elements do not contribute to the definition of quality models but rather to the results of their operationalization.

Additionally, and although not directly reflected in the WE-SMM, in order to control the quality model complexity we recommend the limitation of the hierarchy depth of Measurable Concepts to two levels of detail. Also, following the ISO/IEC 9126 example, these two levels should be characterized by familiar labels and concise definitions. Similarly, attributes associated with Entity Classes should also be familiar and provide concise definitions. Finally, in order to facilitate a hypothetical merging of measurement models at different levels of abstraction into a general, well-structured WE global measurement model, we recommend that attributes for the different models have unique names in the context of the WE field.

From these refinements, the inclusion of stakeholders is, from our point of view, especially relevant. Stakeholders are usually not explicitly identified in existing quality models. However, as stated in [11], they are important in any quality model, as different Stakeholders will generally be interested in different Measurable Concepts. Moody defined four stakeholders for ER models: Business User, External Analyst, Information Architect and Database Designer. The fact that we are interested in assessing the quality of the final Web application (and not of a model *per se*), together with the characteristics of the WE process, has driven use to the definition of a different set of stakeholders, which make up the set of allowed instantiations for the Stakeholder meta-model concept, namely:

- Analysts/Designers: they are the link between customers and developers and are focused on the intermediate products, that is, the WE artefacts (models). According to [45], their concern is the Specification and Usage perspectives of the WE models. Analyst/Designers have a ‘compliance to specifications’ perspective of quality.

- Developers/Maintainers: they are in charge of implementing/maintaining the system (implementation and code level). Therefore they are also focused on intermediate artifacts (models), namely on artifacts that convey the Implementation perspective [45]. They share with designers a ‘compliance to specifications’ perspective of quality.

- Customers: they have a ‘meet customer needs’ perspective of quality. According to the ISO/IEC SQuaRE [47], they can be divided in two subgroups:

    1. Acquirers: they are interested in cost, time and functionality. If we consider that models (intermediate products in the WE development process) may be used as communication artefacts, then according to [45] the Completeness and Understandability of the models may influence the acquirer’s perception of quality in use. Also, acquirers may be interested, regarding the final Web application, in the Efficiency, Effectiveness and Security of such application, as long as all these factors have an impact on productivity, and therefore on cost and time gains.

    2. End Users: they are the ones that will eventually interact with the application. For them, efficiency, effectiveness, security and satisfaction (that is, quality in use) of the deployed application are the only factors that matter.

This classification is slightly different from the perspective presented in [41] where visitors, developers and managers are distinguished. It also differs from that of Dromey [14] in that it adds the analyst/designer perspective of quality.

The concepts and relationships included in this meta-model, together with the additional recommendations, force a certain structure similarity among any quality model defined based on it, what in turn facilitates the understanding and discussion of WE quality models among both researchers and practitioners.

In order to illustrate the use of this meta-model next we are presenting an instantiation example that represents a quality model devoted to evaluate the navigability of a Web application.

2.2. A WE-measurement model example: the Navigability WE-measurement model

The WE-SMM defined above can be easily instantiated to define a complete and structurally sound measurement model over any WE intermediate artefact with the ultimate objective of assuring certain characteristics that may contribute to improving the quality in use of the application. The aim of the WE discipline should be to provide at least one WE measurement model that reflected a general quality model for each pair stakeholder/WE abstraction level. However, in some contexts it may be convenient to fine-tune the elements of such generic WE quality model to adapt it to a given application in a given domain. For instance, more specific decision criteria that better reflect the domain knowledge could be defined, or certain attributes/measures that are present in the quality model may be dismissed to even further simplify the measurement process.

At this point we would like to emphasize the differences between a quality model and a measurement model. A quality model does not need to follow any specific ontology (not even the SMO), nor have any specific requirement on how complete it has to be. Therefore, any quality model proposal found in literature is still considered a valid quality model in our approach. A quality model that deals with any WE artefact (including code) is considered to be a WE-quality model, even if it has not been defined as such in the source. As opposite, we define a WE measurement model as a WE-SMM instantiation. Such instantiation must be based on an underlying WE quality model, but does not need to be completely faithful to it (it may involved certain degree of tailoring). For example, it may happen that the quality model lacks some necessary elements (as defined in the WE-SMM) such as certain indicators or measures. Also, it may happen that it is too complex and therefore only a subset of it is used to instantiate the WE-SMM. This notwithstanding, some elements of the quality model could violate the WE-SMM restrictions, and therefore the WE-measurement model should also solve them accordingly. Finally, it could happen that the particular domain and/or application we are dealing with requires specific quality restrictions, not generally applicable. The necessary tailoring of the quality model to fit those special needs would happen during the WE-SMM instantiation. We could in fact regard the WE-measurement model as an operationalization of an underlying quality model due to the fact that (1) it introduces certain restrictions/refinements on the quality model (e.g. to provide measures for every attribute) and (2) it permits to express the quality model it in a machine-readable format, which in turn opens the path to applying automation techniques.

As a proof of concept, let's present a hypothetical *WE-Measurement Model (WE-MM)* aimed at assessing the **Navigability** of the final application. This Navigability WE-MM intends to reflect the viewpoint of the end-user of the application, that is, the **Stakeholder** involved is the End-User. Therefore, only model qualities that are bound to contribute to increasing the end-user quality in use of the application should be included in this instantiation, which aims at assessing the navigability problems that may arise due to a low-quality definition of navigational paths and navigational nodes. For the sake of simplicity, let's suppose that our instantiation is completely faithful to an existing WE-Navigability quality model. Therefore, from now on, when we refer to elements of the WE-MM the same descriptions will apply to that underlying quality model (that in reality still needs to be empirically validated [7]).

Navigation in WE is captured by means of a Navigational Model, and thus the **Entity Class** will be the Navigational Model belonging to any WE methodology. Navigational Models have two main purposes in the WE development process. On one hand they define the set of abstract pages, that is, the basic information nodes that make up the application. On the other hand, they provide the navigation paths and the navigation facilitators (menus, indexes, guided tours and so on) to improve the user experience. Those two purposes can be reflected in two different *Attributes*: Navigation Node Complexity and Navigation Path Complexity.

The Navigability WE-MM is associated to a **Global Information Need** To Know how good Navigability is. Recall that the description of such global information need must follow the GQM template, and therefore could be defined as follows: *analyzing the WE Navigational Model for the purpose of evaluating it with respect to the navigability of the final application from the viewpoint of the end-user of the application in the context of a testing environment.*

Term	Instantiation for the Understandability Measurable Concept
Stakeholder	End-User
Global Information Need	To know how good navigability is
Information Need	To know how good understandability is
Measurable Concept	Understandability
Entity Class	Navigation model
Attribute	(1) Navigation Node Complexity – (2) Navigation Path Complexity
WE-Quality Model	Navigability WE-Measurement Model
Base Measure	(1) Number of attributes (NA) - (2) Number of Navigational Links (NNL)
Scale	Natural Number
Type of Scale	Ratio
Measurement Method	(1) Count the number of attributes of the model– (2) Count the number of links of the model
Indicator	UND_IND (NA, NNL)
Scale	Acceptable-NonAcceptable
Type of Scale	Ordinal
Analysis Model	f(UND_IND)=NA+NNL
Decision Criteria	If f(UND_IND) <50 then Acceptable else NonAcceptable

Table 5 The Navigability WE-QM as a WE-Measurement Meta-model instantiation example

This Navigability WE-MM contains a set of **Measurable Concepts**. If we consider Navigability as 'Usability of the navigation', we can assume that the main characteristics included for usability in the ISO 9126-1 quality model apply. These characteristics are Understandability, Learnability, Operability, Attractiveness and Compliance. We agree with [2] in that only the first three Measurable Concepts (understandability, learnability, operability) are related with the user performance and can be therefore quantified using objective measures, some of which can be taken over navigational models. Attractiveness is not relevant at this stage of development, where final users are not yet present. Last, as far as we know there are no widely accepted standards or conventions regarding the definition of navigation structures in WE navigational models, and therefore Compliance is not relevant either.

Each Measurable Concept must be related to an **Information Need**. The Information Need covered by Understandability in the context of the Navigability WE-MM is To Know how good Understandability is. The description associated with such concept could be 'the capability of the Web application navigational structure to enable the user to understand whether the application is suitable for her, and how it can be used for particular tasks under certain conditions of use'. Learnability and Operability can be defined similarly.

The partial instantiation of the WE-SMM that gathers all the concepts introduced so far is presented in Figure 8.

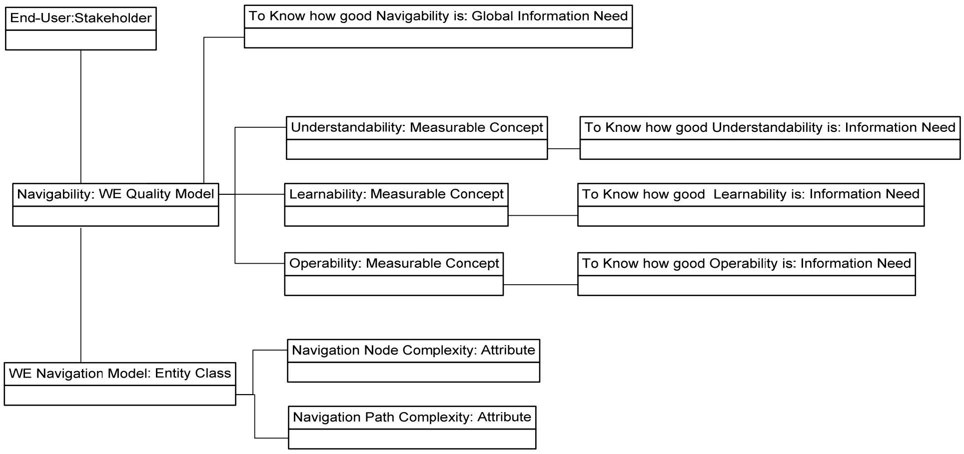


Figure 8 Partial instantiation of WE_Navigability model (Part 1)

According to the meta-model, each model attribute must be related to at least one **measure**. For the sake of the example, let's suppose that we have determined that only two measures are relevant for the evaluation purposes of this Navigability WE-Quality Model: the number of navigational links (NNL) and the number of attributes (NA).

The definition of the Number of Navigational Links (NNL) **Base Measure** includes the **Scale** Natural Number, the **Type of Scale** Ratio and the **Unit of Measurement** Links. This measure is associated with the Navigation Path Complexity Attribute. The **measurement method** is 'to count the number of links of the model'. Similarly, the definition of the Number of Attributes (NA) Base Measure is associated with the Scale Natural Number, the Type of Scale Ratio and the Unit of Measurement Attributes. This measure is related with the Navigation Node Complexity Attribute. The **measurement method** is 'to count the number of attributes of the model' .

Also, each information need requires at least one **Indicator**. Indicators can be regarded as special kinds of measures that are related to decision criteria via an **Analysis Model**.

As an example, let's define the Understandability Indicator UND_IND. Let's suppose that the Analysis Model associated to this indicator is a function that involves the two measures presented above: F(UND_IND)=NNL+NA.

Let's also assume that this indicator belongs to the Scale {Acceptable, Non Acceptable} and Type of Scale Ordinal (Acceptable is better than Non Acceptable).

Last, for the definition of the decision criteria let's assume that the Trellis number applies, and that models with less than 50 elements are understandable enough. This decision criteria is expressed in the meta-model instantiation as if f(UND_IND) <50 then Acceptable else NonAcceptable.

Figure 9 presents a WE-SMM instantiation that reflects all these new elements of the Navigability WE-Quality Model.

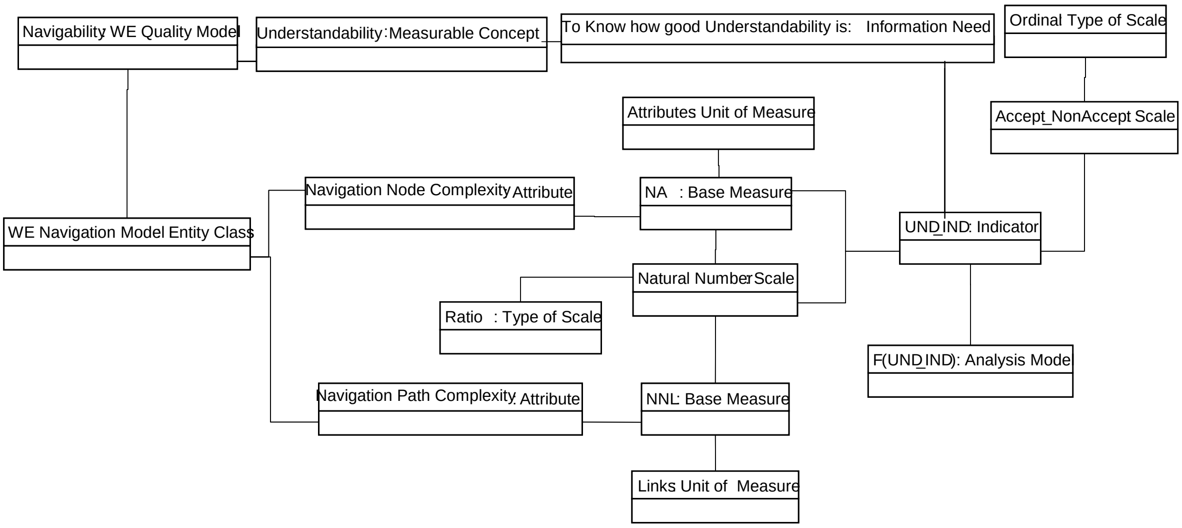


Figure 9 Partial view of Navigability WE-Measurement Model (Part 2)

2.3. Conclusion

In this section we have proposed the operationalization of WE-Quality Models as instantiations of a WE-SMM that is based on a Software Measurement Ontology. This ontology contributes to avoiding terminology inconsistencies (P1). The definition of a measurement meta-model based on such ontology turns the descriptive nature of ontologies into prescriptive, and therefore assures that a set of syntactic and semantic constraints are met by any quality model defined as an instantiation of such meta-model. One of such constraints is the set of elements that must be present in a syntactically correct WE-measurement model that reflects an underlying WE quality model, which partially solves incomplete definitions of quality models (P2). The focus on a given stakeholder and a given WE model as the application context of the measurement models facilitates the task of constructing consolidated, exhaustive yet specialized models. This fact in turn contributes to alleviating the traditional lack of focus quality models suffer from (P3). Last but not least, this way of representing WE Measurement Models by means of a meta-model instantiation is a machine-readable way, which simplifies the tool support (P9) and helps to preserve the development advantages provided by the (semi-)automatic nature of WE processes, as we will see in Section 6. However, for these measurement models to be of real use they should be based on quality models empirically validated. Next we present how we think such validation should be performed in the context of WE.

3. Empirical Validation of WE Quality Models

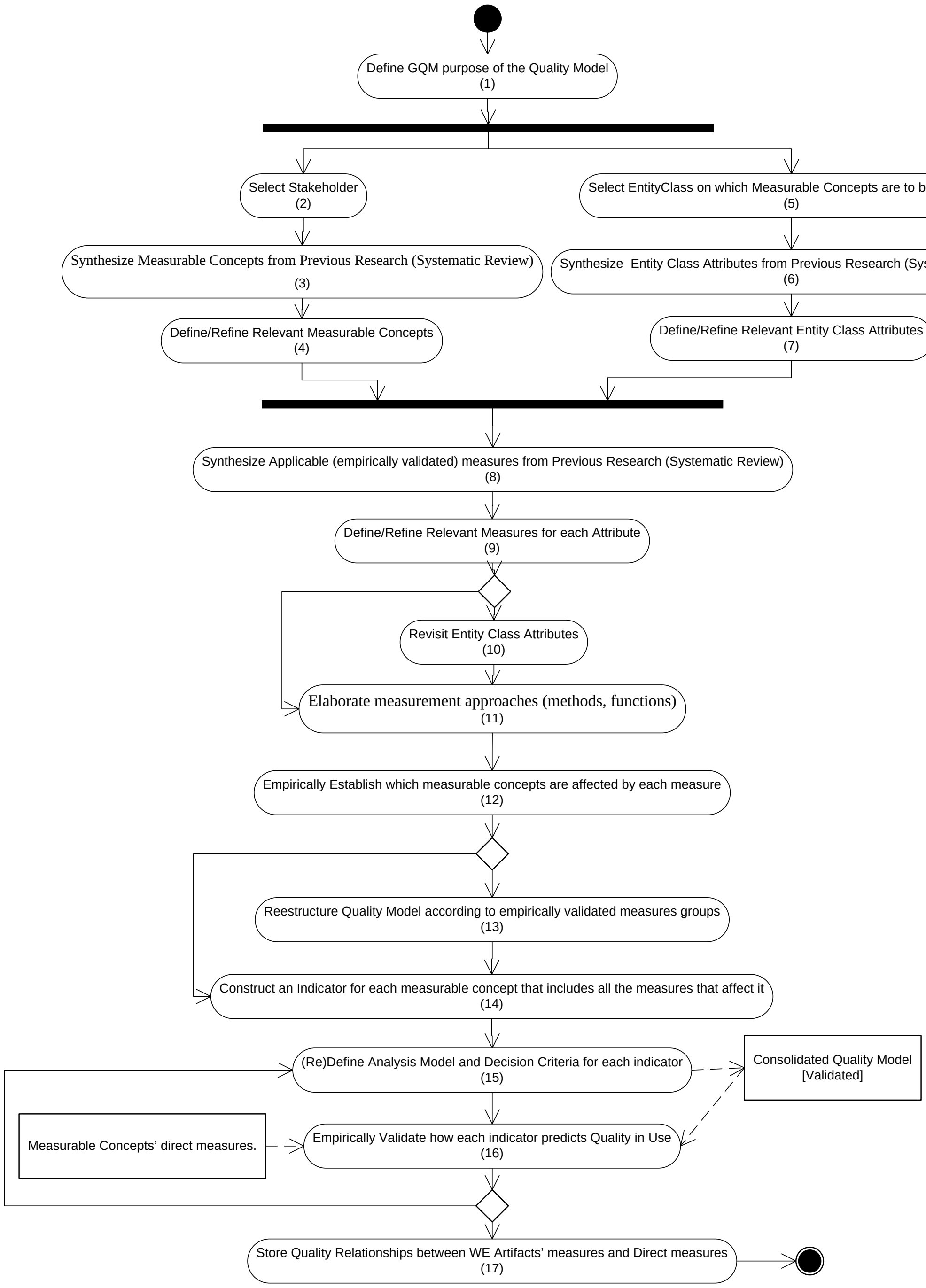


Figure 10. General picture of the proposed process to define empirically validated WE Quality Models

No matter how complete our WE measurement models may seem, if they are not based on empirically validated empirically validated quality models they may still be unable to prevent quality problems. In fact, research shows that the definition of quality models based on intuition or theory generate quality models that, at least, are far too complex.

Additionally, conventional wisdom, intuition, conjecture and proofs of concepts are not reliable sources of credible knowledge [36].

A methodologically rigorous empirical research is therefore needed to make sure that (1) operationalizing a certain quality model is in fact contributing to improve the quality of the final application and (2) that such quality model is in fact the simplest possible to guarantee the desired results. Regarding this last point, recent empirical research has already demonstrated that a few concepts, attributes and measures suffice to obtain significant gains in quality in a context of limited budget and time framework [37] which suggests the convenience of changing the widespread tendency of constructing exhaustive quality models. Of course, such validation should never be done by the user of a given WE methodology but provided by researchers, together with a means to tailor these models to particular needs (for example by means of a measurement model, as we are proposing in this technical report).

The process we are proposing for WE researchers to follow for defining/validating a WE quality model can be summarized as follows (see Figure 10):

1. Define a Global Information Need for the Quality Model following the GQM template
2. Define the point of view that is going to be considered when constructing the quality model (the stakeholder)
3. Perform a systematic review to gather the initial set of measurable concepts that may affect the quality perception of such stakeholder. Try to keep faithful to the ISO/IEC 9126. The main advantage of such compliance is that, being the ISO an agreement between researchers and practitioners, it provides quality frameworks that are based on principles generally accepted, and therefore augment their opportunities of being used in industry.
4. Refine such set of measurable concepts (addition/deletion/modification) with reasoned arguments. The set of low-level measurable concepts identified should be sufficient, necessary and independent [37].
5. Select the Entity Class on which the WE Quality Model is to be applied
6. Gather attributes defined for the considered Entity Class (WE artefact) in literature.

7. Refine such set of attributes (addition/deletion/modification) with reasoned arguments
8. Perform a systematic review of available (if possible validated) measures that may be used to measure the different attributes of the WE artefact under consideration.
9. Refine such set of (validated) measures with reasoned arguments. The number of measures should be kept as low as possible without losing prediction power, in order to maintain the quality model as simple as possible.
10. If necessary, revisit the set of attributes for the Entity Class to remove the ones for which no measures have been identified. Also, it may be necessary to add new attributes to cover measures that cannot be related to any of the existing ones.
11. Elaborate measurement approaches (functions, methods) for each measure.
12. Find the relationships between measurable concepts and measure clusters. This can be done empirically with the aid of statistical approaches such as factor analysis or a regression analysis of some kind. Also, Bayesian networks could be applied, or expert opinions gathered. All these methods are not exclusive.
13. Revise the quality model and check whether the proposed decomposition is conform to the factor analysis For example, if measurable concepts with no measures associated are encountered, such measurable concepts should be removed from the model.
14. Construct an indicator for each measurable concept based on the measures that have empirically proven relevant for such concept. This indicator represents an answer to the information need associated with such measurable concept.
15. Define an Analysis Model and decision criteria for each indicator. Again, here existing literature may be of use. Here we must be careful: sometimes improving certain measures implies hampering others. We should empirically test which threshold values are acceptable and how they can be counterbalanced.
16. Find direct measures for the stakeholder quality perception of the application directly generated from the Entity Class under consideration (without introducing any further refinement at any other level of abstraction) and correlate such measures with the indicators included as part of the WE quality model. Here with the term ‘direct’ we mean measures that are not taken over the Entity Class associated with the quality model. Direct measures will often be

perceptual measures, to be assessed by subjects representative of the quality model's stakeholder. For example, response time is an important measure for end-users that is required to evaluate the efficiency of the software, but it cannot be measured during development. However, during navigation design it is possible to obtain the path length. This measure could therefore be part of an indicator that provided rough estimates of efficiency from the point of view of such end-user under certain conditions.

17. If the quality model shows itself as a good predictor of the quality in use, store the relationships encountered in a tabular form and finish. Otherwise, refine the quality model's indicators and decision criteria and go back to step 15.

When, after verifying the indicators with several similar applications, such indicators show enough liability, the corresponding internal indicators can be marked as 'reliable' and from then on be directly used each time we face a similar application to predict external results. We must be careful however; the accuracy of the indicators to predict quality in use may still be affected by the intended conditions of use.

Also, we must take into account that our validated quality model still may need to be tailored during its conversion into a measurement model to fit the specific needs of specific Web applications. During this tailoring process, the quality model provides the set of available concepts and measures, and the designer ideally should only need to (1) dismiss those elements that are not relevant for her specific needs and (2) adjust the decision criteria, which may vary from application family to application family.

3.1. Risks of the WE quality model construction process

We are aware that the construction of quality models is far from easy. This is clear if we analyze the myriad of quality models, often contradictory, that we find in literature. Although our proposal aims at empirically validating each decision taken, such validation is not without price: the process from an initial proposal to a validated one can be cumbersome. However, even if only parts of the model are validated, we believe that this is better than nothing. Once sufficient amount of WE experiments have been carried out, we can turn up to meta-analysis techniques to gather results.

Also, we are conscious that, given the fact that best known WE methodologies come from universities, and that they have no significant impact in industry, toy problems in toy situations (i.e. the use of artificial problems in artificial situations), are likely to be the focus of experimentation during the first stages [36]. Also, we are aware that students are not real practitioners are likely to be used as subjects. However, we claim that this is better than not conducting any evaluations at all. The results can therefore be of benefit to explore an initial idea or research design. Once practitioners widely adopt WE practices, it will be possible to foresee the replication of experiments under real conditions of use, as we already showed in Figure 2. Also, if we are to compare the effectiveness of WE practices with respect to creative approaches, we must be sure that subjects are equally familiarized with both practices, so that learning effect does not bias the results.

3.2. Conclusion

In this section we have proposed a WE quality model definition process that is driven by a research methodology that assures that the resulting models are systematic and empirically tested. In order to overcome quality models too cumbersome to apply (P4), sufficiency, necessity and independence have been named as criteria to be considered during the definition of measurable concepts (quality factors). These criteria assure that the amount of concepts contained in the quality model is minimal yet complete. Also, in order to simplify the operationalization of the concept measurement, the measure set should be minimal. Our research methodology furthermore includes tasks to determine the dependencies among the different elements in and among quality models in order to facilitate their use early values as predictors (early indicators) of actual quality in use of the application.

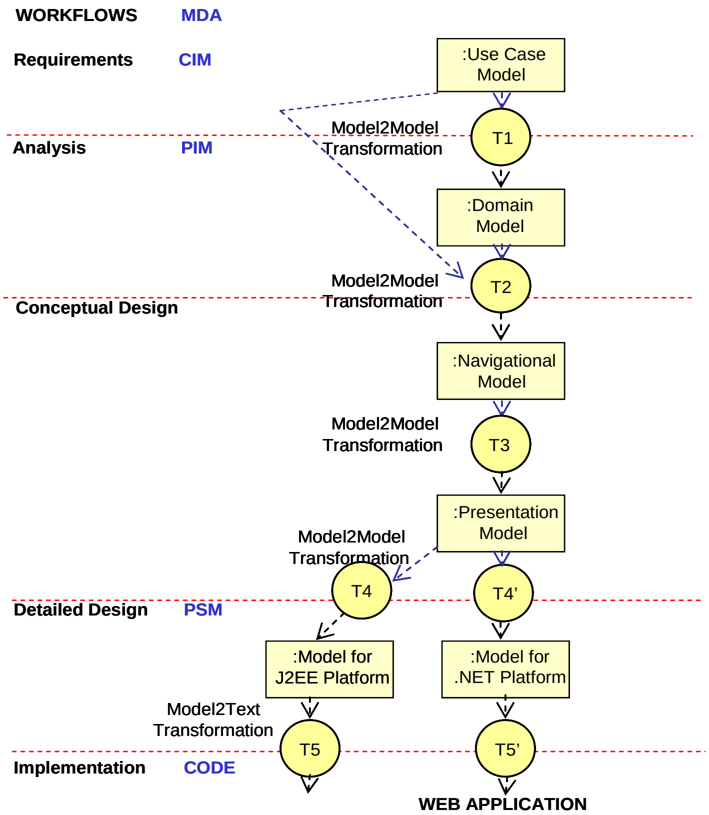
However, providing a measurement model and an underlying quality model, even if it has been empirically assessed, is not enough. We need to guide the practitioner on how to use such measurement model to actually assure a good quality of the final product, that is, a quality evaluation process. Our proposal for such quality evaluation process in the context of WE is presented next.

4. Integration of a WE Quality Evaluation Process with the WE Development Process

As we presented above, a Quality Model (and thus also a measurement model) must be accompanied by a Quality Evaluation Process to be of real use for practitioners. In order to facilitate its adoption, this quality evaluation process should be integrated with current WE practices (P6). Although there is no agreement on a common Web development process, most methodologies share a set of artefacts and activities that may be regarded as a simplified WE process. Figure 11 presents such simplified version together with its related artefacts.

This process, based on the Model Driven Engineering paradigm (MDE) [27], departs from a general business model and includes (1) a (manually performed) functional requirements workflow, whose outgoing artefact is a use case model, (2) an analysis workflow, whose output is a domain model (usually an ER diagram or a UML class diagram), (3) a conceptual design workflow, whose outputs are a navigation and a presentation model (expressed by means of UML profiles or proprietary notations), (4) a detailed design workflow that introduces platform and technology specific features (typically J2EE and .NET) and (5) an implementation workflow, which results in a Web application that is ready to be deployed. Variants of this process model exist, usually to include additional Platform Independent Models (PIM's) and/or Platform Specific Models (PSM's) (architectural models, business process models, different languages and/or platforms, etc.) that further enrich the application specification. Additionally, WE methodologies promote the use of automatic and/or (semi-)automatic transformations among most of these artefacts (represented as stereotyped activities in Figure 11) that, based on the underlying meta-models, streamline the process and guarantee traceability among and between concepts.

The use of a WE process with (semi-)automatic transformations prevents some development problems such as inconsistencies among models, lack of traceability, lack of technical soundness, etc. However, this (semi-)automatic nature of the WE process also may cause the propagation of quality flaws through levels of abstraction. Otherwise stated, quality problems that are now only detected at implementation time may have been introduced not during the implementation phase but at any previous stage of development. As an example, a low cohesion of the Web application requirements [45] may cause that during the construction of the Navigation Model the interface structure is defined in an improper manner. The reason is that the Requirements Model (usually a UML Use Case Diagram) is used in most WE approaches to decide how to perform such division [10]. Even more evident, missing requirements will cause dismissed quality due to the fact that the user perceives a lack of functionality.



This notwithstanding, the fact that during the construction of every WE artefact the system is enriched and refined with respect to previous levels of abstraction causes that the end-user perceived level of quality may be also hampered by the introduction of new quality flaws during such enrichment. As an example, even if the requirements model presents a high quality level, the new information introduced at the domain level may introduce new kinds of quality problems. Imagine for example that we have forgotten to include certain domain relationships (which are present in the end-user's mind) in the domain model. This domain model (see Figure 11) is the basis on which the navigation model, which is in charge of defining the user paths through the application, is constructed. Therefore, missing relationships in the domain model will be propagated to the navigation model and cause missing relationships among concepts in the final application interface. If the user looks for these relationships while interacting with the application, this omission is likely to diminish her perceived degree of quality.

Going one step further, the refinements performed at navigation level may cause new kinds of quality problems to appear. For instance, even in face of high quality requirements and conceptual models, we may design a set of tangled navigation paths that mislead the user in her goals pursuit, and therefore diminish the end-user perceived level of quality. Additionally, a poorly designed presentation model (e.g. a model that does not include position signals, where widgets are poorly chosen and so on) may also induce other kinds of quality problems for the end-user, who may feel that the interface appearance does not fit her needs. Last but not least, usability problems can be introduced on the running code itself, by means of implementation decisions that hamper load times, performance, security and so on.

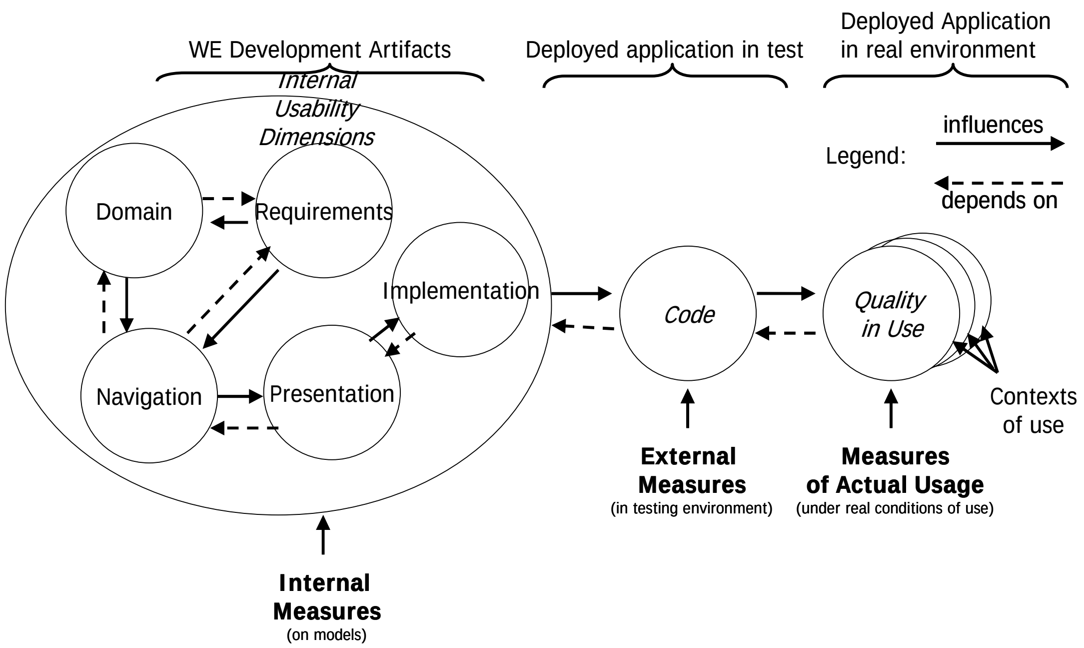


Figure 12 Quality in the WE lifecycle (adapted from ISO 9126)

The six levels of refinement resented in Figure 11 imply therefore six different purposes of evaluation that must be taken into account when defining the quality models and its related operationalizations (measurement models):

- Use Case Model (Requirements Coverage)
- Domain Model (Representational Faithfulness)
- Navigation Model (Navigability)

- Presentation Model (Attractiveness)
- Implementation Model (Implementation Decision Quality)
- Executable Code (Quality as Tested under conditions that emulate as closely as possible the expected conditions of use)

From these six types of WE products, the first five can be regarded as 'internal products' in the sense that they refer to models of the product, and not the product itself, while the deployable Web application is an external product (the product that actually reaches the market). A graphical representation of the products, together with their hypothetical quality inter-relationships, is presented in Figure 12. Such relationships are based on (1) the ISO/IEC assumption that quality at one level of abstraction may be used to predict quality and lower levels of abstraction and (2) the already mentioned underlying traceability of concepts among the different WE models (see Figure 11).
Namely, in Figure 12 we can graphically observe how the internal quality dimensions may affect an external quality dimension, that is, the quality of the final application (code) as perceived under testing conditions. Finally, such external quality may influence the actual quality of the application in real contexts of use.

As we mentioned above, our proposal includes the encapsulation of each pair purpose of evaluation-product type in an independent WE quality model that is translated into one or more WE-measurement models (each one reflecting one possible tailoring of the quality model). Additionally recall that, in order to preserve the MDE paradigm implicit in the WE process that we presented in Figure 11, machine-readable measurement models (greyed in Figure 13) must be derived from underlying WE-quality models (dotted in Figure 13). Last, it is important to note that, in order to assure the reusability of our framework, for the definition of WE-quality models (dotted in Figure 13) it would be necessary to reach a consensus and identify a set of common attributes that characterize any of the WE models proposed by any of the best known WE methodologies, and centre WE-Quality Models on such common concepts. We do claim that such common set of concepts exist at each level of abstraction, as the recent MDWEnet initiative[2] backs. Only such attributes, together with a general definition of measures, independent from particular notations, should be included in WE-quality

[2] [2] Interested readers can follow the lines of work and the state of evolution of this project by contacting the MDWEnet project members (http://www.pst.informatik.uni-muenchen.de/~zhangg/cgi-bin/mdwenet/wiki.cg)

models in order to make them reusable among WE methodologies. How such reuse can be achieved was presented in [9] and will be therefore only briefly revisited in Section 6.

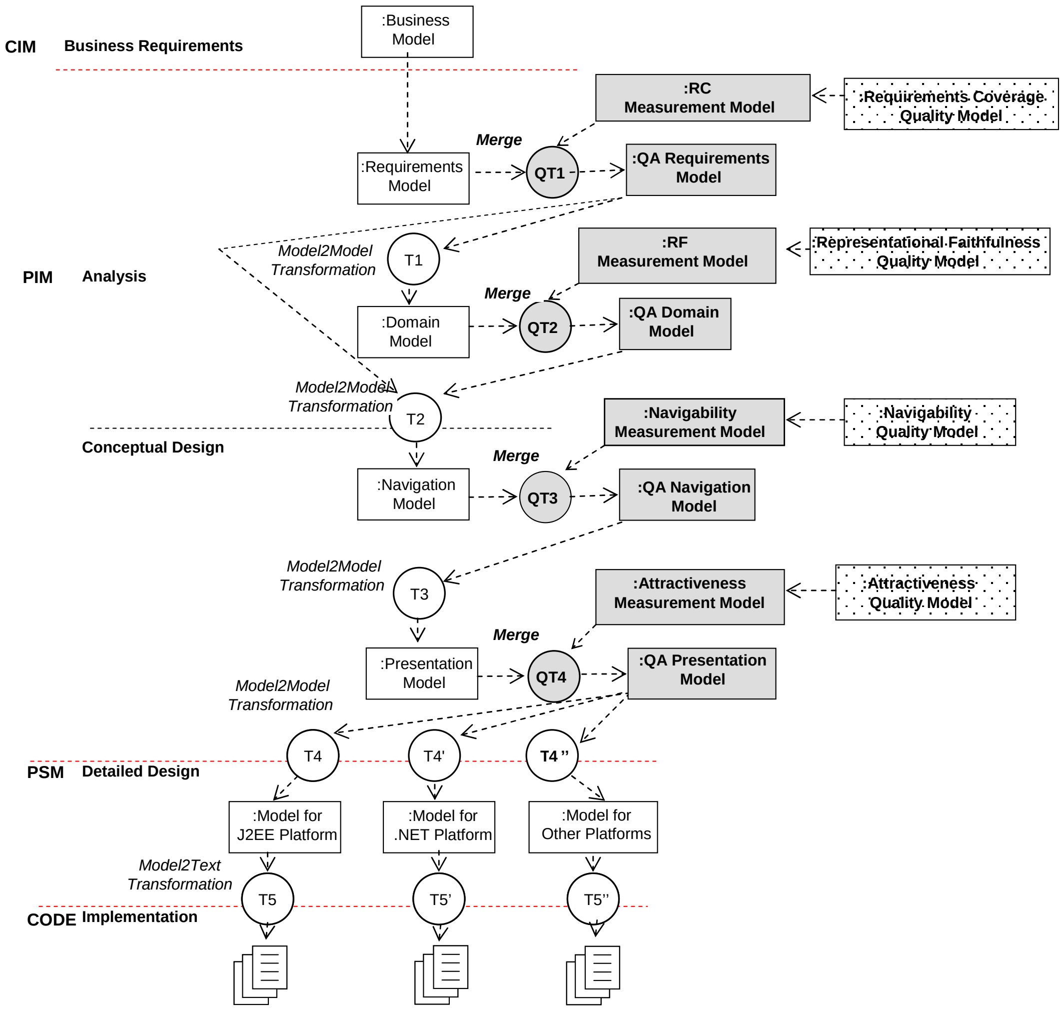


Figure 13. Quality-aware WE Development Process.

4.1. Conclusion

In this section we have presented a quality-aware WE development process (see Figure 13) that merges the traditional WE development process with a quality evaluation process. The resulting process includes a set of WE measurement models that are complete, integrated in current WE practices and that prevent rather than detect errors as soon as possible in the development lifecycle, and it is therefore a sound response to the P6 problem presented in Section 1.3. Also remember that the fact that WE-measurement models are based on quality models that are centred on a given stakeholder and level of abstraction makes such models more concise. We agree with [37] in that the construction of concise quality models that are integrated in a quality evaluation process are of prime importance to focus the quality evaluation task and carry out a comprehensive quality analysis in a very limited timeframe. Also, in this section we have presented a set of ISO based relationships among the quality of intermediate artefacts. Such relationships justify the necessity to assure that the outgoing artefact of each workflow has the required level of quality before going on to the next step of development.

One question that may have arisen in the mind of the reader at this point is why we have not simply proposed to use the ISO/IEC 14598 to define the Web Quality Evaluation Process. The reason for this fact is presented next.

5. An ISO/IEC 14598-compliant TQM-aware WE Development Process

Even if it is true that the ISO set of standards accompanies the definition of quality models (ISO/IEC 9126) with a software evaluation process (defined in the ISO/IEC 14598), it is a well known fact that both standards are not sufficient to direct the practitioner in the quality evaluation process [47]. One reason for this fact may be that ISO/IEC 14598 was finished before the last version of the ISO/IEC 9126, and while it provides generic linkages between the high-level concepts of the ISO/IEC 9126 quality instruments (characteristics, subcharacteristics and measures), the evaluation process is not yet specified in the format of specific prescriptive quality engineering practices. In particular, the current versions of these ISO/IEC standards do not provide a clear mapping between the quality engineering instruments already developed and the various phases of the WE development process [47] (see Figure 14)

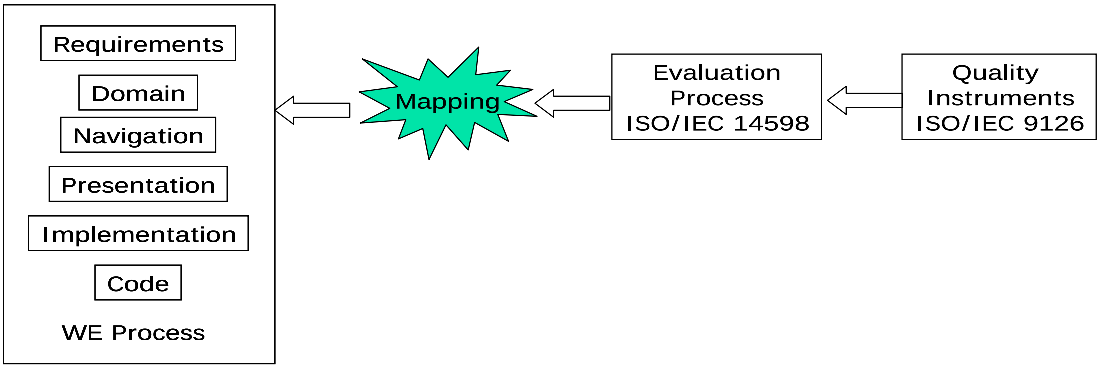


Figure 14. Relationship between ISO/IEC 9126, ISO/IEC 14598, and mapping gap between standards and the WE process (adapted from SQuAre)

Thus the first benefit of using the steps defined in the previous section to perform a Web Quality Evaluation Process such as the one defined in Figure 13 is that it covers such mapping, due to the fact that it relates, through the instantiation of the WE-SMM, specific quality models to specific WE artefacts, and provides an automated way to perform the measurement process on each of the artefacts. Additionally, the merging process we have proposed guarantees that each problem is detected and solved as soon as possible in the development lifecycle, what, as we have already outlined, diminishes costs and time to market of high-quality web applications.

However, while performing such mapping we still were interested in fulfilling the ISO/IEC 14598 requirements, as they reflect an agreement between researchers and practitioners, and therefore sticking to them may simplify the process of convincing practitioners. The good news is that it is possible to superpose ISO/IEC 14598 over the Web Quality Evaluation Process that we have proposed so far. For such superposing, the ISO/IEC 14598-3 (development) is of special interest. ISO/IEC 14598 poses two main requirements for compliance. On one hand, the quality evaluation must be based on a quality model. This end is fulfilled by our approach, as we already saw in Section 2. On the other hand, ISO/IEC 14598 demands that the evaluation process follows the steps presented in Figure 15.

As we saw in the previous section, the WE field restricts both the purposes of evaluation (according to stakeholders) and the set of artefacts involved (according to the workflow we are involved in). By choosing the WE quality model that is going to serve as a basis to instantiate a given WE-measurement model, the designer is in fact already covering some of the ISO 14598 activities (namely, *establish purpose of evaluation* and *identify types of products*, see Figure 15, elements in yellow). The evaluation purpose when operationalizating quality models and applying them to each one of the intermediate WE artefacts is twofold: (1) deciding on the completion of the process and when to send products to the next process (when the internal evaluation is satisfactory) and (2) using that internal evaluation to predict or estimate end product quality. Nonetheless, the purpose of the evaluation once the Web application has already been implemented is also twofold: (1) deciding on the acceptance of the product and (2) deciding when to release the product. The type of product is the one that was associated to the chosen quality model.

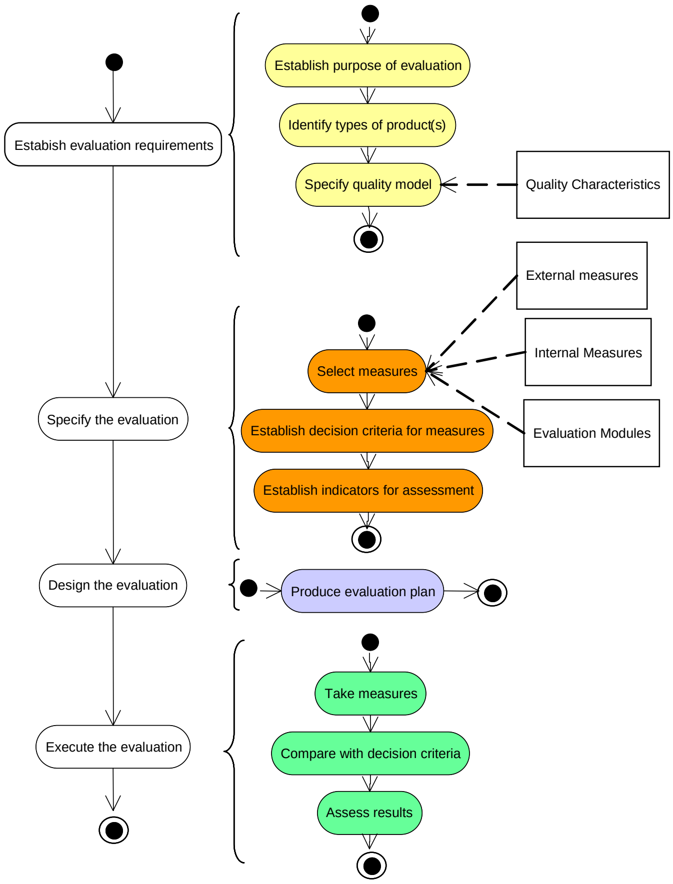


Figure 15. Evaluation Process (adapted from ISO 14598-1)

The construction of the measurement model that operationalizes the quality model covers the specification of quality requirements in terms of quality characteristics and sub-characteristics (measurable concepts in terms of the WE-SMM), as suggested in ISO 14598. The correctness of the resulting measurement model is assured by (1) the WE-SMM and (2) the underlying validated general quality model. The completeness of the measurement model is also influenced (although not guaranteed) by these two elements.

During the measurement model instantiation, the activities marked in orange in Figure 15 (*measure selection, decision criteria establishment* and *indicator assessment*) must be performed. However, the fact that we depart from independent quality models for each level of abstraction again facilitates the execution of these activities. Ideally, all measures contained in a WE quality model should be reflected in any measurement model derived from such quality model, and applicable to any Web application. Only decision criteria and indicators are likely to need to be fine-tuned according to the particular type of application we are dealing with. The reason is that quality requirements are not the same for different domains (e.g. e-commerce applications and educational applications) and, even inside a given domain, requirements could vary (e.g. it is not the same dealing with an educational application for children than dealing with an educational application for computer science professionals).

The e*valuation plan production* (marked in blue in Figure 14) is also implicit in the WE process presented in Figure 13. Briefly speaking, our proposed schedule is to evaluate each artefact as soon as is it produced in the development process. On the other hand the evaluation method is expressed during the measurement model construction.

Finally, we propose to execute the evaluation in an automatic way, by means of transformation rules that interact with the WE-SMM and with the particular WE artifact meta-model to (1) get measures results, (2) calculate indicators, (3) compare indicators with decision criteria and (4) if feasible, evolve the models to improve the indicator value. All these activities, marked in red in Figure 15, will be briefly described in Section 6.

5.1. Conclusion

In this section we have presented how the quality-aware WE process (see Figure 13) is compliant with the ISO/IEC 14598. The fact that it is an MDE-driven approach also covers the tool support demanded by both the ISO standards and the practitioners' community.

6. Automation of the Evaluation Process and Design Guidelines

As we presented in Figure 13, our proposal includes the execution of the WE quality evaluation process in an automatic way, following the MDE paradigm, thus alleviating P9. This is achieved by means of QVT-based transformation rules that interact with the WE-SMM and with the particular WE artefact meta-model provided by WE methodologies. During this interaction the transformation rule permits to (1) calculate measures/indicators results, (2) compare indicators with decision criteria, (3) annotate the models with the evaluation results and (4) if feasible, evolve the models to improve the measure value. To achieve these goals, each transformation rule contains in its *when* clause a translation of the selected measure/indicator and its related threshold values in terms of OCL expressions over the chosen WE artefact meta-model. A detailed description and a proof of concept of how this automation of the evaluation process works can be found in [9]. Here we are just briefly outlining some open issues regarding such automated execution.

First, and given the fact that the ISO/IEC 14598 does not establish what to do if the result of executing the 'Evaluation rule' (the rule that contains the codification of the measure) does not meet the criteria, in the context of our proposed quality-aware WE process two actions are possible:

1. Annotate the model to warn the designer, who would be in charge of manually perform the changes needed
2. Automatically trigger a chain of subordinated 'Evolution rules' that evolve the model to improve the measure until the value is consistent with the quality requirements.

Whatever the case, the result of applying such transformation rules on the original models is a quality-assessed (QA) WE model, as we can observe in Figure 13. Interested readers can find examples of both types of rules in [8] and [9] respectively.

Although the automated nature of the WE process would suggest the second course of action, we are conscious that it is not always possible to automatically decide which changes to make on the models. Therefore extensive research needs to be done in order to come up with model evolutions that truly improve the quality of the final application.
Also we would like to note how measures, indicators and decision criteria defined for a certain application remain coded in such transformation rules, which need to be defined only once for each WE methodology. From them, indicators should be validated as predictors of the actual product quality (measured on the deployed code under real conditions of use), and the result of such validation also stored in any kind of project repository. The shape of such project repository, which ideally should be defined by consensus in order to be able to merge results gathered with different WE methodologies and refine the predictive power of indicators, also remains an open line of research.

7. Conclusions and Further Work

In this technical report we have proposed an approach to evaluating Web quality that provides all the elements that, according to the ISO/IEC 14598, are essential parts of a software quality evaluation, namely (1) *a quality model, (2) a method of evaluation, (3) a software measurement process and (4) supporting tools.* We have integrated all these elements following the claim that, *to develop good software* (and thus good Web-based software), *quality requirements should be specified, the software quality assurance process should be planned, implemented and controlled, and both intermediate products and final products should be evaluated.* Also, *to achieve objective software quality evaluations, the quality attributes of the software should be measured using validated metrics* [23]. How all these elements contribute to alleviate each of the problems detected in Section 1.3. is presented in Table 6.

Table 6. Problems and Solutions: a summary

Problem	Solution
P1	Use of a SMO (Section 2)
P2 and P3	Operationalization of quality models by means of a WE-SMM instantiation that takes into account a specific stakeholder and a specific WE artefact (Section 2)
P4 and P5	Empirical validation of quality models (Section 3)
P6 and P7	Definition of a Quality assurance process that is ISO compliant and that is integrated with the WE development process (Sections 4 and 5)
P8 and P9	Automation of Quality Assessment Process by means of transformation rules that have tool support and that may simply evaluate or evaluate and evolve the WE models according to certain quality criteria (Section 6)

The main purpose of our approach is to ease the shift of the WE community towards addressing quality during the systematic development of Web applications. In order to systematize such quality concerns in a seamless way, our framework operationalizes (validated) quality models by means of measurement models. Such measurement models are then integrated by means of transformation rules with the traditional WE development process, therefore preserving the semi-automated nature of such processes.

The definition of such measurement models as instances of a WE-SMM has several advantages:

- The meta-model provides a table of contents which makes visible what information is necessary to include in the measurement model and how this information is related
- The meta-model supports the manipulation of the measurement model in an automatic way, since information will show an homogeneous structure
- The meta-model supports the reuse of measurement models since it facilitates the establishment and maintenance of libraries of measurement models
- The meta-model supports standardization of measurement models since the format is compliant with an accepted SMO.
- The meta-model covers all the information defined in ISO 14598-6 as necessary parts of an evaluation module documentation.

Additionally, the empirical validation promoted for the quality models that support the measurement models strives to assure that the minimal effort is made in order to get an adequate Web quality in use.

Nonetheless, the definition of a quality evaluation process that is based on the MDE paradigm also provides several advantages:

- Automation of the quality assurance process
- Leverage of costs and timeframes
- Standard tool support

Finally, the integration of the MDE-based quality assurance process with the MDE-based WE process causes that the use of a WE process to develop a Web Application implicitly assumes a planned, synchronized quality assurance of both intermediate and final products, easing the adoption of quality practices in the WE field.

The modified WE process presented in this paper constitutes a step towards the Total Quality Management[49]. We agree with [33] in that the empirical evidence of output quality of WE methodologies, as well as result demonstrability would influence its *perceived usefulness*, which in turns explains up to 40% of technology adoption. As a conclusion, we may say that our approach implies to a certain extent a shift in the “center of gravity” of WE from creating technology-centred solutions towards satisfying the stakeholders [47].

However, the use of our proposal also poses some risks that must be taken into account and that constitute future lines of research:

- We provide a fixed quality evaluation process planning, which depends on the main WE process planning. This could not be feasible and/or advisable in certain circumstances
- The agreement on common quality models for each level of abstraction is far from trivial, and may depend on the background of the researchers/practitioners involved in reaching such agreement.
- The quality model tailoring process necessary to construct a measurement model that meets specific application quality requirements is not a trivial task. First, a previous task of transforming quality requirements into specification requirements at each level of abstraction is needed. Also, decision criteria associated with each specification requirement are not easy to establish.
- The usefulness of assuring quality must be counterbalanced with the extra complexity added to the WE process due to its merging with the quality evaluation process. However, we think that the automation of the measurement process alleviates this problem.
- Quality-assured models are nothing if transformation rules themselves are not also quality-assured. The process followed to transform each model into another should be evaluated to assure not only efficiency but also that they at least preserve the level of quality assured in previous levels of abstraction.
- Extensive work must be done to define each measure in terms of transformation rules for each WE approach. Keeping the number of measures included in the models low and reaching an agreement on a common WE meta-model may leverage such risk.

- Similarly, differences in semantics associated to each level of abstraction among WE methodologies may hamper the task of defining a common quality model that includes measures and that serves as a basis to define measurement models. Again, reaching an agreement on a common WE meta-model may leverage such risk.

We have left out of the scope of this paper related fields of research such as the quality of the models *per se* or the quality of the process as it improves the cost of building applications. However, we believe that some of the concepts presented in this paper may be reused to study the impact of using a WE methodology on such fields of research.

Last, we are aware that research knowledge is not intrinsically valuable: it only becomes valuable if it is used in practice [38]. Successful WE technology practice depends on two-way knowledge transfers between research and practice rather than ideas flowing in only one direction. Therefore there is a need for collaboration between researchers and practitioners if we are to convert our new ideas (inventions) in real innovations, adopted by the Web community.

We do hope that this technical report provides other researchers with the necessary insights to start working on the definition of (validated) quality models at different levels of abstraction in the WE field, as well as on the operationalization of these quality models and its integration with common WE practices. Only if the research community generates enough empirical data to back the WE assumptions will it be possible to convince the practitioner of the advantages of using WE methodologies.

Acknowledgments

This paper has been supported by the Spanish Ministry of Science and Technology (Grant PR2006-0374 for University Teaching Staff Stages at foreign Universities, projects MEC-FEDER (TIN2004-03145), ESFINGE (TIN2006-15175-C05-05), METASIGN (TIN2004-00779) and DSDM (TIN2005-25866-E)). Also, this research is part of the DADASMECA project (GV05/220), financed by the Valencia Government and the DADS (PBC-05-012-2) and the DIMENSIONS (PBC-05-012-1) projects, financed by the Regional Science and Technology Ministry of Castilla-La Mancha (Spain).

REFERENCES

[1] Abrahao, S., Condory-Fernandez, N.; Olsina, L. and Pastor, O. (2003) Defining and Validating Metrics for Navigation Models. In Proceedings of the Ninth IEEE International Software Metrics Symposium, p 200-210, IEEE Press, Wiley, Chichester.

[2] Abrahao, S; Insfran, E. (2006) Early usability evaluation in Model-Driven Architecture Environments. In Proceedings of the Sixth IEEE International Conference on Quality Software, IEEE Press, Wiley, Chichester.

[3] Baresi L, Morasca S. and Paolini P. Estimating the Design Effort of Web Applications. IEEE Metrics 2003: 62-72

[4] Basili, V.R., Galdiera, G. and Rombach, H.D. (1994) Goal Question Metric paradigm, in: J. J. Marciniack (Ed.), Enciclopaedia of Software Engineering, Vol 1, John Wiley and Sons, 1994.

[5] Bohem, B.W., (1981) Software Engineering Economics, Prentice Hall Inc., Englewood Cliffs, NJ, 1981

[6] Briand LL, Morasca S. and Basili V (1999) Defining and Validating Measures for Object-based High-level Design. IEEE Transactions on Software Engineering 25(5), 722-743.

[7] Cachero, C. (2005): *Web Application Internal Navigability Measures*. Available on-line at http://www.dlsi.ua.es/~ccachero/pPublicaciones.htm. Last updated November 2006.

[8] Cachero, C., Genero, M., Calero, C. and Meliá, M (2006) Quality-Driven automatic transformations of Object-Oriented Navigational Models. ER 2006 (Workshops). Springer Verlag, Heidelberg, 329-338*

[9] Cachero, C.; Meliá, M.; Genero, G., Poels, G. and Calero, C. (2006). Towards improving the navigability of Web applications: a Model Driven Approach. Nov 2006. Working Paper D/2006/7012/64. Available On-line at: www.feb.ugent.be/Fac/Research/WP/Papers/ wp_06_419.pdf

[10] Cachero, C., Koch, N., Gómez, J. and Pastor, O. (2002) Conceptual Navigation Analysis: a device and platform independent navigation specification. 2nd International Workshop on Web Oriented Software Technology. Málaga, Spain. June 2002.

[11] Calero, C., Ruiz, J. and Piatinni, M. (2005) Classifying web metrics using the web quality model. In Online Information Review Journal, OIR – 29,3, Emerald Literari, United Kingdom.

[12] Comai S.; Matera, M.; Maurino, A. (2003) A Model and an XSL Framework for Analyzing the Quality of WebML Conceptual Schemas. In Proceedings of the International Workshop on Conceptual Modeling Quality, p 339-350. Springer-Verlag, Heidenberg, LNCS 2784.

[13] Dromey, R.G. (1995) A model for software product quality. IEEE Transactions on Software Engineering Vol 21(2), pp 146-162

[14] Dromey R.G. (1998) Software Product Quality: Theory, Model and Practice. Software Quality Institute, Griffith University, Nathan, Brisbane, QLD 4111, Australia. Available online at: http://scholar.google.com/url?sa=U&q=http://www.sqi.gu.edu.au/docs/sqi/misc/SPQ-Theory.pdf

[15] Ferreira, M; García, F.; Bertoa, M.; Calero, C.; Vallecillo, A.; Ruiz, F.; Piattini, M. and Braga, JL. (2006). Medición del Software: Ontología y Metamodelo. Technical Report, University of Castilla-La Mancha.

[16] Fraternali, P., and Paolini, P. (2000). Model-driven development of Web applications: the Autoweb System. ACM Transactions on Information Systems (TOIS), 18(4), 323–382.

[17] García, F.; Bertoa, M.; Calero, C.; Vallecillo, A.; Ruiz, F.; Piattini, M.; Genero, M. (2005) Towards a consistent terminology for software measurement. Information and Software Technology 48(8), 631-644.

[18] Garvin, D. (1984) What Does “Product Quality” Really Mean?, Sloan Management Review, Fall 1984, pp. 25-45.

[19] Genero, M.. Defining and Validating Metrics for Conceptual Models. PhD thesis, University of Castilla-La Mancha, 2002.

[20] Heuser, L (2004) The real world or Web Engineering? In Proceedings of the Fourth International Conference on Web Engineering, p 1-5, Springer-Verlag, Heidelberg, LNCS 3140.

[21] ISO/IEC 9126 (2001) Software engineering – Product quality – Part 1: Quality model. International Organization for Standardization, Geneva.

[22] ISO/IEC 10027 (1990). Information technology – Information Resource Framework. International Organization for Standardization. Geneva.

[23] ISO/IEC 14598 (1999). Information technology – Software Product Evaluation. International Organization for Standardization. Geneva.

[24] Ivory MY (2004) Automated Web Site Evaluation. Kluwer Academic Publishers, Norwell.

[25] Ivory MY and Hearst MA (2001) The state of the art in automating usability evaluation in user interfaces. ACM Computing Surveys 33(44), 470-516

[26] Ivory MY, Megraw R (2005) Evolution of Web Site Design Patterns. ACM Transactions on Information Systems 23(4), 463-497.

[27] Kent S (2002) Model-Driven Engineering. In Proceedings of the Third International Conference on Integrated Formal Methods, p 286-298, Springer-Verlag, Heidelberg, LNCS 2335.

[28] Kesh, S. (1995): Evaluating the Quality of Entity Relationship Models, Information and Software Technology, 37 (12).

[29] Kitcheham, B., Dyba T. and Jorgensen M. (2004) Evidence-based Software Engineering. In Proceedings of the 26th International Conference on Software Engineering (ICSE'04).

[30] Kitchenham, B., Pfleeger, S.L., (1996) Software Quality: The Elusive Target, IEEE Software, vol. 13(1), pp. 12-21.

[31] Krogstie J., Lindland O.I. and Sindre G. (1995) Defining quality aspects for conceptual models. ISCO 1995: 216-231

[32] Lang, M.; Fitzgerald, B. (2005) Hypermedia Systems Development Practices: A Survey. IEEE Software 22(2), 68-75.

[33] Legris P, Ingham, J. and Collerette P. (2003) Wy do people use information technology? A critical review of the technology acceptance model. Information and Management vol:40 (3) pg:191

[34] Lindland O.I., Sindre G. and Solvberg A. (1994) Understanding Quality in Conceptual Modeling. IEEE Software Vol 11(2) pp 42 – 49. IEEE Computer Society Press. Los Alamitos, CA, USA

[35] McCall, J.A., Richards, P.K., and Walters, G.F. (1977), Factors in Software Quality, Nat'l Tech. Information Service, Vol 1, 2 and 3.

[36] Mendes, E. (2005) A Systematic Review of Web Engineering Research. 2005 International Symposium on Empirical Software Engineering. Pp 10-18 ISBN: 0-7803-9507-7

[37] Moody D.L. and Shanks G.G. (2003). Improving the quality of data models: empirical validation of a quality management framework. Information Systems 28: 619-650. Elsevier Science Ltd.

[38] Moody D.L. (2005) Theoretical and practical issues in evaluating the quality of conceptual models: current state and future directions. Data & Knowledge Engineering 55, p 243-276.

[39] Moraga, M., Calero, C., Piattini, M. (2006) Ontology Driven Definition of a Usability Model for Second Generation Portals. In Proceedings of the 1st International Workshop on Methods, Architectures & Technologies for e-Service Engineering (MATeS 2006) ACM Press. Vol. 155. ISBN:1-59593-435-9

[40] Nielsen J (2000). Designing Web Usability: The Practice of Simplicity. New Riders, Berkeley.

[41] Olsina, L.; Rossi, G, (2002) Measuring Web Application Quality with WebQEM, In IEEE Multimedia Magazine, Vol. 9, Nº 4, pp. 20-29

[42] QUINT2: The Extended ISO Model of Software Quality. Available on-line at: http://www.serc.nl/quint-book/

[43] Ruiz, F. and Hilera, J.R. (2006) Using Ontologies in Software Engineering and Technology. In Ontologies for Software Engineering and Software Technology. C. Calero, F. Ruiz and M. Piattini (Eds). Springer Berlin Heidelberg

[44] Seffah A., Donyaee M. Kline R.B. and Padda H.K. (2006) Usability measurement and metrics: a consolidated model. Software Quality Journal Vol 14. pp 159-178.

[45] Si-Said S., Akoka J. and Comyn-Wattiau, I. Use Case Modeling and Refinement: A Quality-Based Approach. ER 2006: 84-97

[46] Software Quality Models and Philosophies. Chapter 1. Available on-line at http://www.bth.se/tek/besq.nsf/(WebFiles)/CF1C3230DB425EDCC125706900317C44/$FILE/chapter_1.pdf

[47] Suryn W, Abran, A. and April A.(2003) ISO/IEC SQuaRE. The second generation of standards for software product quality. 7th IASTED International Conference on Software Engineering.

[48] Walrad and Moss (1993) Measurement: the key to application development quality. IBM Systems Journal Vol 32 (3). p 445-460.

[49] Zultner, R.E. The Deming way: total quality management for software, Proceedings of Total Quality Management for Software Conference, Washington, DC, April, 1992.