4 Web Quality

Luis Olsina, Guillermo Covella, Gustavo Rossi

Abstract: In this chapter we analyse the different quality perspectives of software and Web applications. In particular, we review quality taking into account the ISO (International Organization for Standardization) standards for software product, and discuss the distinction between quality and quality in use, and how different requirements, from different users' standpoints, should be considered as well. Moreover, we also describe Web quality and how it can be measured and evaluated. In order to illustrate the specific procedures and processes of an inspection evaluation methodology, a case study on the external quality of the shopping cart component of two typical e-commerce Web applications is presented.

Keywords: Web quality, quality measurement, Logic Scoring Preference.

4.1 Introduction

The quality of an entity is easy to recognise but hard to define and evaluate. Although the term seems intuitively self-explanatory, there are actually many different perspectives and approaches to measure and evaluate quality as part of a software or Web development, operation, and maintenance processes.

The meaning of quality is not simple and atomic, but a multidimensional and abstract concept. Common practice assesses quality by means of the quantification of lower abstraction concepts, such as attributes of entities. The attribute can be briefly defined as a measurable property of an entity.[1] An entity may have many attributes, though only some of them may be of interest to a given project's measurement and evaluation purposes. Therefore, quality is an abstract relationship between attributes of entities and information needs (measurement goals).[2] Figure 4.1 specifies some of these terms and their relationships.

To illustrate these concepts let us consider the following example. One of the goal's of an organisation's project, within a quality assurance plan, is to "*evaluate the link reliability of a Web application's static pages*". The

---

[1] Types of entities of interest to software and Web engineering are resource, process, product, product in use, and service.

[2] In fact, quality, quality in use, cost, etc., are instances of a computable concept.

purpose is to *evaluate* the *link reliability* calculable concept for *static Web pages* as the product entity, from a user's viewpoint; we can see that the link reliability sub-concept is a sub-characteristic related to the external quality of a product. Considering the level of abstraction, a calculable concept can be composed of other sub-concepts that may be represented by a concept model (e.g. ISO 9126-1 [13]specifies the external quality model based on characteristics and sub-characteristics). A calculable concept combines one or more attributes of entities. Figure 4.2 shows a simple concept model where three attributes are part of the *link reliability* calculable concept.

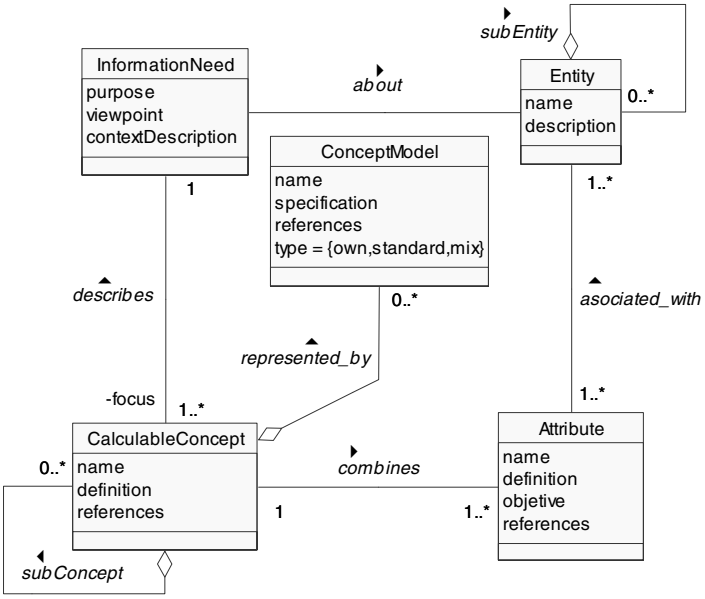

Fig. 4.1. Main terms and relationships related with the calculable concept term where quality or quality in use are instances of it

1. Link Reliability
   1.1 Internal Broken Links (IBL)
   1.2 External Broken Links (EBL)
   1.3 Invalid Links (IL)

Fig. 4.2. A concept model for the link reliability calculable concept

On the other hand, each attribute can be quantified by one or more metrics.[3] The metric contains the definition of the selected measurement method and scale (the conceptual model of metrics and indicators are introduced in Sect. 4.3.2).

The previous example, which does not include other external quality subconcepts, such as efficiency, functionality, and usability, is intended to show that the meaning of quality is not atomic but rather a complex, multidimensional concept. Quality cannot be measured directly, at least not in a trivial and subjective way. On the other hand, the requirements for quality can vary depending on the entity type, user's viewpoint, and context of use. Regarding the entity type (e.g. process, product), quality requirements specified as quality models can differ from one another. Moreover, we can specify different requirements, from different users' standpoints, for the same entity type. In addition, the quality perception for the same software or Web product can vary depending on contexts of use for the same user type!

In Sect. 4.2, we discuss the different perspectives of quality for software. In particular, in Sect. 4.2.1 we review the state of the art of quality according to the ISO standards for software quality; we also address the importance of distinguishing between quality and quality in use (see Sect. 4.2.2); and how different requirements, from diverse users' standpoints, should be considered (see Sect. 4.2.3).

The next section describes Web quality, focusing on the quality of Web products and the perceived quality of real users in a real context of use.

Nowadays, the Web plays a central role in diverse application domains such as business, education, government, industry, and entertainment. The Web's growing importance heightens concerns about Web applications' development and evaluation methods, and requires the systematic use of engineering models, methods, and tools. In particular, we need sound evaluation methods for obtaining reliable information about product quality and quality in use. There are different categories of methods (e.g. inspection, testing, inquiry, simulation) and specific types of evaluation methods and techniques (e.g. heuristic evaluation technique [19,20], the concept model-centred evaluation method [24]). In Sect. 4.3 we present the Web Quality Evaluation Method (WebQEM) as a model-centred evaluation method. Using WebQEM to assess Web applications helps to meet quality requirements in new Web development projects and to evaluate requirements in operational phases. It also helps discover absent attributes or poorly implemented requirements, such as interface-related designs, and implementation drawbacks or problems with navigation, accessibility, search mechanisms, content, reliability and performance, among others.

---

[3] Metric and measure mean the same within the context of this book.

Section 4.4 presents a case study where the external quality of the shopping cart component of two typical e-commerce Web applications is assessed, using the specific models, procedures, and processes of the Web-QEM methodology. In Sect. 4.5 concluding remarks to this chapter are drawn.

4.2 Different Perspectives of Quality

The essential purpose-oriented evaluation of quality characteristics and attributes for different entities is not an easy endeavour in either software or Web engineering [18]. It is difficult to consider all the characteristics and mandatory or desirable attributes of a process, or a product (e.g. Web application), without using sound quality frameworks, models, and methods. These allow evaluators to specify systematically goal-oriented quality concepts, sub-concepts, and attributes. An example of a generic quality model is provided by the ISO standards for specifying quality requirements in the form of quality models to software processes and products.

As previously mentioned, quality requirements can vary depending on the entity type, the users' viewpoint, and the context of use. From a software measurement and evaluation point of view, we can identify different entity types at a high level of abstraction, i.e. a resource, a process, a product, a service, a product or a system in use, as well as a software or Web project. Quality requirements can be specified using a concept model representing quality or quality in use. Studies have shown that resource quality can potentially help improve process quality; process quality can help improve product quality, which can help improve quality in use [13]. In the same way, evaluating quality in use can provide feedback to improve product quality; evaluating a product can provide feedback to improve process quality; and evaluating a process can provide feedback to improve resource quality (see Fig. 4.3).

Within the context of this chapter we focus on product quality and quality in use.

4.2.1 Standards and Quality

One standardisation milestone of the software product quality for assessment purposes occurred at the end of 1991, when ISO/IEC issued the quality model and the evaluation process model [9]. Previously, seminal work defined software quality models and frameworks; among these were the quality models specified by McCall, Boehm, and the US Air Force

(see [9]). The aim of the ISO/IEC organisation was to reach the required consensus and to encourage international agreement.

The ISO/IEC 9126 standard prescribes six characteristics (sub-concepts) that describe, with minimal overlap, software quality. In addition, it presents a set of quality sub-characteristics for each characteristic. As it also specifies an evaluation process model, the two inputs to the quality requirement definition step are the ISO quality model and stated or implied user needs.

The quality definition in this standard is "*The totality of features and characteristics of a software product that bears on its ability to satisfy stated or implied needs*" ([9]this definition is adopted from the previous ISO 8402 standard entitled "*Quality – Vocabulary*" issued in 1986). The six prescribed characteristics useful to evaluate product quality are *Usability*, *Functionality*, *Reliability*, *Efficiency*, *Portability*, and *Maintainability*. For instance, *Usability* is defined as "*A set of attributes that bear on the effort needed for use, and on the individual assessment of such use, by a stated or implied set of users.*" In turn, usability is broken down into three sub-characteristics, namely: *Understandability*, *Learnability*, and *Operability* (e.g. operability is defined as "*Attributes of software that bear on the users' effort for operation and operation control*"*).*

Other aspects of this standard are as follows:

- The meaning of quality is taken as a complex, multidimensional concept that cannot be measured directly.
- Given the complexity that the quality concept embraces, a quality model to specify software product quality requirements is needed.
- The general-purpose quality model contains a minimum number of characteristics by which every type of software can be evaluated.
- For the quality requirement definition step, the stated or implied user needs are considered. In addition, the term user is acknowledged in some definitions of characteristics and sub-characteristics (e.g. usability and its sub-characteristics).
- ISO 9126 differs from traditional quality approaches that emphasise the need to meet requirements that are primarily functional (e.g. the manufacturing quality approach of ISO 9000).

As observed above, the ISO 9126 definitions acknowledge that the goal of quality is to meet user needs. But what is not clearly stated is that the purpose of software quality is to be "perceived with quality": that is, with degrees of excellence by end users in actual contexts of use. Rather, ISO 9126 suggests that quality is determined by the presence or absence of the attributes, with the implication that these are specific attributes which can be designed into the product. As Bevan [2]says:

"Although developers would like to know what attributes to incorporate in the code to reduce the 'effort required for use', presence or absence of predefined attributes cannot assure usability, as there is no reliable way to predict the behaviour of the users of the final product."

To fill this gap, the ISO 9126 standard has been revised to specify a quality framework that distinguishes among three different approaches to software quality – internal quality, external quality, and quality in use. The ISO/IEC 9126-1 standard, which includes these three approaches to quality, was officially issued in 2001 [13]. The evaluation process model initially included in ISO 9126 was moved to and fully developed in the ISO/IEC 14598 series [11,12]. The three approaches of quality in ISO 9126-1 can be summarised as follows:

- *Internal quality*, which is specified by a quality model (similar to the ISO 9126 model), and can be measured and evaluated by static attributes of documents such as specification of requirements, architecture, or design; pieces of source code; and so forth. In early phases of a software lifecycle, we can evaluate and control the internal quality of these early products, but assuring internal quality is not usually sufficient to assure external quality.
- *External quality*, which is specified by a quality model (similar to the ISO 9126 model), and can be measured and evaluated by dynamic properties of the running code in a computer system, i.e. when the module or full application is executed in a computer or network simulating as close as possible the actual environment. In late phases of a software lifecycle (mainly in different kinds of testing, or even in the acceptance testing, or furthermore in the operational state of a software or Web application), we can measure, evaluate, and control the external quality of these late products, but assuring external quality is not usually sufficient to assure quality in use.
- *Quality in use*, which is specified by a quality model (similar to the ISO 9241-11 model [10]), and can be measured and evaluated by the extent to which the software or Web application meets specific user needs in the actual, specific context of use.

The internal quality definition in ISO 9126-1 is *"the totality of attributes of a product that determines its ability to satisfy stated and implied needs when used under specified conditions"*; the external quality definition is *"the extent to which a product satisfies stated and implied needs when used under specified conditions"*; and the quality in use definition is *"the extent to which a product used by specified users meets their needs to achieve specified goals with effectiveness, productivity and satisfaction in*

specified context of use" (note that these definitions are in the ISO/IEC 14598-1 standard [12])

These three slightly different definitions of quality (instead of the unique definition in the previous 9126 standard) refer particularly to the product when it is used under specified conditions and context of use, so making it clear that quality is not an absolute concept, but depends on specific conditions and context of use by specific users.

The same six prescribed quality characteristics have been maintained in the revised internal and external quality models. Moreover, sub-characteristics are now prescriptive. Besides, new sub-characteristics were added and redefined in terms of "*the capability of the software*" to enable them to be interpreted as either an internal or an external perspective of quality. For instance, usability characteristic is defined in [13]as "*The capability of the software product to be understood, learned, used and attractive to the user, when used under specified conditions.*" In turn, usability is subdivided into five sub-characteristics, namely: *Understandability*, *Learnability*, *and Operability*, in addition to *Attractiveness* and *Usability compliance* (see Table 4.1 for the definition of these sub-characteristics).

Table 4.1. Definition of usability sub-characteristics prescribed in ISO 9126-1 [13]for internal and external quality

Sub-characteristic	Definition
Understandability	The capability of the software product to enable the user to understand whether the software is suitable, and how it can be used for particular tasks and conditions of use.
Learnability	The capability of the software product to enable the user to learn its application.
Operability	The capability of the software product to enable the user to operate and control it.
Attractiveness	The capability of the software product to be attractive to the user.
Compliance	The capability of the software product to adhere to standards, conventions, style guides or regulations relating to usability.

External quality is ultimately the result of the combined behaviour of the software component or application and the computer system, while quality in use is the effectiveness, productivity, safety, and satisfaction of specific users when performing representative tasks in a specific, realistic working environment. By measuring and evaluating the quality in use (by means of metrics and indicators) the external quality of the software or Web application can be validated. Quality in use evaluates the degree of excellence, and can be used to validate the extent to which the software or

Web application meets specific user needs. In turn, by measuring and evaluating external quality, a software product's internal quality can be validated. Similarly, taking into account suitable software/Web application attributes for internal quality is a prerequisite to achieve the required external behaviour, and to consider suitable software attributes to external behaviour is a prerequisite to achieve quality in use (this dependency is suggested in Fig. 4.3).

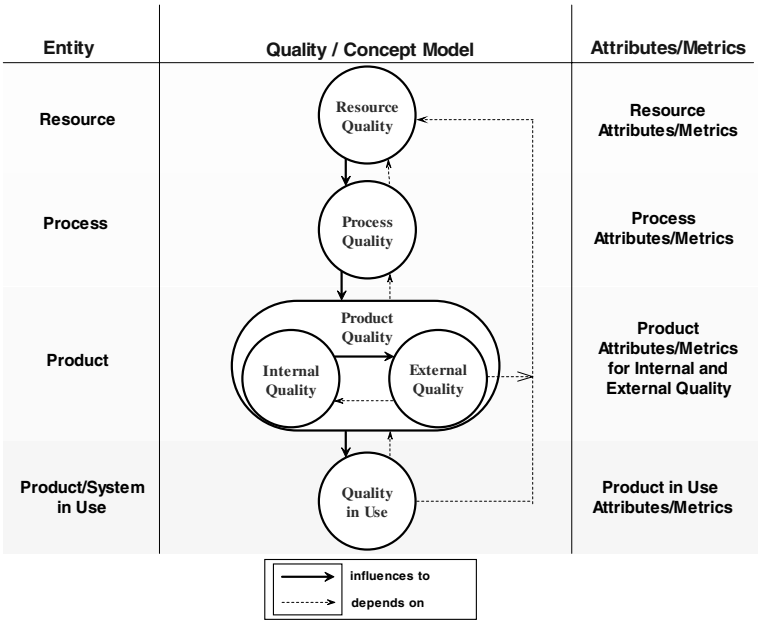

Fig. 4.3. Framework of quality regarding different entity types and potential quality models

The basic example introduced in Figure 4.2 focuses on external quality because we cannot measure such application attributes (i.e. IBL, EBL, IL) without Web server and network infrastructure support.

4.2.2 Quality Versus Quality in Use

While users are becoming more and more mature in the use of IT systems and tools, there is greater demand for the quality of software and Web applications that match real user needs in actual working environments.

The core aim in designing an interactive (software or Web) application is to meet the user needs; that is, to provide degrees of excellence or quality in use by interacting with the application and by performing its tasks

comfortably. Within the context of the ISO 9126-1 standard, quality in use is the end user's view of the quality of a running system containing software, and is measured and evaluated in terms of the result of using the software, rather than by properties of the software itself. A software product's internal and external quality attributes are the cause, and quality in use attributes are the effect. According to Bevan [2]:

> *"Quality in use is (or at least should be) the objective, software product quality is the means of achieving it."*

Quality in use is a broader view of the ergonomic concept of usability as for ISO 9241-11 [10]. Quality in use is the combined effect of the internal and external quality characteristics for the end user. It can be measured and evaluated by the extent to which specified users can achieve specified goals with effectiveness, productivity, safety, and satisfaction in specified contexts of use. Table 4.2 shows the definition of these four characteristics, and Fig. 4.4 outlines a partial view of the quality in use (concept) model and associated attributes.

Table 4.2. Definition of the four quality in use characteristics prescribed in ISO 9126-1

Characteristic	Definition
Effectiveness	The capability of the software product to enable users to achieve specified goals with accuracy and completeness in a specified context of use.
Productivity	The capability of the software product to enable users to expend appropriate amounts of resources in relation to the effectiveness achieved in a specified context of use.
Safety	The capability of the software product to achieve acceptable levels of risk of harm to people, business, software, property or the environment in a specified context of use.
Satisfaction	The capability of the software product to satisfy users in a specified context of use. Note [by ISO]. Satisfaction is the user's response to interaction with the product, and includes attitudes towards use of the product.

In order to design and select metrics (and indicators) for assessing quality in use it is first necessary to associate attributes to the effectiveness, productivity, safety, and satisfaction characteristics. Figure 4.4 shows attributes for two characteristics, namely effectiveness and productivity.

Quality in Use
1. Effectiveness
   1.1 *Task Effectiveness* (TE)
   1.2 *Task Completeness* (TC)
   1.3 *Error Frequency* (EF)
2. Productivity
   2.1 *Efficiency related to Task Effectiveness* (ETE)
   2.2 *Efficiency related to Task Completeness* (ETC)

Fig. 4.4. Specifying an instance of the Quality in Use model

Note that effectiveness, productivity, safety, and satisfaction are influenced not only by the usability, functionality, reliability, and efficiency of a software product, but also by two resource components of the context of use. The context of use depends on both the infrastructure (i.e. the computer, network, or even the physical working medium) and the user-oriented goals (i.e. the supported application tasks and the properties of the user type such as level of training, expertise, and cultural issues as well). Care should be taken when generalising the results of any quality in use assessment to another context of use with different types of users, tasks, or environments [2].

As a consequence, when designing and documenting quality in use measurement and evaluation processes, at least the following information is needed:

- Descriptions of the components of the context of use, including user type, equipment, environment, and application tasks (tasks are the steps or sub-goals undertaken to reach an intended goal).
- Quality in use metrics and indicators for the intended purpose and measurement goal(s).

As a final remark, it can be observed that quality is not an absolute concept; there are different quality perspectives both to a product and to a product in a context of use. Internal quality, external quality, and quality in use can then be specified, measured and evaluated. Each of these perspectives has its own added value considering a quality assurance strategy in the overall lifecycle. However, the final objective is the quality in use. How a concept model (quality, quality in use) can be instantiated for different user standpoints is discussed next.

4.2.3 Quality and User Standpoints

In a measurement and evaluation process, the quality requirements specified in the form of a quality model should be agreed upon. The quality model can be a standard-based quality model, a project or organisation's proprietary quality model, or a mixture of both.

Depending on the goal and scope of the evaluation, the concept model and corresponding characteristics and attributes that might intervene should be selected. Moreover, the importance of each characteristic varies depending on the application's type and domain, in addition to the user standpoint taken into account. Therefore, the relative importance of characteristics, sub-characteristics, and attributes depends on the evaluation's goal and scope, the application domain, and the user's viewpoint.

When designing an evaluation process, the assessment purpose and scope may be manifold. For instance, the purpose can be to understand the external quality of a whole software application or one of its components; we might want to predict the external quality by assessing the internal quality of a software specification, or to improve the quality in use of a shopping cart component, or to understand and compare the external quality of two typical e-commerce Web applications to incorporate the best features in a new development project. On the other hand, the type of applications can be at least categorised as mission-critical, or non-mission-critical, and the domain can be diverse (e.g. avionics, e-commerce, e-learning, information-oriented Web applications).

Lastly, the user standpoint for evaluation purposes can be categorised as one of an acquirer, a developer, a maintainer, a manager, or a final (end) user. In turn, a final user can, for instance, be divided into a novice user or an expert user. Thus, final users are mainly interested in using the software or Web application, i.e. they are interested in the effects of the software rather than in knowing the internal aspects of the source code or its maintainability. For this reason, when the external quality requirements are, for example, defined from the end user's standpoint, generally usability, functionality, reliability, and efficiency are the most important. Instead, from the maintainer's viewpoint, analysability, changeability, stability, and testability of application modules are the most important.

As a final comment, we would like to draw the reader's attention to the conceptual model shown in Fig. 4.1. That basic model is a key piece of a set of tools we are currently building for measurement and evaluation projects. Given an *entity* (e.g. e-learning components to support course tasks), it allows us to specify an evaluation *information need:* that is to say, the *purpose* (e.g. understand), the *user viewpoint* (e.g. a novice student), in a given *context* of use (e.g. the software is installed in the engineering school server as support to a preparatory mathematics course for pre-enrolled

students, etc.), with the *focus* on a *calculable concept* (quality in use) and *sub-concepts* (effectiveness, productivity, and satisfaction), which can be *represented by* a *concept model* (e.g. the ISO quality in use model) and associated *attributes* (as shown in Fig. 4.4).

The next section describes Web quality. The main focus is on the quality of Web products and the perceived quality of real users in a real context of use.

4.2.4 What is Web Quality?

According to Powell [26]Web applications "*involve a mixture between print publishing and software development, between marketing and computing, between internal communications and external relations, and between art and technology*".

Nowadays, there is a greater awareness and acknowledgement in the scientific and professional communities about the multidimensional nature of Web applications; it encompasses technical computing, information architecture, contents authoring, navigation, presentation and aesthetic, multiplicity of user audiences, legal and ethical issues, network performance and security, and heterogeneous operational environments.

As pointed out in Chap. 1, Web applications, taken as product, or product in use entities (without talking about distinctive features of Web development processes), have their own features, distinct from traditional software [18,26], namely:

- Web applications will continue to be content-driven and document-oriented. Most Web applications, besides the increasing support to functionalities and services, will continue aiming at showing and delivering information. This is a basic feature stemming from the early Web that is currently empowered by the Semantic Web initiative [4].
- Web applications are interactive, user-centred, hypermedia-based applications, where the user interface plays a central role; thus, Web applications will continue to be highly focused on the look and feel. Web interfaces might be easy to use, understand, and operate because thousand of users with different profiles and capabilities interact with them daily.
- The Web embodies a greater bond between art and science than that encountered in software applications. Aesthetic and visual features of Web development are not just a technical skill, but also a creative, artistic skill.
- Internationalisation and accessibility of content for users with various disabilities are real and challenging issues in Web applications.
- Searching and browsing are two basic functionalities used to find and explore documents and information content. These capabilities are inherited from hypermedia-based applications.

- Security is a central issue in transaction-oriented Web applications. Likewise, performance is also critical for many Web applications, although both are also critical features for traditional applications.
- The entire Web application, and its parts, are often evolutionary pieces of information.
- The medium where Web applications are hosted and delivered, is generally more unpredictable than the medium where traditional software applications run. For instance, unpredictability in bandwidth maintenance, or in server availability, can affect the perceived quality that users could have.
- Content privacy and intellectual property rights of materials are current issues too. They involve ethic, cultural, and legal aspects as well. Most of the time it is very difficult to establish legal boundaries due to the heterogeneity of legislation in different countries, or even worse, the absence of them.

Most of the above features make a Web application a particular artefact. However, like a software application, it also involves source and executable code, persistent structured data, and requirements, architecture, design, and testing specifications as well.

Therefore, we argue that the ISO quality framework introduced in previous sections is also applicable to a great extent to intermediate and final lifecycle Web products. A discussion of this statement follows, as well as how we could adapt specific particularities of Web quality requirements into quality models.

Like any software line production, the Web lifecycle involves different stages of its products, whether in early phases as inception and development, or in late phases as deployment, operation, and evolution. To assure the quality of products, we can plan to do it by evaluating and controlling the quality from intermediate products to final products. Thus, if we can apply to the general question the same ISO internal and external quality, and quality in use models, the natural answer is *yes* – we believe this does not need further explanation. However, to the more specific question of whether we can use the same six prescribed quality characteristics for internal and external quality, and the four characteristics for quality in use, our answer is *yes* for the latter, but some other considerations might be taken into account for the former.

In particular, as highlighted at the beginning of this section, the very nature of Web applications is a mixture of information (media) content, functionalities, and services. We argue that the six quality characteristics (i.e. *Usability*, *Functionality*, *Reliability*, *Efficiency*, *Portability*, and *Maintainability*) are not well suited (or they were not intended) to specify requirements for information quality. As Nielsen [19] writes regarding

Web content for informational Web applications: "*Ultimately, users visit your Web site for its contents. Everything else is just the backdrop.*" Hence, to follow the thread of our argument, the central issue is how we can specify and gauge the quality of Web information content from the internal and external quality perspectives.

Taking into account some contributions made in the area of information quality [1,7,8,15,17] we have primarily identified four major sub-concepts for the *Content* characteristic. The following categories can help to evaluate information quality requirements of Web applications:

- *Information accuracy.* This sub-characteristic addresses the very intrinsic nature of the information quality. It assumes that information has its own quality *per se*. Accuracy is the extent to which information is correct, unambiguous, authoritative (reputable), objective, and verifiable. If a particular piece of information is believed to be inaccurate, the Web site will likely be perceived as having little added value and will result in reduced visits.
- *Information suitability.* This sub-characteristic addresses the contextual nature of the information quality. It emphasises the importance of conveying the appropriate information for user-oriented goals and tasks. In other words, it highlights the quality requirement that content must be considered within the context of use and the intended audience. Therefore, suitability is the extent to which information is appropriate (appropriate coverage for the target audience), complete (relevant amount), concise (shorter is better), and current.
- *Accessibility.* This emphasises the importance of technical aspects of Web sites and applications in order to make Web content more accessible for users with various disabilities (see, for instance, the WAI initiative [27]).
- *Legal compliance.* This concerns the capability of the information product to adhere to standards, conventions, and legal norms related to contents and intellectual property rights.

Besides the above categories, sub-concepts of information structure and organisation should be addressed. Many of these sub-characteristics, such as global understandability,[4] learnability, and even internationalisation, can be related to the *Usability* characteristic.

On the other hand, other particular features of Web applications, such as search and navigation functionalities, can be specified in the *Functionality* sub-characteristics (e.g. are the basic and advanced searches suitable for

[4] implemented by mechanisms that help to understand quickly the structure and contents of the information space of a Web site like a table of contents, indexes, or a site map.

the end user, o are they tolerant of mis-spelled words and accurate in retrieving documents?). In the same way, we can represent link and page maturity attributes, or attributes to deficiencies due to browsers' compatibility, in the *Reliability* sub-characteristics.

As a consequence, in order to represent software and Web applications, quality information requirements accordingly, we propose to include the *Content* characteristic in the internal and external quality model of the ISO standard. A point worth mentioning is that in the spirit of the ISO 9126-1 standard it is stated that "*evaluating product quality in practice requires characteristics beyond the set at hand*"; and as far as the requirements for choosing the prescribed characteristics, an ISO excerpt recommended "*To form a set of not more than six to eight characteristics for reasons of clarity and handling*."

Finally, from the "quality in use" perspective, for the *Satisfaction* characteristic, specific items for evaluating the quality of content as well as items for navigation, aesthetics, functions, etc., can be included. In addition, for other quality in use characteristics such as *Effectiveness* and *Productivity*, specific user-oriented evaluation tasks that include performing actions with content and functions can be designed and tested.

4.3 Evaluating Web Quality using WebQEM

As introduced in Sect. 4.1, the Web currently plays a central role in diverse application domains for various types of organisations and even in the personal life of individuals. Its growing importance heightens concerns about Web processes being used for the development, maintenance, and evolution of Web applications, and about the evaluation methods being used for assuring Web quality, and ultimately argues for the systematic use of engineering models, methods, and tools. Therefore, we need sound evaluation methods that support efforts to meet quality requirements in new Web development projects and assess quality requirements in operational and evolutionary phases. It is true that one size does not fit all the needs and preferences, but an organisation might at least adopt a method or technique in order to judge the state of its quality, for improvement purposes. We argue that a method or technique is usually not enough to assess different information needs for diverse evaluation purposes.

In this section we present the Web Quality Evaluation Method (WebQEM) [24] as a model-centred evaluation method for the inspection category; that is, inspection of concepts, sub-concepts, and attributes stemming from a quality or quality in use model. We have used the WebQEM methodology since the late 1990s. The underlying WebQEM strategy is evaluator-driven by domain experts rather than user-driven; quantitative and

model-centred rather than qualitative and intuition-centred; and objective rather than subjective. Of course, a global quality evaluation (and eventual comparison), where many characteristics and attributes, metrics, and indicators intervene, cannot entirely avoid subjectivity. Next, a robust and flexible evaluation methodology must properly aggregate subjective and objective components controlled by experts.

The WebQEM process steps are grouped into four major technical phases that are now further described:

1. Quality Requirements Definition and Specification.
2. Elementary Measurement and Evaluation (both Design and Implementation Stages).
3. Global Evaluation (both Design and Implementation Stages).
4. Conclusion and Recommendations.

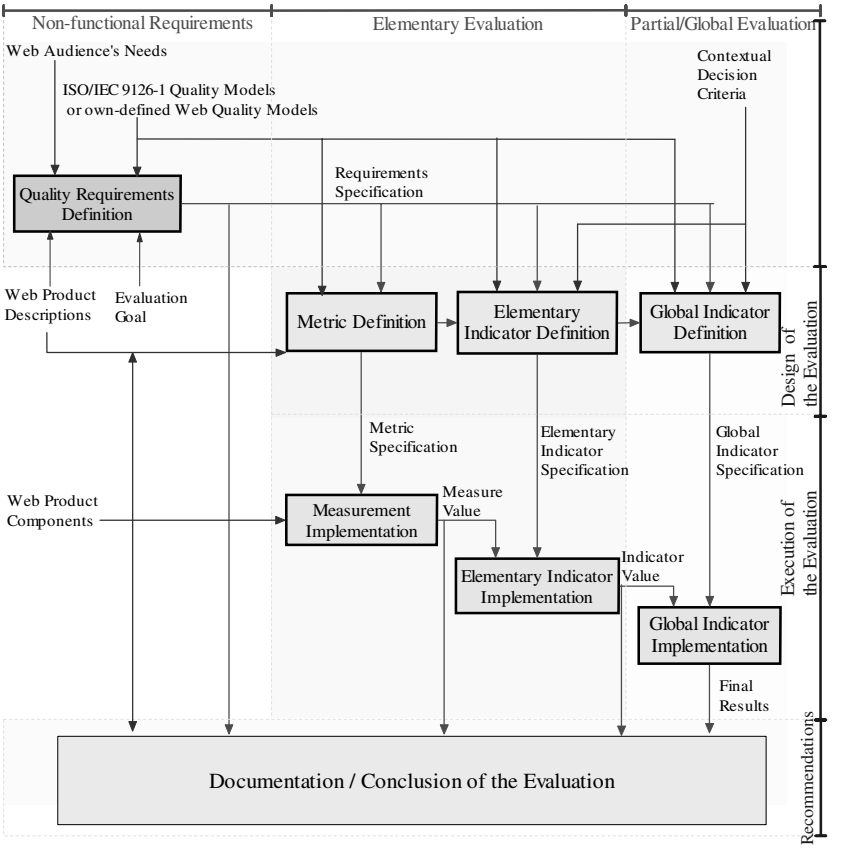

Fig. 4.5. The evaluation processes underlying the WebQEM methodology. The technical phases, main processes, and their inputs and outputs are represented

Figure 4.5 shows the evaluation process underlying the methodology, including the phases, main processes, inputs, and outputs. This model follows to some extent the ISO's process model for evaluators [11]. Next we give an overview of the major technical phases, and some used models.

4.3.1 Quality Requirements Definition and Specification

During the definition and specification of quality requirements, evaluators clarify the evaluation goals and the intended user's viewpoint. They select a quality model, for instance the ISO-prescribed characteristics, in addition to attributes customised to the Web domain. Next, they identify these components' relative importance to the intended audience and the extent of coverage required.

Once the domain and product descriptions, the agreed goals, and the selected user view (i.e. the explicit and implicit user needs) are defined, the necessary characteristics, sub-characteristics, and attributes can be specified in a quality requirement tree (such as that shown in Figs. 4.5 and 4.9). This phase yields a quality requirement specification document.

4.3.2 Elementary Measurement and Evaluation

The elementary measurement and evaluation phase defines two major stages (see Fig. 4.5): elementary evaluation design and execution (implementation). Regarding the elementary evaluation design, we further identify two main processes: (a) *metric definition* and (b) *elementary indicator definition*.

In our previous work [16,25], we have represented the conceptual domain of metrics and indicators from an ontological viewpoint. The conceptual framework of metrics and indicators, which was based as much as possible on the concepts of various ISO standards [12,14], can be useful to support different quality assurance processes, methods, and tools. That is the case for the WebQEM methodology and its supporting tool (WebQEM_Tool [23]), which are based on this framework.

As shown in Fig. 4.6, each attribute can be quantified by one or more metrics. For *the metric definition process* we should select just a metric for each attribute of the quality requirement tree, given a specific measurement project.

The metric contains the definition of the selected measurement and/or calculation method and scale. The metric m represents the mapping m: $A \rightarrow X$, where A is an empirical attribute of an entity (the empirical world), X the variable to which categorical or numerical values can be assigned

(the formal world), and the arrow denotes a mapping. In order to perform this mapping a sound and precise definition of measurement activity is needed by specifying explicitly the metric's method and scale (see Fig. 4.6). We can apply an objective or subjective measurement method for direct metrics, and we can perform a calculation method for indirect metrics; that is, when an equation intervenes.

To illustrate this, we examine the following direct metrics, taken from the example shown in Fig. 4.2:

1) *Internal Broken Links Count* (#IBL, for short),
2) *External Broken Links Count* (#EBL), and
3) *Invalid Links Count* (#IL).

In case we need a ratio or percentage, with regard to the *Total of Links Count* (#TL), the next indirect metrics can be defined:

4) %IBL = (#IBL / #TL) * 100, and so forth to
5) %EBL; and
6) %IL.

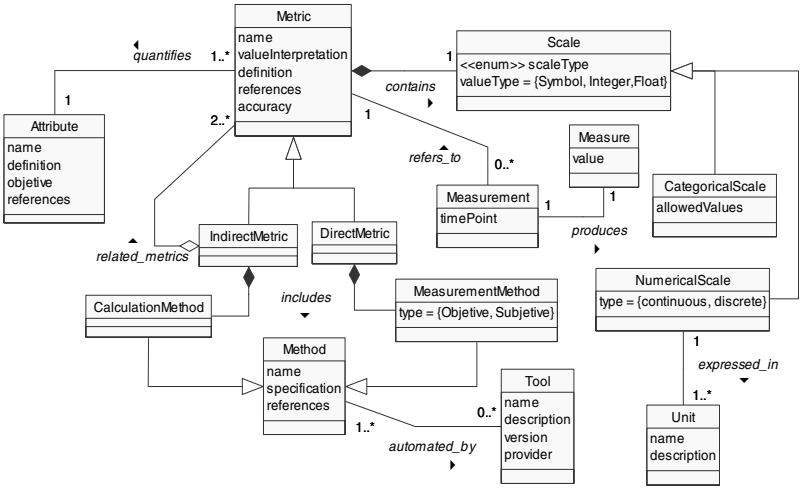

Fig. 4.6. Main terms and relationships with the metric concept

The scale type for the direct metrics presented above is absolute, represented by a numerical scale with integer value type. For the direct metrics 1) and 2), a specific objective measurement method can be applied (e.g. a recursive algorithm that counts each 404 HTTP status code). In addition, to automate the method, a software tool can be utilised; conversely, for the direct metric 3), it is harder to find a tool to automate it. On the other hand,

for the indirect metrics 4), 5), and 6), we can use a calculation method in order to perform the specified equation.

However, because the value of a particular metric will not represent the elementary requirement's satisfaction level, we need to define a new mapping that will yield an elementary indicator value.

In [16,25] the indicator term is stated as:

> *"the defined calculation method and scale in addition to the model and decision criteria in order to provide an estimate or evaluation of a calculable concept with respect to defined information needs."*

In particular, we define an elementary indicator as that which does not depend upon other indicators to evaluate or estimate a concept at a lower level of abstraction (e.g. for associated attributes to a concept model); in addition, we define a partial or global indicator as that which is derived from other indicators to evaluate or estimate a concept at a higher level of abstraction (i.e. for sub-characteristics and characteristics). Therefore, the elementary indicator represents a new mapping coming from the interpretation of the metric's value of an attribute (the formal world) into the new variable to which categorical or numerical values can be assigned (the new formal world). In order to perform this mapping, a model and decision criterion for a specific user information need is considered. Figure 4.7 represents these concepts.

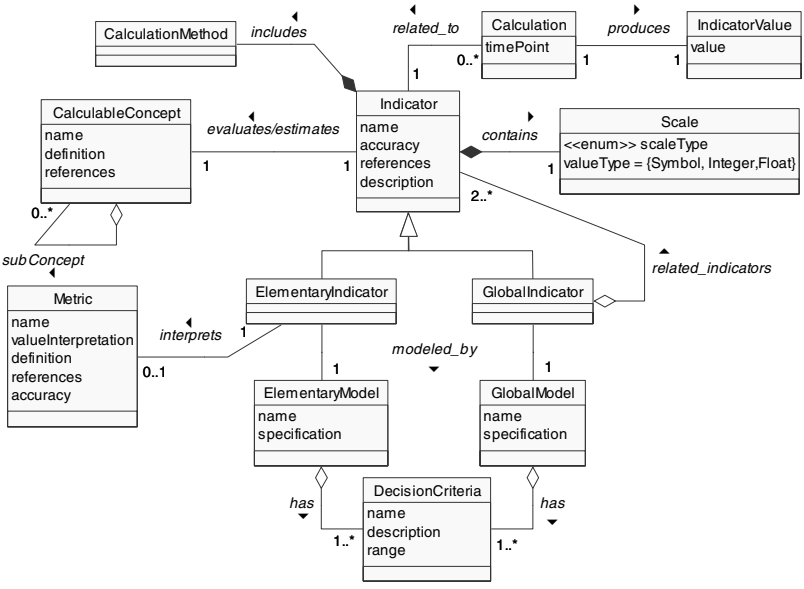

Fig. 4.7. Main terms and relationships with the indicator concept

Hence, an elementary indicator for each attribute of the concept model can be defined. To the 1.1 attribute of Fig. 4.2, the name of the elementary indicator can be for example *Internal Broken Links Preference Level* (IBL_P). The specification of the elementary model can look like this:

IBL_P = 100% if %IBL = 0; IBL_P = 0% if %IBL >= X_{max}
otherwise IBL_P=((X_{max} – %IBL) / X_{max}) * 100
if 0 < %IBL < X_{max} where X_{max} is some agreed upper threshold

The decision criteria that a model of an indicator may have are the agreed acceptability levels in a given scale; for instance, it is *unsatisfactory* if the range is 0 to 40%; *marginal* if it is greater than 40% and less than or equal than 60%; otherwise, *satisfactory*.

One fact worth mentioning is that the selected metrics are useful for a measurement process, as long as the selected indicators are useful for an evaluation process. Indicators are ultimately the foundation for interpretation of information needs and decision-making. Finally, Fig. 4.5 depicts the execution stage for the specified metrics and elementary indicators.

4.3.3 Global Evaluation

The global evaluation phase has two major stages: design and execution of the partial and global quality evaluation.

Regarding the global evaluation design, we identify the definition process of partial and global indicators. In this process, an aggregation and scoring model, and decision criteria, must be selected. The quantitative aggregation and scoring models aim at making the evaluation process well structured, objective, and comprehensible to evaluators. At least two types of models exist: those based on linear additive scoring models [6[, and those based on non-linear multi-criteria scoring models [5], where different attributes and characteristic relationships can be designed. Both use weights to consider an indicator's relative importance. For example, if our procedure is based on a linear additive scoring model, the aggregation and computing of partial/global indicators (P/GI), considering relative weights (W), is based on the following equation:

$$P/GI = (W_1\ EI_1 + W_2\ EI_2 + ... + W_m\ EI_m) \quad (4.1)$$

such that, if the elementary indicator (EI) is on a percentage scale, the following holds: $0 <= EI_i <= 100$.

Also the sum of weights for an aggregation block, or group, must fulfil:

$$(W_1 + W_2 + \ldots + W_m) = 1; \quad \text{if } W_i > 0; \text{ to } i = 1 \ldots m \quad (4.2)$$

where m is the number of sub-concepts at the same level in the aggregation block's tree.

The basic arithmetic aggregation operator for inputs is the plus (+) connector. We cannot use Equation 4.1 to model input simultaneity, or replaceability, among other limitations, as we discuss later.

Therefore, once we have selected a scoring model, the aggregation process follows the hierarchical structure as defined in the quality or quality in use requirement tree (see Fig 4.4), from bottom to top. Applying a stepwise aggregation mechanism, we obtain a global schema. This model lets us compute partial and global indicators in the execution stage. The global quality and 'quality in use' indicator ultimately represents the global degree of satisfaction in meeting the stated requirements, from a user's viewpoint.

4.3.4 Conclusions and Recommendations

The conclusion of the evaluation comprises documenting Web product components, the specification of quality requirements, metrics, indicators, elementary and global models, and decision criteria; and also it records measures and elementary, partial, and global indicator values. Requesters and evaluators can then analyse and understand the assessed product's strengths and weaknesses with regard to established information needs, and suggest, and justify, recommendations.

4.3.5 Automating the Process using WebQEM_Tool

The evaluation and comparison processes require both methodological and technological support. We have developed a Web-based tool (WebQEM_Tool [23]) to support the administration of evaluation projects. It permits editing, relating non-functional requirements, and calculating indicators based on the two aggregation models previously presented. Next, by automatically or manually editing elementary indicators, WebQEM_Tool aggregates the elements to yield a schema and calculates a global quality indicator for each application. This allows evaluators to assess and compare a Web product's quality to quality in use. WebQEM_Tool relies on a Web-based hyperdocument model that supports traceability of evaluation projects. It shows evaluation results using linked pages with textual, tabular, and graphical information, and dynamically generates pages with these results, obtained from tables stored in the data layer.

Currently, we are implementing a more robust measurement and evaluation framework, so-called INCAMI (Information Need, Concept model, Attribute, Metric, and Indicator). Its foundation lies in the ontological

specification of metrics and indicators [16,25]. The Web-based tool related to the INCAMI framework is called INCAMI_Tool.

4.4 Case Study: Evaluating the Quality of Two Web Applications

We have used WebQEM to evaluate the quality of Web applications in several domains, which is documented elsewhere [3,21,22]. We discuss here its application in an e-business domain.

4.4.1 External Quality Requirements

Many potential attributes, both general and domain-specific, can contribute to the quality of a Web application. However, an evaluation must be focused, and purpose-oriented for a real information need. Let us establish that the purpose is to understand and compare the external quality of the shopping cart component of two typical e-stores, from a general visitor's viewpoint, in order to incorporate the best features in a new e-bookstore development project. To this end, we chose a successful international application – Amazon (www.amazon.com/books), and a well-known regional application – Cuspide (www.cuspide.com.ar).

Figure 4.8 shows a screenshot of Cuspide's shopping cart page with several highlighted attributes, which intervene in the quality requirements tree of Fig. 4.9. For the definition of the external quality requirements, we considered four main characteristics: *Usability* (1), *Functionality* (2), *Content* (3), and *Reliability* (4), and 32 attributes related to them (see Fig. 4.9). For instance, the *Usability* characteristic splits into sub-characteristics, such as *understandability* (1.1), *learnability* (1.2), *operability* (1.3), and *attractiveness* (1.4). We also consider another two separate characteristics: *Functionality* and *Content*. *Functionality* is decomposed into *function suitability* (2.1) and *accuracy* (2.2). *Content* is decomposed into *information suitability* (3.1) and *content accessibility* (3.2). As the reader can observe (see Fig. 4.9), we relate five measurable attributes to the function suitability sub-characteristic, and three to the function accuracy. In the latter sub-characteristic, we mainly consider precision attributes to recalculate values, after making supported edit operations.

On the other hand, as mentioned in Sect. 4.2.4, information suitability stresses the contextual nature of the information quality. It emphasises the importance of conveying the appropriate information for user-oriented goals and tasks.

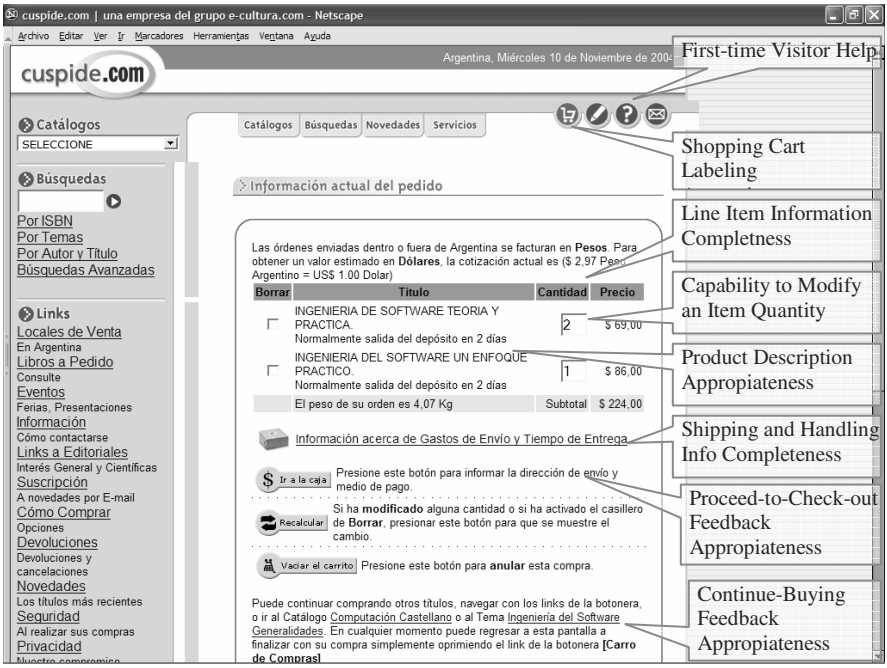

Fig. 4.8. A screenshot of Cuspide's shopping cart page with several attributes

INCAMI_Tool records all the information for an evaluation project. Besides the project data itself, it also saves to the *InformationNeed* class (see Fig. 4.1) the purpose, user viewpoint, and context description metadata; for the *CalculableConcept* and *Attribute* classes, it saves all the names, and definitions, respectively. The *ConceptModel* class permits one to instantiate a specific model, i.e. the external quality model in our case, allowing evaluators to edit and relate specific concepts, sub-concepts, and attributes. The resulting model is similar to that in Fig. 4.9.

4.4.2 Designing and Executing the Elementary Evaluation

As mentioned in Sect. 4.3.2, the evaluators should design, for each measurable attribute of the instantiated external quality model, the basis for the elementary evaluation process, by defining each specific metric and elementary indicator accordingly.

In the design phase we record all the information for the selected metrics and indicators, regarding the conceptual schema of *Metric* and *Elementary Indicator* classes shown in Figs. 4.6 and 4.7, respectively.

1. Usability
- 1.1. Understandability
  - *1.1.1. Shopping cart icon/label ease to be recognized*
  - *1.1.2. Shopping cart labeling appropriateness*
- 1.2. Learnability
  - *1.2.1. Shopping cart help (for first-time visitor)*
- 1.3. Operability
  - *1.3.1. Shopping cart control permanence*
  - *1.3.2. Shopping cart control stability*
  - *1.3.3. Steady behaviour of the shopping cart control*
  - *1.3.4. Steady behaviour of other related controls*
- 1.4. Attractiveness
  - *1.4.1. Color style uniformity (links, text, etc.)*
  - *1.4.2. Aesthetic preference*

2. Functionality
- 2.1. Function Suitability
  - *2.1.1. Capability to add items from anywhere*
  - *2.1.2. Capability to delete items*
  - *2.1.3. Capability to modify an item quantity*
  - *2.1.4. Capability to show totals by performed changes*
  - *2.1.5. Capability to save items for later/move to cart*
- 2.2. Function Accuracy
  - *2.2.1. Precision to recalculate after adding an item*
  - *2.2.2. Precision to recalculate after deleting items*
  - *2.2.3. Precision to recalculate after modifying an item quantity*

3. Content
- 3.1. Information Suitability
  - 3.1.1. Shopping Cart Basic Information
    - *3.1.1.1. Line item information completeness*
    - *3.1.1.2. Product description appropriateness*
  - 3.1.2. Shopping Cart Contextual Information
    - *3.1.2.1. Purchase Policies Related Information*
      - *3.1.2.1.1. Shipping and handling costs information completeness*
      - *3.1.2.1.2. Applicable taxes information completeness*
      - *3.1.2.1.3. Return policy information completeness*
    - *3.1.2.2. Continue-buying feedback appropriateness*
    - *3.1.2.3. Proceed-to-check-out feedback appropriateness*
- 3.2. Content Accessibility
  - 3.2.1. Readability by Deactivating the Browser Image Feature
    - *3.2.1.1. Image title availability*
    - *3.2.1.2. Image title readability*
  - 3.2.2. Support for text-only version

4. Reliability
4.1. Nondeficiency (Maturity)
4.1.1. Link Errors or Drawbacks
4.1.1.1. Broken links
4.1.1.2. Invalid links
4.1.1.3. Reflective links
4.1.2. Miscellaneous Deficiencies
4.1.2.1. Deficiencies or unexpected results dependent on browsers
4.1.2.2. Deficiencies or unexpected results independent on browsers

Fig. 4.9. Specifying the external quality requirements tree of the shopping cart component for a general visitor standpoint

Table 4.3. Summary of elementary indicators' values of the shopping cart of both applications

Code	Attribute name	Amazon	Cuspide
2.1.1	Capability to add items from anywhere	50.0	50.0
2.1.2	Capability to delete items	66.0	100.0
2.1.3	Capability to modify an item quantity	100.0	100.0
2.1.4	Capability to show totals by performed changes	66.0	66.0
2.1.5	Capability to save items for later/move to cart	100.0	0.0
3.1.1.1	Line item information completeness	100.0	33.0
3.1.1.2	Product description appropriateness	100.0	30.0
3.1.2.1.1	Shipping and handling costs information completeness	100.0	100.0
3.1.2.1.2	Applicable taxes information completeness	100.0	100.0
3.1.2.1.3	Return policy information completeness	100.0	66.0
3.1.2.2	Continue-buying feedback appropriateness	100.0	60.0
3.1.2.3	Proceed-to-check-out feedback appropriateness	100.0	100.0
3.2.1.1	Image title availability	50.0	50.0
3.2.1.2	Image title readability	100.0	50.0
3.2.2	Support for text-only version	0.0	0.0
4.1.1.1	Broken links	100.0	100.0
4.1.1.2	Invalid links	100.0	100.0
4.1.1.3	Reflective links	50.0	50.0
4.1.2.1	Deficiencies or unexpected results dependent on browsers	100.0	66.0
4.1.2.2	Deficiencies or unexpected results independent of browsers	30.0	30.0

In addition, in the execution phase, we record for the *Measurement* and *Calculation* classes' instances the yielded final values for each metric and indicator. Table 4.3 contains calculated elementary indicators' values for the shopping cart component of Amazon and Cuspide. The data collection for the measurement activity was performed from 15 to 20 November 2004.

Once evaluators have designed and implemented the elementary evaluation, they should consider not only each attribute's relative importance, but also whether the attribute (or sub-characteristic) is mandatory, alternative, or neutral. For this task, we need a robust aggregation and scoring model, described next.

4.4.3 Designing and Executing the Partial/Global Evaluation

The design and execution of the partial/global evaluation represents a phase where we select and apply an aggregation and scoring model (see Fig. 4.5). Arithmetic or logic operators will then relate the hierarchically grouped attributes, sub-characteristics, and characteristics accordingly.

As mentioned earlier, we can use a linear additive or a non-linear multi-criteria scoring model (or even others). We cannot use the additive scoring model to model input simultaneity (an *and* relationship among inputs) or replaceability (an *or* relationship), because it cannot express, for example, simultaneous satisfaction of several requirements as inputs. Additivity assumes that insufficient presence of a specific attribute (input) can always be compensated by sufficient presence of any other attribute. Furthermore, additive models cannot model mandatory requirements; that is, a necessary attribute's or sub-characteristic's total absence cannot be compensated by others' presence.

A non-linear multi-criteria scoring model lets us deal with simultaneity, neutrality, replaceability, and other input relationships by using aggregation operators based on the weighted power means mathematical model. This model, called Logic Scoring of Preference [5](LSP), is a generalisation of the additive scoring model, and can be expressed as follows:

$$P/Gl(r) = (W_1 El_1^r + W_2 El_2^r + \cdots + W_m El_m^r)^{\frac{1}{r}} \tag{4.3}$$

where

$$-\infty \le r \le +\infty; \quad P/Gl(-\infty) = \min(El_1, El_{2,} \ldots El_m); \quad \text{and}$$

$$P/Gl(+\infty) = \max(El_1, El_2, \ldots, El_m)$$

The power r is a parameter selected to achieve the desired logical relationship and polarisation intensity of the aggregation function. If $P/GI(r)$ is

closer to the minimum, such a criterion specifies the requirement for input simultaneity. If it is closer to the maximum, it specifies the requirement for input replaceability. Equation 4.3 is additive when $r = 1$, which models the neutrality relationship; that is, the formula remains the same as in the first additive model. Equation 4.3 is supra-additive for $r > 1$, which models input disjunction or replaceability. And it is sub-additive for $r < 1$ (with $r <> 0$), which models input conjunction or simultaneity.

For our case study we selected this last model and used a 17-level approach of conjunction–disjunction operators, as defined by Dujmovic [5]. Each operator in the model corresponds to a particular value of the r parameter. When $r = 1$ the operator is tagged with A (or the + sign). The C or conjunctive operators range from weak (C–) to strong (C+) quasi-conjunction functions; that is, from decreasing r values, starting from $r < 1$.

In general, the conjunctive operators imply that low-quality input indicators can never be well compensated by a high quality of some other input to output a high-quality indicator (in other words, a chain is as strong as its weakest link). Conversely, disjunctive operators (D operators) imply that low-quality input indicators can always be compensated by a high quality of some other input. Designing an LSP aggregation schema requires answering the following key basic questions (which are part of the *Global Indicator Definition* task in Fig. 4.5):

- What is the relationship between this group of related attributes and sub-characteristics: conjunctive, disjunctive, or neutral? (For instance, when modelling the attributes' relationship for the *Function Suitability* (2.1) sub-characteristic, we can agree that they are neutral or independent of each other.)
- What is the level of intensity of the logic operator, from a weak to strong conjunctive or disjunctive polarisation?
- What is the relative importance or weight of each element in the aggregation block or group?

WebQEM_Tool (which is being integrated into INCAMI_Tool) lets evaluators select the aggregation and scoring model. When using the additive scoring model, the aggregation operator is A for all tree aggregation blocks. If evaluators select the LSP model, they must indicate the operator for each group.

Figure 4.10 shows a partial view of the enacted schema for Amazon.com, as generated by our tool.

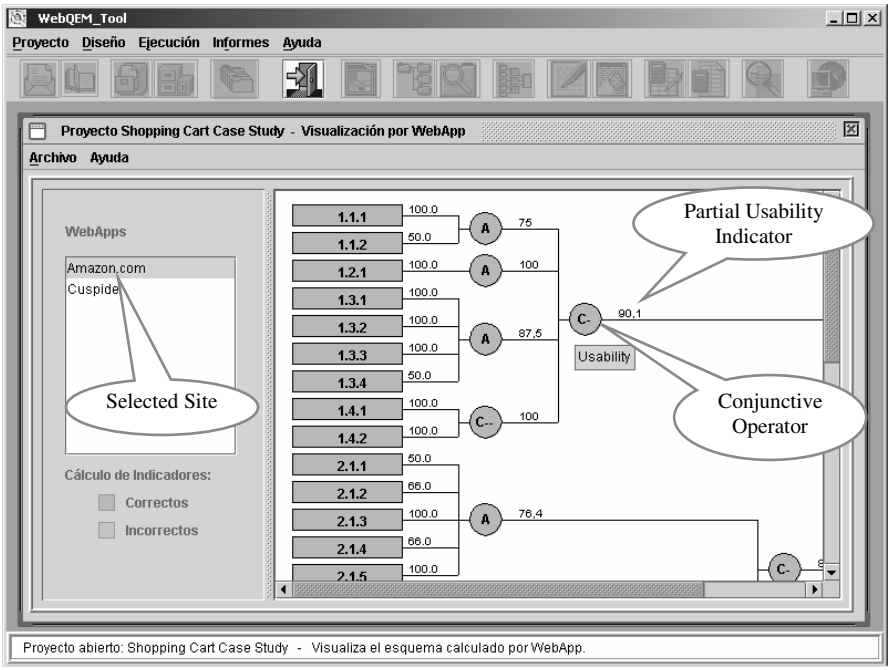

Fig. 4.10. Once the weights and operators were agreed and the schema checked, WebQEM_Tool yields partial and global indicators as highlighted in the right-hand pane

4.4.4 Analysis and Recommendations

Once we have performed the final execution of the evaluation, decision-makers can analyse the results and draw conclusions and recommendations.

As stated in Sect. 4.4.1, one of the primary goals of this study is the understanding and comparison of the current level of fulfilment of required external quality characteristics and attributes (see Fig. 4.9) for the shopping cart of two typical e-commerce applications, from a general visitor's standpoint. In addition, the best features of both shopping carts can be incorporated in a new e-bookstore development project. The underlying assumption of this study is that at the level of characteristics at least they are within the satisfactory acceptability range.

Table 4.4 shows the final values for the *Usability*, *Functionality*, *Content*, and *Reliability* characteristics, and the global quality indicator to both the Amazon and Cuspide shopping carts. The quality bars in Fig. 4.11 indicate the acceptability ranges and the quality level each shopping cart has reached. Amazon scored a higher quality level (84.32%) than Cuspide (65.73%). We suggest that scores between 40% and 60% (marginal acceptance) indicate

the need for improvement. An unsatisfactory rating, obtained by a score below 40%, means that improvements must be made very soon, so taking high priority. A score above 60% indicates a satisfactory quality.

Table 4.4. Summary of partial and global indicators' values of the Amazon.com and Cuspide.com shopping carts

Code	Characteristic/Subcharacteristic name	Amazon	Cuspide
	External Quality Indicator	84.32	65.73
1	Usability	90.1	90.1
1.1	Understandability	75.00	75.00
1.2	Learnability	100.00	100.00
1.3	Operability	87.50	87.50
1.4	Attractiveness	100.00	100.00
2	Functionality	87.61	80.05
2.1	Function Suitability	76.40	63.20
2.2	Function Accuracy	100.00	100.00
3	Content	81.61	45.11
3.1	Information Suitability	100.00	47.30
3.1.1	Shopping Cart Basic Information	100.00	31.47
3.1.2	Shopping Cart Contextual Information	100.00	81.17
3.1.2.1	Purchase Policies Related Information	100.00	88.68
3.2	Content Accessibility	56.79	41.91
3.2.1	Readability by Deactivating the Browser Image Feature	67.75	50.00
4	Reliability	75.34	67.61
4.1	Nondeficiency (Maturity)	75.34	67.61
4.1.1	Link Errors or Drawbacks	94.35	94.35
4.1.2	Miscellaneous Deficiencies	58.00	44.40

Looking at the *Usability* and *Functionality* characteristics we see similar scores in both applications, so that we can emulate such attributes in a new development project. We can just highlight that the *Capability to save items for later/move to cart* (2.1.5) desirable attribute is absent in Cuspide, and the *Capability to delete items* (2.1.2) attribute is more suitable in Cuspide, as users can delete several items at once from the shopping cart (see the elementary indicators in Table 4.3).

Nonetheless, the greatest score differences can be observed in the *Content* characteristic (see Tables 4.3 and 4.4). Cuspide must plan changes in the *Shopping Cart Basic Information* sub-characteristic mainly in the 3.1.1.1 and 3.1.1.2 attributes. For instance, the *Line item information completeness* has to have at least the author description besides the title description, because when users add another item with the same starting title

(e.g. Software Engineering …) they cannot, looking at the shopping cart, determine who is the author of each title. Even worse, users might navigate back to find out who the authors are because they have no link to a detailed product description.

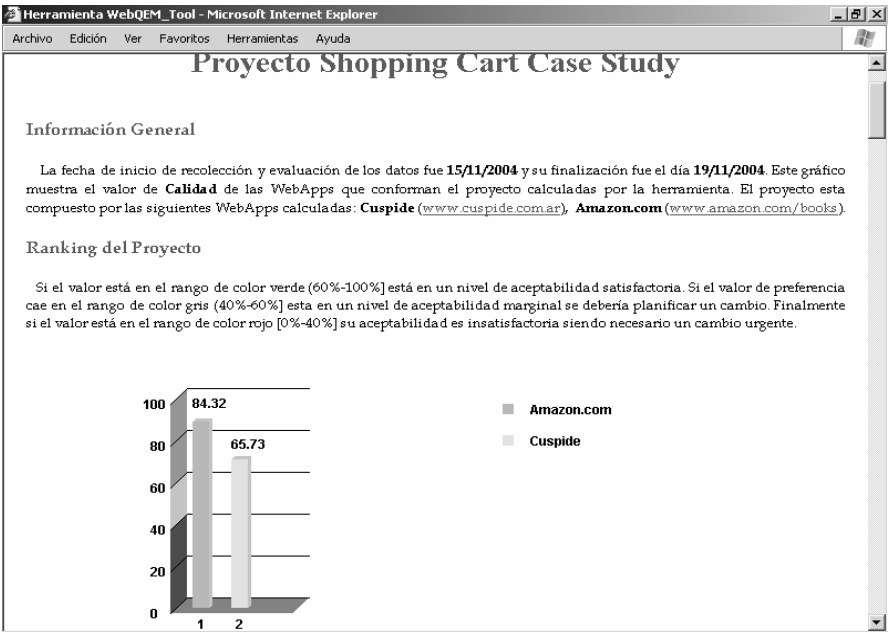

Fig. 4.11. WebQEM_Tool shows diverse information types (as textual, tabular, and graphical). The graph depicts the final shopping cart ranking

With regard to the *Content Accessibility* sub-characteristic, we may not emulate both applications because they are in the marginal acceptability level. On the other hand, we found *Deficiencies or unexpected results independent of browsers* (4.1.2.2) in both shopping carts; that is, there is no input validation in the quantity field so that a user can type decimal numbers or alphanumeric inputs, which can lead to unexpected outcomes.

Finally, we observe that the state of the art of the shopping cart quality on typical e-bookstores, from the visitor's point of view, is rather high, but the wish list is not empty because of some poorly designed or absent attributes. Notice that elementary, partial, and global indicators reflect the results of these specific requirements for this specific audience and should not be regarded as generalised rankings. Moreover, results themselves from a case study are seldom intended to be interpreted as generalisations that can be applicable to any other applications.

4.5 Concluding Remarks

Developing successful Web applications with economic and quality issues in mind requires broad perspectives and the incorporation of a number of principles, models, methods, and techniques from diverse disciplines such as information systems, computer science, hypertext, graphic design, information structuring, knowledge management, and ultimately software engineering as well. Web engineering is therefore an amalgamation of many disciplines, but with its own challenges. It has a very short history compared with other engineering disciplines, but is rapidly evolving. Like any other engineering science, Web engineering is concerned with the establishment and use of sound scientific, engineering, and management principles, and disciplined and systematic approaches to the successful development, deployment, maintenance, and evolution of Web sites and applications within budgetary, calendar, and quality constraints.

As mentioned above, the quality of an entity is easy to recognise but hard to define and evaluate, and sometimes costly to incorporate in the end product. In this chapter we have discussed what quality in general, and what Web quality in particular, is about. We adhere to the ISO approaches of quality: that is, internal quality, external quality, and quality in use. Because quality is not achieved at the end of a development without a carefully designed quality assurance strategy in the early stages, we argue that the three perspectives of quality per se have their own relative importance. However, we also adhere to the saying "*Quality in use is (or at least should be) the objective, software product quality is the means of achieving it*" [2].

We have highlighted that the very nature of Web applications is a mixture of information content, functionalities, and services. Next, we proposed to include *Content* as an extra characteristic in the internal and external quality models to the ISO 9126-1 standard (see Sect. 4.2.4).

On the other hand, regarding Web engineering evaluation approaches, we posed the need for counting with sound evaluation frameworks, methods, and techniques that support efforts to meet quality requirements at different stages of a Web project. We also stated that very often a method or technique is not enough to assess different information needs for diverse evaluation purposes. In this context, we presented WebQEM as a quantitative evaluation method for the inspection category whose underlying strategy is evaluator-driven by domain experts rather than user-driven; quantitative and model-centred rather than qualitative and intuition-centred; and objective rather than subjective. We are aware that a global quality evaluation (and eventual comparison), where many characteristics and attributes, metrics, and indicators intervene, cannot entirely avoid subjectivity. Then a robust and flexible evaluation methodology must properly aggregate subjective and objective components controlled by experts.

In order to illustrate WebQEM and its applicability, we conducted an e-business case study by evaluating the external quality of the shopping cart components of Amazon and Cuspide sites, taking into account a general visitor's standpoint. As a matter of fact, the data collection and evaluation were made by two expert evaluators working simultaneously. Note the important difference between evaluating external quality and quality in use. The former generally involves only experts and the latter always involves real end users. The advantage of using expert evaluation without extensive user involvement is minimising costs, time, and potential misinterpretation of questions (i.e. end users may sometimes interpret instructions and questionnaire items in a different way than they were intended to). The choice of whether to involve end users or not should be carefully planned and justified. Ultimately, without end user participation, it is unthinkable to conduct task testing in a real context of use. Nielsen indicates that commonly up to five subjects in the testing process for a given audience produce meaningful results minimizing costs: "*The best results come from testing no more than 5 users and running as many small tests as you can afford*" [19].

As a last remark, we are currently implementing a more robust measurement and evaluation framework called INCAMI which stands for Information Need, Concept model, Attribute, Metric, and Indicator; its foundation lies in the ontological specification of metrics and indicators [24]. WebQEM_Tool, which is part of this measurement and evaluation framework, allows consistently saving of not only metadata of metrics and indicators but also data for specific evaluation projects. Inter- and intra-project analyses and comparisons can now be performed in a consistent way. This applied research is thoroughly discussed in a follow-up manuscript.

Acknowledgements

This research is supported by the UNLPam-09/F022 project, Argentina. Gustavo Rossi has been partially funded by Secyt's project PICT No 13623.

References

1 Alexander J, Tate M (1999) Web Wisdom: How to Evaluate and Create Information Quality on the Web. Lawrence Erlbaum, Hillsdate, NJ

2 Bevan N (1999) Quality in Use: Meeting User Needs for Quality. Journal of Systems and Software, 49(1):89–96

3 Covella G, Olsina L (2002) Specifying Quality Attributes for Sites with E-Learning Functionality. In: Proceedings of the Ibero American Conference on Web Engineering (ICWE, 02), Santa Fe, Argentina, pp 154–167

4 Davies J, Fensel D, Van Harmelen F (2003) Towards the Semantic Web: Ontology-driven Knowledge Management. John Willey & Sons

5 Dujmovic J (1996) A Method for Evaluation and Selection of Complex Hardware and Software Systems. In: Proceedings of the 22nd International Conference for the Resource Management and Performance Evaluation of Enterprise CS, CMG 96 Proceedings, 1, pp 368–378

6 Gilb T (1976) Software Metrics. Chartwell-Bratt, Cambridge, MA

7 Herrera-Viedma E, Peis E (2003) Evaluating the Informative Quality of Documents in SGML Format from Judgements by Means of Fuzzy Linguistic Techniques Based on Computing with Words. J Information Processing. & Management 39(2):233–249

8 Huang K, Lee YW, Wang RY (1999) Quality Information and Knowledge. Prentice Hall, Englewood Cliffs, NJ

9 ISO/IEC 9126 (1991) Information technology – Software product evaluation – Quality characteristics and guidelines for their use

10 ISO 9241–11 (1998) Ergonomic requirements for office work with visual display terminals (VDT)s – Part 11 Guidance on Usability

11 ISO/IEC 14598–5 (1998) Information technology – Software product evaluation – Part 5: Process for evaluators

12 ISO/IEC 14598–1 (1999) Information technology – Software product evaluation – Part 1: General Overview

13 ISO/IEC 9126–1 (2001) Software Engineering – Product Quality – Part 1: Quality Model

14 ISO/IEC 15939 (2002) Software Engineering – Software Measurement Process

15 Lee YW, Strong DM, Kahn BK, Wang RY (2002) AIMQ: A Methodology for Information Quality Assessment. Information & Management, 40(2):133–146

16 Martín M, Olsina L (2003) Towards an Ontology for Software Metrics and Indicators as the Foundation for a Cataloging Web System. In: Proceedings of the 1st Latin American Web Congress, Santiago de Chile, pp 103–113

17 Mich L, Franch M, Gaio L (2003) Evaluating and Designing the Quality of Web Sites. IEEE MultiMedia, 10(1):34–43

18 Murugesan S, Deshpande Y, Hansen S, Ginige A (2001) Web Engineering: A New Discipline for Development of Web-based Systems. In: Murugesan S, Deshpande Y (eds) Web Engineering: Managing University and Complexity of Web Application Development, LNCS 2016, Springer, Berlin,pp 3–13

19 Nielsen J (1995–2004) The Alertbox column, http://www.useit.com/alertbox/

20 Nielsen J, Molich R, Snyder C, Farrell S (2001) E-Commerce User Experience, NN Group

21 Olsina L, Godoy D, Lafuente G, Rossi G (1999) Assessing the Quality of Academic Web sites: a Case Study. New Review of Hypermedia and Multimedia, 5:81–103

22 Olsina L, Lafuente G, Rossi G (2000) E-commerce Site Evaluation: a Case Study. In: Proceedings of the 1st International Conference on Electronic Commerce and Web Technologies. LNCS 1875, Springer London, UK, pp 239–252

23 Olsina L, Papa MF, Souto ME, Rossi G (2001) Providing Automated Support for the Web Quality Evaluation Methodology. In: Proceedings of the 4th Workshop on Web Engineering, at the 10th International WWW Conference, Hong Kong, pp 1–11

24 Olsina L, Rossi G (2002) Measuring Web Application Quality with WebQEM. IEEE Multimedia, 9(4):20–29

25 Olsina L, Martín M (2004) Ontology for Software Metrics and Indicators. J of Web Engineering. 2(4):262–281

26 Powel TA (1998) Web Site Engineering: Beyond Web Page Design. Prentice Hall

27 WWW Consortium, Web Content Accessibility Guidelines 1.0, http://www.w3.org/TR/WAI-WEBCONTENT/ (accessed on 10th November 2004)

Authors' Biographies

Luis Olsina is an Associate Professor in the Engineering School at National University of La Pampa, Argentina, and heads the Software and Web Engineering R&D group (GIDISWeb). His research interests include Web engineering, particularly Web metrics and indicators, quantitative evaluation methods, and ontologies for the measurement and evaluation domain. He authored the WebQEM methodology. He earned a PhD in software engineering and an MSE from National University of La Plata, Argentina. In the last seven years, he has published over 50 refereed papers, and participated in numerous regional and international events both as programme committee chair and member. He is an IEEE Computer Society member.

Guillermo Covella is an Assistant Professor in the Engineering School at National University of La Pampa, Argentina. He is currently an MSE student in the Informatics School at National University of La Plata, developing his thesis on quality in use evaluation of web applications. His primary research interests are web quality and quality in use, specifically in the field of e-learning.

Gustavo Rossi is Full Professor at Universidad Nacional de La Plata, Argentina, and heads LIFIA, a computer science research lab in the College of Informatics. His research interests include context-awareness and Web design patterns and frameworks. He coauthored the Object-Oriented Hypermedia Design Method (OOHDM) and is currently working on the application of design patterns in context-aware software. He earned a PhD in Computer Science from Catholic University of Rio de Janeiro (PUC-Rio), Brazil. He is an ACM member and IEEE member.