A METRICS ECOSYSTEM FOR DESIGNING QUALITY E-COMMERCE SYSTEMS

Antonia Stefani

Dynamic Ambient Intelligent Sociotechnical Systems Group, Hellenic Open University, Patras, Greece

ABSTRACT

E-commerce has become a prime channel for doing commerce both globally and locally. It has gained the trust of buyers worldwide not only because of the supporting technologies but for the ever-increasing quality of service provided by most vendors as well. Quality is and will be a competitive advantage in e-commerce as competition on an international scale increases. Designing quality e-commerce software is one battle that needs to be won in this war and tools such as formal standards are key weapons. In this paper an eco-system of software quality metrics is presented based on the ISO25010 standard. These metrics are mapped to quality characteristics of the software and to facets of user-system interaction so as to provide designers and developers insights and guidelines on how produce quality e-commerce systems. A representative set of 29 new metrics for the quality characteristics of Functional Suitability and Usability is presented.

KEYWORDS

E-Commerce, Software Design, Software Quality, Metrics, ISO25010.

1. INTRODUCTION

The software industry is steadily moving towards the adoption of a software-as-a-service model. A factor that is increasingly gaining attention is the quality of service in terms of service availability, service functionality and most importantly unprecedented levels of user (buyer) experience [1]. The shift from traditional, COTS software to e-services has brought significant changes not only on how software is designed and developed but also on software-user and software-software interaction [2]. Underlying this shift is the use of technologies well beyond the classic hypertext/hypermedia model: HTML5, CSS, server-side scripting, VRML are some of the technologies introduced in the last years, literally re-invented e-commerce (as far as the buyer experience is concerned) [3]. An equally important change is the target audience extension: over the Internet, millions of users with heterogeneous needs can now be served anytime, everywhere. It still remains important, and actually it has become imperative, that the user-system interaction is performed in a way that vendors gain added value from it and that value is translated into a competitive advantage.

In this context, quality of service is not only limited to the ease-of-use or accessibility, although these are important parameters for the success of e-commerce software. Even well designed hyper-media / hypertext environments seem to respond well to such requirements. The interface is not the goal but the means by which software "transfers" its virtues to the user. Often, software that is not built to deliver the most to its users but implements an innovative business or research idea has short-term success. Many times the practice has shown that the medium/long-term

DOI:10.5121/ijcsit.2018.10207

success (acceptance by users) of software is based not so much on the idea (which can be copied by competitors) but also on the quality of the services it offers [4,6].

It is generally accepted that increasing the quality of a product increases its manufacturing cost [5]. Quality results are best achieved when formal procedures, best practises or knowledge from user experience feedback are incorporated from the very first stages of software development (analysis and design phases). Assessing software that has already been developed with the goal to improve it, usually means increasing costs directly relevant to the design changes that need to take place. Software standards such as ISO9126 and the latest ISO25000 series provide formal guidelines on the quality dimensions of software, albeit not specific ways on how to achieve them. Their contribution is significant because formal guidelines, although general, carry the knowledge of the information systems community within. Generality however useful (standards are applied to many domains and are technology independent), reduces practicality [24].

This rule also applies to e-commerce software. The construction of quality software that best meets all the ISO 25000 series guidelines is technically feasible but maybe economically unprofitable for many small vendors [2,6]. ISO standards follow a top-down hierarchy, organising the overall quality (of software in this particular case) in quality characteristics, which in turn are, correspond to quality sub-characteristics. One solution to the practicality issue is to use metrics, that is, quality measures of quality sub-characteristics that quantify quality of specific software components. Metrics can be used at the design phase of a software lifecycle to provide insights for targeted quality design, having in mind the significant restrictions of available assets during software development projects [7]. They can also be used after software development for upgrading and maintenance, processes that have gained increasing importance in service-oriented software. Furthermore, since e-commerce software provides various (and in many cases, interconnected) ways of communication with the users, a set of general quality guidelines would prove somewhat weak in terms of impact and practicality [21].

Based on the above observations, some of the key research problems of quality targeted e-commerce software design include the use of formal standards such as ISO for optimizing quality of service, how to increase the practicality with metrics that are context-specific and how can they be applied to the different software components [12]. These questions are difficult to answer for all kinds of web-based software. This paper focuses on these aspects for a particular and extremely important category of such applications: Business to Consumer E-Commerce applications. A new framework of metric - quality and quality characteristics of ISO 25010 process standard is proposed. It treats the subjectivity normally encountered in any software qualitative assessment method with emphasis on the user side. Research, following a rigorous methodology, resulted in categorization of metrics for B2C systems assessment further extending the work initially presented in [8]. The framework defines six key components of user interaction to e-commerce systems: Presentation, Navigation, Purchasing, Social, Interaction and Support. This work focuses primarily on metrics for the Functional Suitability and Usability quality characteristics of ISO25010 since these are not only the most frequent features assessed in existing products but amongst the most challenging and important targets of e-commerce system design.

Based on previous work described in [8] for the ISO9126 standard, a new metric ecosystem is presented based on ISO25000 series standard which enhances the key components of user interaction (from three to six). Each component includes a set of system features. Features define the functions and services that the system provides to the end user. The concept of quality also includes the concept of measuring using the appropriate metrics. Metrics typically provide a quantitative performance of the qualitative components that make up the system either by counting a numeric value or by rendering a value in a binary variable. In particular, the evaluation of the quality of web-based systems is done by using a set of web metrics that are appropriate to

quantify mainly, with the measurements, the characteristics of the system. In order to contribute to the research of quality-supported design, this work introduces only new metrics that can be used to assess the quality as perceived by the user specifically for customer-type e-commerce systems.

This paper is structured as follows: in section 2, a discussion on the state of the art of quality and formal assessment standards takes place. In section 3, a more in-depth examination of e-commerce design based on quality, sets the research challenges. Section 4, describes the basic features of the ISO25010 standard in which the metrics ecosystem is based. The eco-system itself is presented in section 5.

2. SOFTWARE QUALITY FOR SYSTEM DESIGN

Software development companies follow specific methods for designing new products, that generate added value using any combination of business, technology or even quality breakthroughs. Designing with quality in mind (e.g. speed or accuracy of response to specific business process) in relation to the competition may be standard practice. This tactic is tantamount to targeted quality software development since it facilitates economy of scale by devoting resources to empower certain qualitative features. The remaining features are held, initially, at a medium quality level. It achieves a reduction in development costs and limits the time-to –market parameter. ISO standards, and in particular ISO 25000, are suitable for micro-management of quality during the development or upgrading phase of the software [22]. In other words, it allows design and development teams to focus on specific features and sub-features of the external quality of software features that are likely to provide a competitive advantage. But what is software quality and how can it be achieved?

The term "Quality" seems to be self-evident. Its interpretation in the field of Computer Science is more difficult than it seems. In the field of Software Engineering, there are many aspects of what Quality is and how it can be achieved. The scientific (but also practical) use of the term is bounded by standards, i.e. commonly accepted definitions and instructions for use. The ISO (International Organization for Standardization) is an international structure responsible for drafting standards, including software quality standards. But ISO standards also include different interpretations, that are not mutually exclusive. The ISO 9000 Quality Assurance standard essentially defines that a software is as good as its specifications [9]. A different approach is given by ISO 8402 that defines quality as ‘the set of features (of the software) that allow it to satisfy declared or implied (user) needs’ [10]. Other interpretations of quality given by ISO standards are based on the existence or not of certain characteristics. Newer standards such as ISO 25010 define the characteristics that make up the quality of a software and organises it in quality dimensions [11].

Quality according to the above mentioned definitions highly depends on the satisfaction of user requirements. If these requirements are well defined and documented, quality could be approached satisfactorily through the features that should be designed or developed. In the case of a user-centric web-based software, such as e-commerce software, where there are a multitude of users with different needs, cultural backgrounds, perceptions of business commerce processes and software use, the exact definition of qualitative features is rather difficult. And not just the precise definition but often, a good approach is somewhat hypothetical. Garvin's definition of Quality defines it as the combination of software features that satisfy (as much as possible) a particular user [12]. An obvious problem with this type of approach is subjectivity. 'Opinions about the Quality of a software can be as different as the views of what is beautiful aesthetically' according to Bevan [13]. Thus, the quality of web-based software is strongly associated with users' perceptions of what is qualitative and therefore called "User-perceived Quality". It is worth noting

that many software development projects fail because specifications (and software quality) are determined by the development team's perceptions rather than by users. The result may satisfy the development team but not the users [24].

There is a more important reason for connecting quality and users. Each product (and software) is directly related to the purpose for which it is being build. For example, typically high-quality features of software designed for exclusive use on a personal computer are different from software designed for use on the Internet. Software-wide quality features are even more difficult to define. An on-line email client is equally used by a company's managerial staff for organizing their work and by the secretary for sending simple letters. Between the two cases, the requirements for some qualitative characteristics are the same, others differ. The software is the same. These observations have led to the definition of External Quality, according to which the centre of gravity is transferred from the evaluation of isolated software to its assessment based on certain conditions, i.e. in relation to defined needs of defined user groups [11]. One important question that arises is: How is the Quality of software related to how it is used by the user? The separation of External Quality from the use of software has led to the definition of the term Quality in Use. This type of quality refers to how defined users behave in order to achieve their goals by using the software in defined environments. It focuses more on the user's behaviour towards software rather than on the contrary (as does external quality). This term is embedded in the general term of Quality and was initially defined in ISO DIS 14598-1 [14]. Later it was included in the newer versions of ISO 9126 and its successor, ISO25000 series. Its difference with the external quality is slim but distinct [25]. In fact, Bevan in his article "Usability is Quality in Use" has set the foundation for the need to distinguish the two early on in the design process of software [13].

The definition and initial analysis of Quality features from ISO standards helps to design and develop better quality software systems because it organises qualitative properties in a formal way, easily communicated and understood by teams participating in all software development phases. Standards define and organise quality attributes but not the values of these properties: they report what properties quality software should have and not how it will be built so as to exhibit these qualities. Standards are therefore broad enough to preserve their correctness regardless of how the software is made. But this generality greatly reduces their practical value [15].

Besides of this general approach, another characteristic of e-commerce software is a source of heterogeneity in quality measurement. It consists of many, heterogeneous multiple sub-systems at the architectural level. The technologies used by these subsystems may be different as well as the philosophy with which they have been developed. Since the technical and operational characteristics of these subsystems are different, should their quality be addressed in different perspectives as well? In other words, accurate assessment of the quality of software, apart from its assessment as a whole, should include the evaluation of its individual components, which should be tailored to their characteristics? By reversing the question, is it practical (and how much) for the quality assessment methods to focus only on the whole and not evaluate (separately) the individual software features. What adjustments should be made to the ISO based assessment method to qualitatively assess the different technological and functional components of software so as to provide guidelines to designers of other systems or to maintenance engineers of the same system?

Continuing the debate on the practicality of software quality assessment, it would be an omission not to refer to the problem of quantifying its results. Evaluating software over standards such as ISO 25000 series (and those of ISO9126 where there is a larger corpus of evidence) provides results that do not help software engineers enough so as to develop or improve the software [22]. For example, evaluating software as of "moderate" quality in the domain of functionality (of all

services it provides to the user) does not provide necessary information such as: 'which services do not perform properly and why?' or 'what services have not been implemented at all?' and most importantly, 'how will we increase the quality of functionality?'. There has been a lot of debate in this area about how clear, in terms of guidance, it is a software model [16,18,21,24]. As mentioned above, a standard defines the presentation of certain qualitative attributes rather than how they will be constructed.

One characteristic example is the ISO 9126 usability directive, which essentially states that the software user interface should be capable of serving satisfactorily any type of user [25]. But how is such an interface to be built? Obviously, the directive barely helps a software engineer in the interface development process. This knowledge vacuum in the use of standards is normal (since standards are general and independent of technology by nature) albeit creates difficulties in their application. This fact has been early highlighted by the community of software engineers with articles in top engineering publications such as the Comm. of the ACM and Wired [17]. The rate of use of ISO 9126 has been proven to be small [18], but this does not reduce its value. It successor, ISO 25000 series follows the same philosophy. The solution to this problem (or an acceptable solution) is the use of tangible, measurable measures that offer application guidelines without being disconnected from the standard itself. In this way, information is not lost and practice is increased. ISO 9126 provides measures (quality metrics), which, although formally defined in the latest versions of the standard, refer to any type of software. Metrics as well as standards are non-application-specific. In recent years, an attempt has been made to designate metrics to evaluate the quality of web-based software with interesting results. But again, the number of different kinds of these applications maintains (to a lesser extent) the aforementioned problem [21,22].

3. ASSESSMENT AS A MEANS FOR BETTER SYSTEM DESIGN

E-commerce systems are quite complex as software. They are composed of many subsystems and can work on different parameters that affect their overall performance. These parameters consist of hardware (network infrastructure, server configuration, routers, etc.) as well as software (operating system, web server software, back-end application software, use of eternal applications and services) and especially software used by the user. A common mistake made in practice is to isolate the software from the environment in which it will work. This practice leads to erroneous qualitative assessments [19].

Business-to-consumer e-commerce software is also heterogeneous, both technologically and functionally. It is heterogeneous because different sub-systems use different technologies. For example, the interface can use HTML5 and order processing in the background programming language C #. Heterogeneity can be observed even within the same subsystem: the interface uses VRML for 3D imaging and DHTML for handling forms. Functional heterogeneity is also important. A B2C software is essentially an Information System (or part of the enterprise's broader IT system) and therefore has a process workflow. The functions are strictly defined to serve different user needs (the B2C software must also be strictly human-centred). Therefore, functions are strictly separated without overlapping: some functions serve the same purpose in a different way (e.g. electronic catalogues and search engines are different means of accessing information).

How can a software system so heterogeneous be evaluated? Certainly, its various components should be evaluated differently as mentioned in the previous section. E-commerce systems are extremely demanding in terms of user satisfaction: the range of requirements is large, while failure to meet user needs may lead to the rejection of the system from a large population. Qualitative assessment of such systems presents several difficulties. An enterprise-style e-

commerce system offers a set of services that use heterogeneous or rapidly evolving technologies, services that in turn can be evaluated through many different parameters. In addition to analysing a system in services and qualitative parameters, it is also necessary to be formal, i.e. to use an official evaluation model. The literature does not exhibit as many works as one would expect in this issue since there are not many models, methodologies or tools that can thoroughly assess an enterprise-client e-commerce system by identifying individual qualities and / or strengths of the company based on an official standard [19-22,23].

Based on the questions raised in the previous sections, the following research question is set: 'how can the qualitative and in-depth quality of an enterprise-to-business e-commerce system be assessed?'. Going one step further, the following additional question can be raised: 'In this context, how can targeted quality software design and development of e-commerce systems be supported?'.

One approach to answering the above question is to find solutions to the following issues: What needs to be evaluated? How should it be evaluated? What methods should be used for evaluation so as to help system design and/or maintenance? A technique is to design a framework of metrics mapped to quality and quality characteristics of ISO 25000 series standard. Its goal should be to treat the subjectivity normally encountered in the software qualitative assessment method with emphasis on the user side.

4. THE ISO25000 SERIES STANDARD

The structure of ISO standards is usually hierarchical. At the top of the hierarchical structure the qualitative features are placed. These are categories of qualitative components that are non-overlapping. Each feature contains (or is broken down into) a set of qualitative sub-features that is also non-overlapping. Due to the non-overlapping of the characteristics it is implied that the relation of characteristic to sub-characteristics is one to many. These two levels of the structure of a standard describe, in general terms, the qualitative components in which absolute values cannot be attributed. This is necessary to ensure the generality of the standards, i.e. their independence from specific techniques or technologies of implementation of the assessment object. Standards are usually accompanied by application instructions or usage examples. These are not part of their structure. It is customary to issue new standards that contain only specifications or guidelines for the implementation of other standards (containing quality models), management of standard implementation procedures or frameworks which, if properly adapted, can provide a basis for quality assessment systems or quality standards.

The software quality measurement model as defined in ISO25010 [11], defines the intrinsic properties of the software, which can be distinguished quantitatively or qualitatively. Quality attributes are intrinsic properties of software that contribute to quality. Quality attributes are categorized into one or more (sub) attributes. Quality characteristics are measured by applying a measurement method. A measurement method is a logical sequence of operations used to quantify a feature relative to a particular scale. The result of applying a measurement method is called quality measure element. Qualitative features and sub-features can be quantified by applying measurement functions to these elements. A function is essentially an algorithm used to combine elements. The result of applying a measurement function is called a quality measure. In this way, the quality measurement elements become quantified reflections of qualitative features and sub-features. More than one type of measure can be used to measure a feature or sub-feature. The philosophy of ISO standards is to use, in the same model, different assessment approaches depending on the life cycle stage in which a product is or in which part of the product is being evaluated. Each approach is based on specific, distinct features and sub-features.

ISO25000 - Software Quality Requirements and Evaluation (SQuaRE) is a new version of software quality standards. It was designed to replace models of the 9000 and 10000 series with the aim of homogenizing and eliminating overlaps. It is comprised of several sub-standards, as depicted in table 1.

Table 1. The ISO 25000 series family of standards.

Category	Description
2500n	Quality Management Division Standards in this category define all common models, terms and concepts of the 25000 series. They also provide requirements and guidelines for managing requirements, specifications and software product evaluation.
2501n	Quality Model Division Standards in this category define detailed quality models for software, quality in use and data. They also provide some practical instructions for implementing the models.
2502n	Quality Measurement Division Standards in this category include a reference model for measuring software product quality, mathematical definitions of quality measures and practical guidelines for their implementation. This includes internal, external and in-use measures.
2503n	Quality Requirements Division Standards in this category help define quality standards. They correspond to the technical processes of ISO / IEC 15288.
2504n	Quality Evaluation Division Standards in this category include requirements and recommendations for the evaluation of software products by independent evaluators.

With respect to software and data series standards, the most important are depicted in table 2.

Table 2. The ISO 25000 series standards related to software and data quality.

Standard	Description
ISO / IEC 25010: 2011	Evaluation of software and services. Includes a data model
ISO / IEC 25012: 2008	Data evaluation
ISO / IEC 25020: 2007	Instructions for selecting and designing quality measures of 2500x standards
ISO / IEC 25030: 2007	Design of software quality requirements (on the system side). Applies the ISO9126-1 or ISO25010 model
ISO / IEC 25040: 2011	Requirements and recommendations for the pre-developed, COTS or customized software product evaluation process. Not applicable in assessment of requirements

The most important standard is ISO 25010, which is actually an improvement of ISO9126 [11,25]. It has a similar hierarchical structure and the majority of its features and sub-features are the same. It is comprised of two models (Internals and External Quality and Quality in Use) in contrast to the three models of ISO9126. Other changes include the increased attention to security (addressed as a separate characteristic than a sub-characteristic in ISO9126) and the addition of several new sub-characteristics. ISO 25010 has 8 features versus 6 of ISO9126 and 39 sub-

features (table 3). The new standard still lacks its own defined metrics and is based on the metrics of ISO9126.

Table 3. Structure of the ISO 25010 Internal and External quality model.

	Quality Characteristics	Quality Sub-characteristics
ISO25010 - Internal and External Quality	**Compatibility**	Coexistence Interoperability
	Portability	Installability Replaceability Adaptability
	Maintainability	Modularity Reusability Analysability Modifiability Testability
	Performance Efficiency	Time behaviour Resource Utilisation Capacity
	Functional Suitability	Functional Completeness Functional Correctness Functional Appropriateness
	Reliability	Maturity Availability Recoverability Fault Tolerance
	Usability	Learnability Appropriateness Operability User error protection User interface aesthetics Accessibility
	Security	Confidentiality Integrity Non-repudiation Accountability Authenticity

As a formal taxonomy of software quality dimensions, ISO25010 can be used to identify software and system requirements, validate the comprehensiveness of requirements definition, and identify software design and testing objectives.

5. THE METRICS ECOSYSTEM

5.1 Metrics categorization

Metrics reflect the quality of both the services and data of a software system. They can be measured at different stages of the software lifecycle and from different aspects. Assigning values to metrics and interpreting the results is a critical point for moving the software to the next stage of the lifecycle or for getting feedback that will guide corrections, upgrades or maintenance.
Metrics can be categorized according to the type of object they are evaluating or the type of feature they are referring to. They are distinguished in process metrics, which refer to the development process, resource metrics that refer to the available resources for object development and product metrics that refer to the system's characteristics. Basic categorization of metrics is performed according to the type of features they measure:

- internal metrics are used to measure attributes for which there is a tangible understanding of their physical significance and the ability to measure them. The use of internal metrics is mainly addressed to system engineers contributing to an in-house assessment (internal evaluation).

- external metrics are used to measure attributes that are evaluated when using the software in real-world conditions and (usually) when it is complete. External metrics refer to system features that are not easily measurable and are distinguished by a high level of abstraction. Metrics are based on the definition of quality that emphasizes on end-user satisfaction and directly measure the desired external characteristics. They are grouped by qualitative sub-feature. Their basic feature is that they require user participation and engineering engagement. They capture the external quality of the system in conjunction with the in-house knowledge of the features that the system provides. The importance of external metrics lays in the ability to attribute the external quality of the system in terms of the functions and services it provides to the end user under actual conditions of use.

Measurement of external metrics for end-user quality features can be based on three categories of methods:

- Analytical methods. Analytical methods are performed in the laboratory and end users are not involved. Special assessors evaluate the attributes of the system by checking whether it complies with compatibility rules and standards.

- Experimental Methods. Experimental methods are performed in the laboratory with the participation of end users while specialists monitor them and measure their reactions.

- Inquiry methods. They are performed in real conditions and require active user involvement. Indicative examples are the user interview method, where the evaluator records the user's views and completes questionnaires where users are asked to freely express their views on the qualitative characteristics of the evaluated system or product.

5.2 Functional Suitability metrics

Functional Suitability refers to '*the degree to which a product or system provides functions that meet stated and implied needs when used under specified conditions*' [11]. It is comprised of three sub-characteristics:

- Functional completeness: whether the set of functions (under design or already developed) cover the specified tasks and user objectives (recognized during the analysis or evaluation phase).

- Functional correctness: whether the system provides correct and precise results.

- Functional appropriateness: whether system functions facilitate the accomplishment of specified tasks.

Table 4. A representative set of Functional Suitability metrics

Functional Suitability			
Metric Name	Goal	Application Method	Measurement type
FReq	To measure the satisfaction of functional requirements	Assessment of the functional requirements defined during the design of the system	Likert Scale (ordinal)
CountF	To measure the number of different functions provided by the system	Count	Number (absolute)
CountNav	To measure the number of different navigation functions provided by the system	Count	Number (absolute)
CountFeed	To measure the number of different feedback functions provided by the system	Count	Number (absolute)
CountSearch	To measure the number of different search functions provided by the system	Count	Number (absolute)
RatAd	To measure the available/designed functions vs. the functions that were described in the requirements phase	Ratio #available-designed functions / #required functions	Number (absolute)
PrecSearch	To measure the accuracy of search engine results	Assessment of the accuracy of the search engine	Likert Scale (ordinal)
LiveCom	To measure the number of different live communications functions	Count	Number (absolute)
SocialCom	To measure the number of different social media functions	Count	Number (absolute)
CustTrack	To measure the number of different customer tracking services	Count	Number (absolute)
CustServ	To measure the number of different customer services	Count	Number (absolute)
UpSell	To measure the number of different Upselling services	Count	Number (absolute)
CrossSell	To measure the number of different Cross-selling services	Count	Number (absolute)
AfterSell	To measure the number of different aftersales functions	Count	Number (absolute)
InfoServ	To measure the number of different information services	Count	Number (absolute)
PayMeth	To measure the number of different payment methods	Count	Number (absolute)
Mark	To measure the number of different marketing channels	Count	Number (absolute)
Cback	To measure the number of back-end Information systems linked to the system	Count	Number (absolute)

This characteristic is directly linked to the definition of functional specifications during the analysis and design phase of a software lifecycle. Functions are the core of the system, their number, usability and availability are amongst the parameters that contribute not only to the quality of the e-commerce system but to the added value produced by the software and is capitalised by the vendor. It is therefore a key quality characteristic that usually consumes a large percentage of assets during system design. A representative set of e-commerce related metrics related to Functionality Suitability are proposed, as depicted in the table 4. Metrics calculating ratios (i.e. intended designed functions to actually developed or actually used) are possible.

5.3 Usability metrics

Usability refers to the degree to which '*a product or system can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use*' [11]. A large corpus of relevant literature exists for assessing the usability of web based systems and for tailoring e-services to user needs [8,26]. It is generally acknowledged that for user-centred systems, such as e-commerce systems, to be more usable is to be more competitive in an ever-increasing international business environment.

Table 5. A representative set of Usability metrics

Usability			
Metric Name	Goal	Application Method	Measurement type
Cpurch	To count the number of steps for completing a purchase	Count	Number (absolute)
Csign	To count the number of steps for completing a sign in	Count	Number (absolute)
Clocate	To count the number of number of alternative methods to locate a product	Count	Number (absolute)
Cfilt	To count the number filters available for categorising products	Count	Number (absolute)
Qimag	Quality of images of product	Assessment	Likert Scale (ordinal)
Cimag	To count the number of images available per product	Count	
Qchar	Quality of description of product characteristics	Assessment	Likert Scale (ordinal)
Avfont	Average font size used per page	Average number	Number (absolute)
Avcolr	Average number of colours used per page	Average number	Number (absolute)
R3d	Ratio: number 3D models/ number of products available	Ratio	Number (absolute)
Cinfo	To count the number of different ways available for identifying the important information per product page	Count	Number (absolute)

Usability is comprised of six sub-characteristics:

- Appropriateness recognizability: whether users recognize and to what extend, that the system is appropriate for their needs.
- Learnability: whether the software can be used learned to be used effectively by the users.
- Operability: whether the software is easily operable and controlled.
- User error protection: whether the software protects users against making errors.
- User interface aesthetics: whether the user interface is pleasing and satisfying for the users.
- Accessibility: whether the software is accessible by users with a wide range of characteristics and capabilities.

A representative set of metrics related to Usability are proposed, as depicted in table 5.

5.4 Mapping of metrics to sub-characteristics and facets

Facets define the different aspects of the user interaction with system components. Their nature is more process –depended than technical (i.e. user interacts with the user interface). The customer decision process has been extensively researched in the past and the classic five-stage customer decision process (need recognition, information search, evaluation of alternatives, selection and post-purchase) has been proposed and partially adopted for on-line systems as well [20]. Each decision is a transition in the decision process, and gives valuable insights on how customers think and act when making a purchase. On-line behavior tends to be more complex since many parameters are different or new: vast amount of choices, intangibility of products (e.g. purchasing of services), interaction with other user (e.g. through product reviews), lower costs, adaptation and fast searching, just to mention a few. E-commerce software design should take into account the customer's line of thinking and target to increase the quality if those characteristics that specifically contribute to each purchase stage. These characteristics are not purely technical but of a rather business-technical nature. In this context, six main facets are proposed:

- The Presentation facet: it includes the features the system uses to present the product to the user (presenting the product through image, video, 3D rendering and audio, adaptation).

- The Navigation facet: it includes the mechanisms that support user navigation in services and system data (e.g. mechanisms for searching, filtering information, and navigating through the system).

- The Purchasing Process facet: it includes the mechanisms that serve the user and system interaction when purchasing the product (e.g. electronic basket, favourites, payment and shipping options).

- The Social facet: it includes mechanisms that permit users to interact with each other either directly or indirectly (e.g. reviews sections, use of external social tools).

- The Support facet: it includes pre-sales and after-sales mechanisms that support buyers (e.g. cross-selling and upselling mechanisms, newsletters, on-line chat with support staff).

- The Interaction facet: it includes mechanism that enable the software to interact with back –end systems (e.g. ERP, CRM, Inventory information System) or third party services that are used by the e-commerce software (e.g. search engines, social tools). This facet is not directly visible by the customer but it usually supports important process (e.g. updated on-line inventory of products).

A targeted quality design of e-commerce software can be practically supported by mapping metrics to quality characteristics and facets. By acquiring and acceptable range of values for the metrics through an assessment of existing successful e-commerce software (through benchmarking) or in-house previous efforts (in the case of redesigning the software or upgrading it), a metrics knowledge base is constructed. This knowledge base depicts what is considered qualitative; and in (some) contrast to classic methods, the practicality is increased since measures are available. For example, if the metric Cpurch (of the Usability facet, table 5) is measured to be an average of 4 (meaning most successful vendors complete the purchasing process in an average of 4 steps), then a very specific design goal can be set. Agile software design methods can also benefit from this approach. Targeted design can also be supported by the mapping of metrics to

facets because although the latter are not technical depended they usually correspond to discrete sub-systems or units of the software. Thus metrics can be used for fine-tuning specific software components. Furthermore, the design team can focus on the quality of specific characteristics (e.g. usability) if there is such a need originating from specifications (e.g. targeted user population has specific needs), from lack of available resources (to invest to all quality characteristics) or a need for increasing the quality of specific components (due to user demands or to battle competition). Table 6 presents the mapping of metrics to facets for the quality sub-characteristic of Functional completeness (Functional Stability characteristic of ISO25010). The metric – facet relation is a many-to-many relation: one metric may be relevant to more than one facet. In this case, the metric may take a single value (in case of general-purpose metrics) or a facet-specific value (i.e. different values depending on the facet). The use of metrics depends on the precision targeted by the design team and the available knowledge base data.

Table 6. Mapping of metrics to facets and sub-characteristics of Functional Completeness

Facet	Quality sub-characteristic	Metric used
Presentation	Functional completeness	FReq, CountF, RatAd, InfoServ
Navigation	Functional completeness	FReq, CountF, CountNav, CountFeed, CountSearch, RatAd
	Functional correctness	PrecSearch
Purchasing	Functional completeness	FReq, CountF, RatAd, PayMeth
Social	Functional completeness	FReq, CountF, RatAd, LiveCom, SocialCom
Interaction	Functional completeness	FReq, CountF, RatAd, Cback
Support	Functional completeness	FReq, CountF, CountFeed, RatAd, LiveCom, CustTrack, CustServ, UpSell, CrossSell, AfterSell, Mark

In a similar fashion, Usability metrics (table 5) can be mapped to facets and sub-characteristics.

6. CONCLUSIONS

The large number of standards is an issue for the research community and software developers. The ISO organization has set the goal of reducing available standards, a task that is quite difficult since new technologies require the extension of existing standards. The speed at which old standards are withdrawn is not enough to reduce the confusion that exists. In addition to ISO, organizations or communities such as the W3C issue their own technology-oriented guidelines. Although there is no direct conflict between the two organizations, the standards and guidelines they offer are not compatible in terms of philosophy (technology independent of technology-dependent). Major companies such as IBM have established their own software development standards. Other powerful vendors such as Microsoft and Adobe are introducing new technologies that are genuinely popular becoming standards or setting new standards. Although there is no compatibility issue for users, manufacturers do not have a common definition point for designing e-commerce systems based on quality.

Quality assurance and user satisfaction are some of the main goals in the process of developing business-to-business e-commerce. Quality measurement can be perfrmed by assigning a number or symbol to an entity or a component of the evaluated system and is achieved by using the appropriate metrics. The use of metrics addresses the basic problem of determining measurable quantities in software as the concepts of qualitative features and sub-features are distinguished by flexibility and broad interpretation. The main purpose of using metrics is to provide measurable values for the system's characteristics, which make up the quality of the system. Particularly in client-to-business e-commerce systems, the definition of metrics contributes to the detailed and quantitative mapping of the external quality of the system to design and development goals.

Metrics therefore provide, to a certain extent, objectivity, but there are few frameworks to guide the evaluator to which they should use, for what quality purpose. The literature reports a plethora of external metrics for general use in Web applications. In order to answer the above question, a framework for the use of e-commerce-specific metrics was designed. The framework uses function categorization according to the interaction facets identified in the model that uses metrics to evaluate the behavioural characteristics of metric and quality features of ISO 25010. It is a multi-dimensional model. The views match the metrics in the system functions (answers which parts of the system are evaluated). By matching the metrics to qualitative characteristics, the question regarding the qualitative purpose of the evaluation is answered. The combination of many dimensions in one framework provides a functional quality assessment tool based solely on metrics. The framework is complete but not entirely self-contained in the sense that it could be used complementarily to other design and evaluation techniques.

REFERENCES

[1] Chen, L. and Holsapple, C.W. (2013) “E-business adoption research: state of the art”. Journal of Electronic Commerce Research, Vol. 14, No. 3.

[2] Lee I., and Lee, I. (2013). Trends in E-Business, E-Services, and E-Commerce: Impact of Technology on Goods, Services, and Busines’s Transactions (1st ed.). IGI Global, Hershey, PA, USA.

[3] Zhu, Y. & Yan, Z. (2016). “A Survey on Trust Evaluation in E-commerce”. In Proceedings of the 9th EAI International Conference on Mobile Multimedia Communications (MobiMedia '16). ICST, Brussels, Belgium, Belgium, pp. 130-139.

[4] Turban, E., King, D., Lee, J.K, Liang, T.P. and Turban, D.B (2015). Electronic Commerce A Managerial and Social Networks Perspective. Springer.

[5] Heidrich J. et al. (2014). "Model-based quality management of software development projects". Software Project Management in a Changing World, Springer, pp. 125-156.

[6] Goldfarb, A., Greenstein, S.M. and Tucker, C.E. (2015). Economic Analysis of the Digital Economy. University of Chicago Press

[7] Puspaningrum, A., Siti R. & Akbar J. (2017). “Functional Suitability Measurement using Goal-Oriented Approach based on ISO/IEC 25010 for Academics Information System”. Journal of Information Systems Engineering and Business Intelligence. Vol.3. No. 68.

[8] Stefani A., Xenos M. (2009). “Meta-metric Evaluation of E-Commerce-related Metrics”. Electr. Notes Theor. Comput. Sci., Vol. 233, pp. 59-72.

[9] ISO9000. International Organisation for Standardisation. http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=42180.

[10] ISO 8402. Quality management and quality assurance. ISO Organisation. https://www.iso.org/standard/20115.html

[11] ISO 25010: Software engineering -- Software product Quality Requirements and Evaluation (SQuaRE) -- Measurement reference model and guide. ISO, Switzerland.

[12] Garvin, D.A. (1984). "What does product quality really mean?". Sloane Management Review, pp. 25-43.

[13] Bevan, N. & Azuma, M. (1997). "Quality in Use: Incorporating Human Factors into the Software Engineering Lifecycle". 3rd international Software Engineering Standards Symposium (ISESS '97), 169.

[14] ISO 14598 (2001). Software engineering — Product evaluation — Part 6: Documentation of evaluation modules. https://www.iso.org/obp/ui/#iso:std:iso-iec:14598:-6:ed-1:v1:en

[15] Fenton, N., Pfleeger S.L. (1997). Software Metrics A Rigorous & Practical Approach. Thomson Computer Press.

[16] Bevan, N. (2014). "How you could benefit from using ISO standards". In CHI '14 Extended Abstracts on Human Factors in Computing Systems (CHI EA '14). ACM, New York, NY, USA, pp. 1037-1038.

[17] Glass, R.L. (2007). "A research project with important practitioner-oriented findings". Com. of the ACM, Vol. 50, No. 11, pp. 15-16.

[18] Al-Kilidar, H., Cox, K., Kitchenham, B. (2005). "The use and usefulness of the ISO/IEC 9126 quality standard". 2005 International Symposium on Empirical Software Engineering.

[19] Nakai, H., Tsuda, N., Honda, K., Washizaki, H., Fukazawa, Y. (2016). "A SQuaRE-based software quality evaluation framework and its case study". Region 10 Conference (TENCON) 2016 IEEE, pp. 3704-3707, ISSN 2159-3450.

[20] Mohanty, R.P., Seth, D., Mukadam , S. (2010). "Quality Dimensions of E-Commerce and their Implications". Total Quality Management & Business Excellence, Vol. 18, No. 3, pp. 219-247.

[21] Carrillo, A.B., Mateo, P. R. and Monje, M. R. (2012). "Metrics to evaluate functional quality: A systematic review" .7th Iberian Conference on Information Systems and Technologies (CISTI 2012), Madrid, 2012, pp. 1-6.

[22] Rodriguez, M, Oviedo, J.R., Piattini, M. (2016). "Evaluation of software product functional suitability: A case study". Software Quality Professional, Vol 18, No 3, pp. 18-29.

[23] Forsgren, N. and Kersten, M. (2018) " DevOps metrics:. Commun. ACM, Vol. 61, No 4, pp.44-48. DOI: https://doi.org/10.1145/3159169

[24] Gordieiev O., Kharchenko V., Fominykh N., Sklyar V. (2014) "Evolution of Software Quality Models in Context of the Standard ISO 25010". In: Zamojski W., Mazurkiewicz J., Sugier J., Walkowiak T., Kacprzyk J. (eds), Proceedings of the Ninth International Conference on Dependability and Complex Systems DepCoS-RELCOMEX. June 30 – July 4, 2014, Brunów, Poland. Advances in Intelligent Systems and Computing, Vol 286. Springer.

[25] Abran, A. (2010). Software Metrics and Software Metrology. Wiley-IEEE Computer Society Pr.

[26] Lalmas, M. and Hong, L. (2018) "Tutorial on Metrics of User Engagement: Applications to News, Search and E-Commerce". In Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining (WSDM '18). ACM, New York, NY, USA, 781-782. DOI: https://doi.org/10.1145/3159652.3162010