QUIM Map: A Repository for Usability/Quality in Use Measurement

Harkirat Kaur Padda

A Thesis

in

The Department

of

Computer Science

Presented in Partial Fulfillment of the Requirements

for the Degree of Master of Computer Science at

Concordia University

Montreal, Quebec, Canada

March 2003

0-612-77913-0

ABSTRACT

QUIM Map: A Repository for Usability/Quality in Use Measurement

Harkirat Kaur Padda

QUIM - Quality in Use Integrated Measurement - is a framework being developed by the human-centered software engineering group and it aims to bridge the gaps between HCI practices and software engineering quality models for measuring quality in use or the user perspective of software quality. QUIM is based on the decomposition of factors into measurable criteria and metrics. Empirical rules for understanding and interpreting metrics are also included in QUIM. In this thesis, we identified and built a large, consistent repository of factors, criteria and metrics by surveying existing usability and software engineering quality models. QUIM Map includes 10 factors namely - Efficiency, Effectiveness, Satisfaction, Productivity, Learnability, Safety, Trustfulness, Accessibility, Universality and Usefulness that quantify the quality in use of software products. These factors are mapped to a total of 27 criteria that are further measured through more than 125 metrics for assessing quantitatively as well as qualitatively the quality in use.

To My Mother and Mother in-law

"Mawan Thandian Chhawan"

Acknowledgments

Hereby, I would like to thank all persons who helped and encouraged me in writing the thesis.

I am especially indebted to my supervisor Dr. Ahmed Seffah, who supported and encouraged my research from the beginning and offered me valuable suggestions time to time.

I would also like to thank all my friends who gave valuable comments in improving the thesis. Last, but not least I wish to thank my family.

I am grateful to my husband, Sukhwinder Singh Padda, for his moral support in learning, and writing the thesis. His encouragement always helped me to do something better.

In December 2001 our daughter, Banmeet Kaur Padda was born. That beautiful event made me have a couple of months' break in my masters work, which was fully compensated by the new and so far never experienced feeling of becoming a mother. It was really pleasant to watch 'Bannu' grow, learn to walk and talk, watch her discovering the world around her. I wish to thank her for all unrepeatable experiences she gave to our family.

My parents and my in-laws are the ones who gave me inspiration to do research, and maintained my belief of being capable to do it.

Being a mother, I learned the value of motherhood and experienced the hard times that all mothers may have while taking care of their children. Therefore, I would like to dedicate this thesis to my mother and my mother in-law.

Table of contents

List of Figures

List of Tables

Chapter 1

Introduction

The main objective of this thesis is to develop a large, exhaustive and consistent repository of quality in use factors, criteria and metrics. The sub objectives are to:

- Bridge HCI standards and software engineering models for software quality and usability measurement.
- Define the empirical foundations for understanding and interpreting quality in use measures.

1.1 Motivations

An analysis of existing models and standards shows us that there is an extreme need for a consistent and interpretive repository or knowledge base of Quality in Use measurement; which includes factors, criteria, metrics descriptions and interpretations. Therefore, we decided to build a repository that fulfills these demands. Our repository is useful to any person interested in quality in use. We organized it in such a way that it is easier to measure user's perspective of software quality. The factors considered in this repository are based on an extensive research conducted by several usability professionals worldwide. We considered the most recent and challenging factors such as accessibility and universality of websites. The factors are decomposed into measurable sub factors called criteria. These criteria are then mapped into measurable components called metrics. The metrics measure the quantitative as well as qualitative quality in use aspects of software products.

1.2 Research Methodology used

Our research methodology involved the following activities:

1. The use of QUIM framework developed by (Donyaee, 2001), which provided the theoretical foundations of our research.
2. Identification of quality in use factors via an extreme analysis of the literature investigation to contrast their meanings
3. Decomposition of these factors into measurable criteria.
4. The analysis of these criteria and collection of various quantifiable and qualitative metrics to measure those criteria.
5. Interpretation and analysis of the values of various metrics in order to highlight their contribution to quality in use.

1.3 Structure of the thesis

This thesis consists of 6 chapters.

- **Chapter 1**

Chapter 1 describes the goals of the thesis and research motivation along with a research methodology used by the author.

- **Chapter 2**

Chapter 2 presents some basic definitions and concepts on software quality and quality in use. It also explains the measurement process. Many existing measurement models and standards that are used to quantify software quality are also briefly introduced.

- **Chapter 3**

Chapter 3 deals with the measurement model proposed by HCSE group at Concordia University. It briefly explains the driving forces behind the development of this model, its structure, and the relationships involved among factors, criteria and metrics. This chapter also highlights important attributes of QUIM along with a brief introduction of the QUIM editor.

- **Chapter 4**

Chapter 4 is a narrative description of the criteria we defined. Based on a literature review, various definitions of the criteria are proposed. Some related metrics are also explained for a few criteria in order to make it easier to understand these criteria. So far, our repository is a collection of more than 125 metrics. It is a long list; therefore we decided to keep it in the appendix section of the thesis. Each of the metric described here is presented with a brief definition, a formula (if it exists) and a useful interpretation of its value. Interpretation of these metrics makes them useful and usable.

- **Chapter 5**

Chapter 5 describes the factors, which are quality in use characteristics of any software product. It provides detailed information on 10 quality in use factors, including - Efficiency, Effectiveness, Productivity, Satisfaction, Learnability, Safety, Trustfulness, Accessibility, Universality and Usefulness. A definition of each of these factors is proposed. Each factor criteria mapping is also sketched to show a clear image of the factor decomposition.

- **Chapter 6**

Finally, in the conclusion we summarize the work described in this thesis along with a few basic problems encountered while building the repository. We also point out the possible directions to our research.

Chapter 2

The Concept of Quality in Use and Its Measurement

2.1 Software quality

Quality is an important issue that impacts all aspects of human life. Quality of material products can be defined in a precise and measurable way such as the speed of a car is 130Km/h. However, the quality of software is hard to define, basically because of the difficulties arising in finding adequate quality factors, attributes and proper quality- measuring methods and tools.

(German Industry Standard DIN 55350 Part 11,1995) states that quality comprises all characteristics and significant features of a product or an activity, which relate to the satisfying of given requirements.

(IEEE Std. 729,1983) defines software quality as:

1. The totality of features and characteristics of a software product that bear on its ability to satisfy given needs: for example, conformance to specifications.
2. The degree to which software possesses a desired combination of attributes.
3. The degree to which a customer or user perceives that software meets his/her composite expectations.
4. The composite characteristics of software that determine the degree to which the software in use will meet the expectations of the customer.

(ISO 8402,1986) defines quality as:

'The totality of features and characteristics of a product or service that bear on its ability to satisfy specified or implied needs.'

(ISO 8402,1986) definition provides the product oriented view of quality and it indicates that quality is determined by the presence or absence of the specific attributes. Both IEEE and ISO definitions associate quality with the ability of the product or service to fulfill its function. This is achieved through the features and characteristics of the product.

Ince (1994) describes the modern view of quality as:

A high quality product is one which has associated with it a number of quality factors. These could be described in the requirements specification; they could be cultured, in that they are normally associated with the artifact through familiarity of use and through the shared experience of users; or they could be quality factors which the developer regards as important but are not considered by the customer and hence not included in the requirements specification.

(ISO/IEC 9126,1991) categorizes software quality from a user perspective as functionality, reliability, usability, efficiency, maintainability and portability. Here the objective is to meet user needs. According to Bevan (1995) ISO/IEC 9126 definition depicts that quality is not an absolute property, but depends on the context of use. The context of use includes the users, tasks, hardware, software, and materials, as well as the physical and social environment. (ISO/IEC 9126,1991) also gives user, developer and manager views on software quality (See Table 1).

Table 1: Views on software quality

User	Using the software, its performance, and effects of using the software product. User is not interested in the internal aspects of the software product.	Functionality, reliability, efficiency, usability and portability.
Developer	Intermediate and final product quality	In order to evaluate the intermediate product quality, at each phase of the development cycle the developers have to use different metrics for the same characteristics because the same metrics are not applicable to all phases of the cycle.
Manager	Overall quality and not a specific quality characteristic.	The manager assigns weights to the quality characteristics, reflecting business requirements. The manager may also need to balance the quality improvement with management criteria such as schedule delay or cost overrun because he/she wishes to optimize quality within limited cost, human resources and time frame.

2.2 Usability as an attribute of quality

As a quality attribute, usability is an extremely difficult quality factor to define. The main reason for this is that the attributes, which a product requires for usability, depend on the nature of the user, task and environment. Many usability professionals and standards have defined their own set of usability attributes as shown in Table 2.

Table 2: Perceptions of usability attributes

Efficiency in use	Efficiency	Speed of performance	Efficiency of Use	Throughput	Effectiveness (Speed)
Learnability		Time to Learn	Learnability (Ease of learning)	Learnability (Ease of learning)	Learnability (Time to learn)
Rememberability		Retention Over Time	Memorability		Learnability (Retention)
Reliability in use		Rate of errors by users	Errors/Safety	Throughput	Effectiveness (Errors)
User Satisfaction	Satisfaction (Comfort & Acceptability Of use)	Subjective Satisfaction	Satisfaction	Attitude	Attitude
	Effectiveness				
				Flexibility	Flexibility

2.3 Quality in use

(ISO/IEC 9126,1991) includes quality in use as one of the approach to product quality that measures the degree of excellence and defines it as:

'The extent to which the software meets the needs of the user in the working environment.'

(ISO/ IEC 14598-1, 1998) defines quality in use as:

'The effectiveness, efficiency and satisfaction with which the specified users can achieve specified goals in specified environments.'

A product meets the user requirements if it is effective (accurate and complete), efficient in use of time and resources, and satisfying, without taking into consideration the specific attributes the product possesses.

(ISO 9241-11,1998) highlights that quality in use is not independent of context of use and the amount of quality in use achieved depends on the particular context in which the product is used.

(ISO/IEC 9126-1,1998) says that quality in use is the user's view of quality and is measured by using the software product in a specified context of use. Quality in use was introduced as a high-level quality objective in the revised (ISO/IEC 9126-1,1998) as:

'The capability of the software product to enable specified users to achieve specified goals with effectiveness, productivity, safety and satisfaction in a specified context of use.'

ISO/IEC 9126 defined the following framework to distinguish the three approaches (Figure 1) to software product quality.

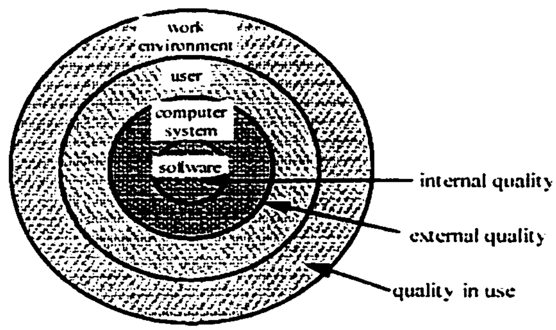


Figure1: Approaches to software product quality

According to Bevan (1999), the internal quality is the reflection of the design philosophy and strategy. It includes internal software properties, which are mainly related to the programming environment, and also internal quality requirements for the quality characteristics defined in ISO/IEC 9126. External quality is estimated or predicted for the end software product at each stage of development, and is based on knowledge of the internal quality.

Bevan (1999) explains that quality in use measures the degree of excellence, and can be used to validate the extent to which the software meets user needs.

The three qualities internal, external and quality in use are related in such a way that appropriate internal attributes of the software are a pre-requisite for achieving the required external behavior, and appropriate external behavior is a pre-requisite for achieving quality in use (Figure 2).

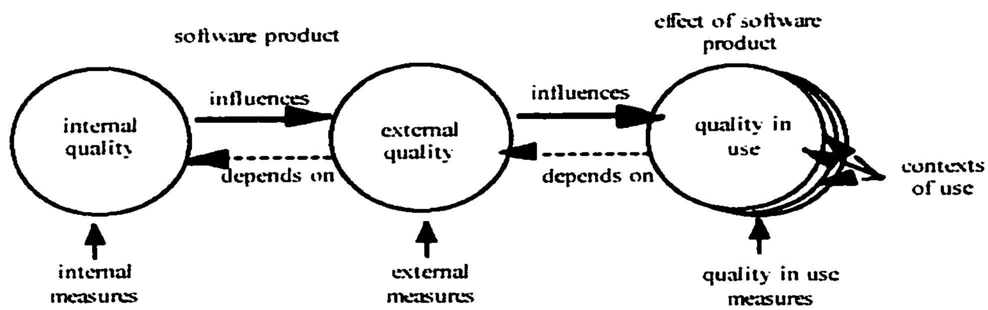


Figure2: Relationships among internal, external and quality in use

2.3.1 Quality in use versus Usability

The difference between usability and the quality in use is a matter of focus. When usability is evaluated, the focus is on improving the software product while the context of use (user, task, equipment and environment) is treated as given. It means that the level of usability achieved will depend on the specific circumstances in which a product is used. On the other hand when quality in use is evaluated, any component of context of use may be subject to modification or improvement.

2.3.2 Significant benefits of quality in use

'A focus on quality in use requires not only easy-to-use interfaces, but also the appropriate functionality and support for real business activities and work flows. Quality in use should be the major design objective for an interactive product….' (Bevan, 1999)

Bevan (1999) has listed the following potential benefits of increased quality in use:

1. Increased efficiency: Increased quality in use implies that users will operate effectively and efficiently on a software product that has improved ergonomic design and functionality suited to the users needs.
2. Productivity Improvement: User productivity in terms of the user performance while operating a software product will increase because the well-laid out user interface of a good designed product allows the user to concentrate on the task rather than the tool.
3. Errors Reduction: A significant decrease in user errors will result with the reduction in interface design faults, not matching to the user's task needs like inconsistencies, ambiguities etc.
4. Training Reduction: A well-designed user interface will decrease the time and effort spent on learning the functionality.
5. Improved acceptance: Users would like to use a software product that will be fulfilling their demands and this will enhance their trustfulness in the software product. This will increase the product usage as well.

2.4 Measurement

The business of pinning numbers on things—which is what we mean by measurement, has become a pandemic activity in modern science and human affairs. The attitude seems to be: if it exists, measure it. Impelled by this spirit, we have taken the measure of many things formerly considered to lie beyond the bounds of quantification.... (Stevens, 1959)

Measurement can be defined as:

- 'The assignment of numerals to objects or events according to a rule.' (Stevens, 1959)
- 'Measurement is the process by which numbers or symbols are assigned to attributes of entities in the real world in such a way as to characterize the attributes by clearly defined rules.' (Fenton, Whitty, 1995)
- 'Quantities are measurements of qualities.' (Kirchner, 1959)

2.4.1 Metrics

The term metrics has been used widely to describe the act of measurement, and to imply the qualitative or quantitative performance indication. Its purpose is to accurately quantify an aspect of an existing or proposed system.

'A mathematical function based on distances, or on quantities treated as analogous to distance for the purpose of analysis.' Oxford English Dictionary [75]

(ISO/IEC 9126, 1991) defines a software quality metrics as:

'A quantitative scale and method which can be used to determine the value a feature takes for a specific software product.'

This is not an appropriate definition because sometimes metrics are based on qualitative judgment or facts (ISO/IEC 9126, 1991) also suggests that if appropriate metrics are unavailable and cannot be developed, verbal descriptions or "rule of thumb" may sometimes be used.

A more precise definition of a metric is given by IEEE standard, which is: 'A software metric is a function whose inputs are software data and whose output is a single numerical value that can be interpreted as the degree to which the software possesses a given attribute that affects its quality. '

2.4.1.1 Classification of metrics

- Objective & Subjective metrics: Objective metrics should always result in identical values for a given metric as measured by two or more observers, whereas for subjective metrics the subjective assessment by observers may give different values for a given metric.
- Primitive & Computed metrics: Primitive or direct metrics are those that can be directly observed. For example: number of errors, task time etc. Computed or indirect metrics are those that cannot be directly observed but are computed from other metrics in some way. For example: human efficiency is given as task time by human effort expended while performing the task. Computed metrics are combinations of other metric values and thus are often more valuable than primitive metrics.

2.4.1.2 Attributes of metrics

Mills (1993) states that a good metric should facilitate the development of models that are capable of predicting process or product parameters and not be limited to describing them. Mills (1993) suggests that metrics should be:

- Simple & precisely defined: It means that a metric should be simple and precisely defined so that it is clear how the metric is evaluated.

- Objective: The metrics should be objective to the greatest possible extent, however some metrics may require subjective evaluation.
- Easily obtainable: It should be relatively easy to capture and/or compute the metric.
- Valid: The metric should measure what it is intended to measure.
- Robust: The metric should be relatively insensitive to insignificant changes in the process or product.

2.4.1.3 Validation of metrics

(IEEE Std 1061,1998) defines validation as the act or process of ensuring that a metric reliably predicts or assesses a quality factor.

(Fenton, Pfleeger, 1997) say that validation of a software metric is the process of ensuring that the metric is a proper numerical characterization of the claimed attribute; this means showing that the representation condition is satisfied.

2.4.1.3.1 Validation conditions

The normative part of standard (ISO/IEC 9126-4,2000) has identified the following conditions for metrics validation:

1. Correlation: It means that the variation in metric value is associated with the variation in quality factor values and that is given by the square of the linear correlation coefficient. Schneidewind (1992) states that this criterion assesses whether there is a sufficient linear association between Factor (F) and Metric (M) to warrant using M as an indirect measure of F whose direct measurement is not possible.

2. Tracking: This condition tests that if a metric is directly related to quality factor then whether the metric (M) value changes in same direction (i.e. either increases or decreases), as there is a change in the value of corresponding factor (F). This criterion helps to measure the change in quality characteristics of the software product or process along time period indirectly through the use of metrics having tracking ability.

2. Consistency: This criterion states that if the quality characteristics values F1, F2... Fn corresponding to products or processes 1, 2... n has the relationship F1 > F2 >... Fn, then the correspond metric values would have the relationship M1 > M2 >, ... Mn. The use of metrics having consistent ability helps in finding the exceptional and error prone components in a software product.

4. Predictability: A metric M used at time t1 must predict the factor (F) value measured at time t2 with the following allowed prediction error.

   Allowed Prediction error = [predicted F (t2)-actual F (t2)] /actual F (t2)

   Metrics with the prediction power helps in assessing the future values of the quality factors.

5. Discriminative: A metric should have the discriminate power to distinguish between high quality software components and low quality software components.

2.4.2 Measure

(ISO 9126-4,2000) defines measure as the number or category assigned to an attribute of an entity by making a measurement.

2.4.2.1 Classification of measures

- Direct & indirect measure: Direct measure is a measure of an attribute that does not depend upon a measure of any other attribute. For example: size of page, total number of menu options in an application or number of colors used. The indirect measure is a measure of an attribute that is derived from measures of one or more other attributes. For example: the task time depending not only on the software product but also on its context of use.
- Objective & subjective measure: Objective and subjective measurements often address fundamentally different needs. It is admirable to strive for measurements that are as objective as possible but subjective measurements are also the good assessment of the software product and help in achieving the measurement goals.
- Internal & external measure: An external measure is an indirect measure of the software product derived from measures of the behavior of the system of which it is a part. For example: the number of faults found during the testing of the software products. External measures can be used to evaluate quality attributes closer to the ultimate design objectives. An internal measure is a measure derived from the product itself, either direct or indirect; it is not derived from measures of the behavior of the system of which it is a part. For example: lines of code are an internal measure of the software product.

2.4.3 Metrics versus Measures

Metrics numerically characterize simple attributes like number of errors found, task time etc. On the other hand, measures are the functions of metrics that can be used to assess or predict more complex attributes of the software product like quality.

2.4.4 Driving forces behind measurement and metrics

There are four reasons for measuring software processes, products, and resources:

- Characterization: To gain understanding of processes, products, resources, and environments, and to establish baselines for comparisons with future assessments the characterization should be done.
- Evaluation: Evaluation helps in assessing the achievement of quality objectives and the impacts of technology and process improvements on products and processes. Evaluations are done to determine the status with respect to plans. Measures are the detectors that detect when the projects and processes are drifting off track, so that they could be brought back under control.
- Prediction: Prediction is formally the future assessment. It is done so that the future plans can be prepared. Prediction helps in analyzing the risks and it estimates the design/cost tradeoffs. Measuring for prediction involves gaining understandings of relationships among processes and products and building models of these relationships, so that the values that are observed for some attributes can be used to predict others. It

helps in establishing achievable goals for cost, schedule, and quality—so that appropriate resources can be applied. Predictive measures are also the basis for extrapolating trends, so estimates for cost, time, and quality can be updated based on current evidence.

- Improvement: Measurement helps in improving the software product quality and process as it gathers the quantitative information about the root causes or inefficiencies in the existing product or process. Measures also help us plan and track improvement efforts. Measures of current performance give us baselines to compare against, so that judgment can be made that whether or not the improvement actions are working as intended and what the side effects may be.

2.4.5 Measurement process

In 1988, Kriz [84] suggested a measurement process, which depicts that direct measurement of the real world entities is not possible because of the obstructive forces to intelligence that hinders the production as well as the interpretation of the empirical results. This is because humans are not capable enough to make a precise and objective judgment about the real world entities and so Kriz called it as "Intelligence Barrier" as depicted in Figure 3.

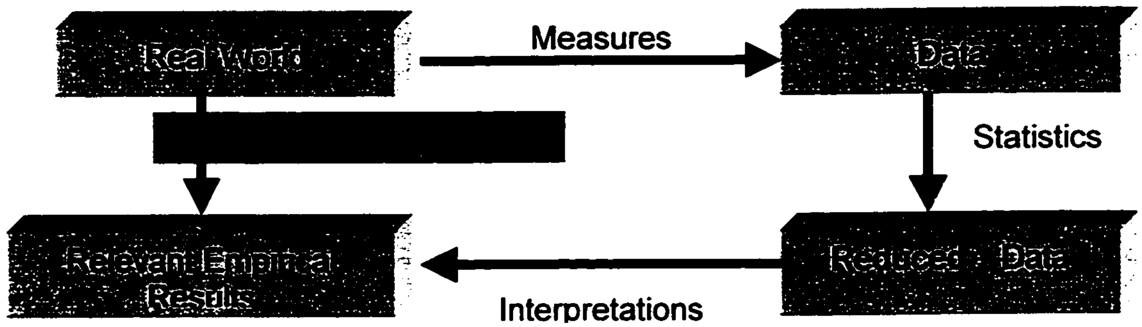


Figure 3: Measurement process by Kriz

The measurement process then begins with using the mathematical and statistical models of measurement that firstly measure the data and then reduce it using the statistical procedures like means, variances etc. The reduced data is finally interpreted to produce the relevant empirical results. The interpretation is the necessary part of any measurement process because otherwise the metrics or the measures are useless.

2.4.6 Measurement scales

Scales provide values and units for describing attributes. For example, size of page may be 40KB; number of color used is 4. Each of these observations has been quantified (or labeled) with a value from a (presumably) well-defined scale.

(ISO 9126-4,2000) defined the five types of scales as follows:

- Nominal: A nominal scale provides a name or label as the value for an attribute. The order of values on the scale has no significance. For example, the color of a non-visited hyperlinked text = blue, Nominal measures are often used to classify entities so that they can be sorted prior to counting the number of occurrences or aggregating measured values.

- Ordinal: An ordinal scale permits measured results to be placed in ascending (or descending) order. For example, ordering software failure by severity levels (negligible, marginal, critical, catastrophic).
- Interval: An interval scale adds the concept of distance. Interval scales permit us to add and subtract values. This includes artificial rating scales. For example, rating scales of sensitive questionnaire for asking about usability.
- Ratio: A ratio scale adds an origin (a meaningful, non arbitrary zero value). With a true origin all the mathematical operations that are customarily used for real numbers become meaningful. Examples more directly related to software include time between software failures.
- Absolute: Absolute scales are special cases of ratio scales in which the only admissible multiplier is 1. For example, counting the number of occurrences in each of several nominal classes (e.g., the number of links per page, number of languages supported by the site etc.) is an instance of the use of an absolute (counting) scale for the attribute "number of occurrences found."

2.4.7 Data

Choosing valid measures or metrics for a measurement process is not sufficient to guarantee that they will portray the real world attributes; it is the data that will decide about their mapping to the real world attributes.

(Fenton, Pfleeger, 1997) listed the following desirable properties of data:

1. Correctness: The data should be collected according to the exact rules of definition of the metric. For example: a measure of time spent on a particular task is correct if time is measured from the beginning of that specified activity and ends at the completion of that activity.
2. Accuracy: It is the difference between the theoretical and actual value.
3. Precision: It is the number of decimal places needed to represent the data.
4. Replication: For case studies, experiments or surveys etc., the measurement process is normally repeated under various context of use so that the measurement results could be compared to establish the baseline measures and the objectives of organization.
5. Consistency: It means the evaluation results should be same for the same context of use. It implies that for different context of use, to have comparable results of the measures, the data should be captured in the same way.

Figure 4 shows the data collection in the software measurement process.

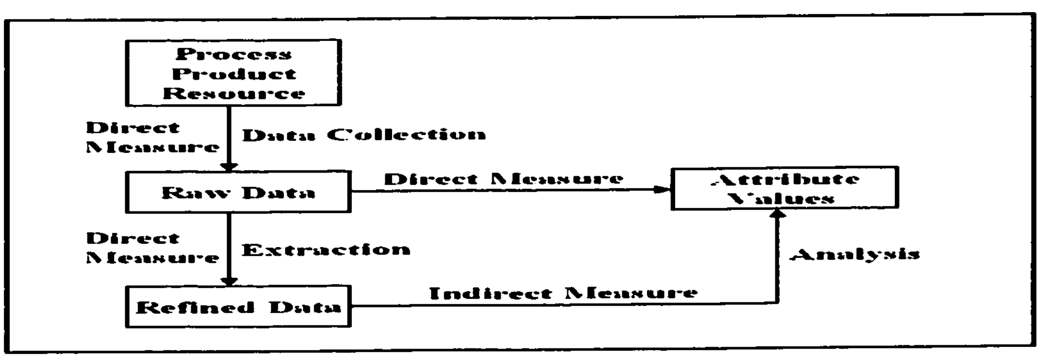


Figure 4: Data collection in software measurement process

This figure depicts two types of data, which are:

- Countable or raw data, which are the simple countable entities that result from the initial measurement of process, product or resources.
- Calculable or refined data are the results of calculations that are extracted from raw data.

2.4.7.1 Data collection

Data collection involves human observation and reporting. Data can be collected either manually or automatically using tools. Although manual recording is subject to error, omission and delay, yet sometimes it is the only way to collect data where automatic data collection facilities are not available. Automatic data collection is desirable, and sometimes it is essential for safety critical systems where the accuracy and precision of the data is most required. Quality in Use data can be collected from a variety of sources, for example: user observations, usability professionals' assessments, questionnaires, user documentation, system, task analysis and others software requirements techniques, user interface prototypes such as storyboard, paper, video and software prototypes etc.

2.5 Quality model

(ISO 9126,1991) introduces the concept of a quality model as:

'A description in terms of a structured set of quality characteristics and sub-characteristics of those attributes which contribute to a software product being considered to be a good one of its kind.'

'The set of characteristics and the relationships between them that provide the basis for specifying requirements and evaluating quality.' (ISO/IEC 14598-1,1999)

2.5.1 Quality in use model characteristics

A quality in use model should have these properties:

- Efficiency: An efficient quality in use model is the one, which considers all the factors that are required for a product to meet desired usability goals in a specified context of use.
- Explicit relationships: It should explicitly define the relationships that exist between the factors, sub-factors and the applicable metrics.
- Decomposability: A good quality in use model should decompose the characteristics of a product as defined by the stakeholder into measurable attributes.
- Flexibility: It should be used at various phases of the software development lifecycle.
- Usability: The model should be understandable by both the novice and expert developers involved in the software development lifecycle.
- Automated Support: A tool should help in usability requirements collection as well as the measurement of usability.

2.5.2 Software quality decomposition

Decomposition involves identifying and defining a set of software factors, also called attributes or characteristics, and decomposing them into sub-factors or criteria. At the lowest level, they can be mapped to corresponding metrics and assessment techniques, as shown in Figure 5.

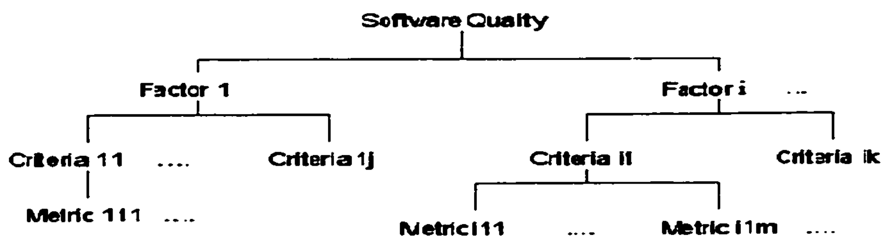


Figure 5: Software quality decomposition

2.5.3 Quality models and standards

There are many models and standards developed for measuring the software product quality. For example:

McCall's model

McCall et al. (1977) proposed one of the earliest quality models.

This model is also called GE (General Electric) model or FCM (Factor, Criteria and Metric) model. This model describes quality as being made up of a hierarchical relationship between the quality factors, quality criteria, and quality metrics. The term "quality factor" is a key characteristic of the software product. "Quality criterion" is an attribute of the quality factor that defines the product. "Quality metric" denotes a measure that can be used to quantify the criterion. McCall described a systematic approach to quantify quality as:

1. Determine all of the factors that would have an effect on the software quality.
2. Identify the criteria for judging each factor.

3. Define metrics for each of the criteria and establish a normalization function that defines the relationship between the metrics of all the criteria pertaining to each factor.
4. Evaluate the metrics.
5. Correlate the metrics to a set of guidelines that every software development team could follow.
6. Develop recommendations for the collection of metrics.

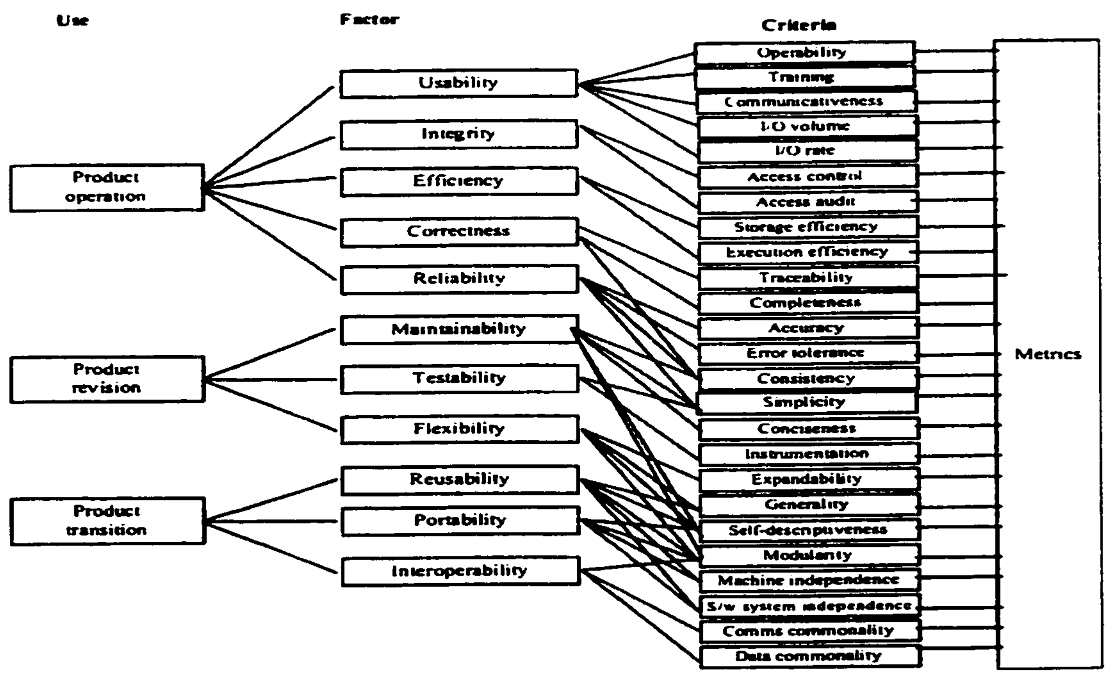


Figure 6: McCall' software engineering quality model

As depicted in Figure 6, McCall has identified 11 quality factors, 25 criteria and 41 metrics to measure these criteria. These metrics involved questions dealing with the degree of compliance to the criteria and had either a "yes" or a "no" for an answer. That is why the metrics results are highly subjective and it is generally difficult to interpret them. It is based on three uses of a software product, which are:

- Product revision i.e. how easily can the product be changed?
- Product operation i.e. is the product usable in its present state?
- Product transition i.e. will it be possible to transfer the product into another hardware / software environment?

Boehm model

This model was developed in 1978 by a team of researchers, lead by Barry W. Boehm, working at a software house of TRW Systems and Energy Inc. Like the "McCall model", this model also focuses on the final product. Both McCall and Boehm models assume that the quality attributes are on a too high level to be meaningful or to be measurable therefore further decomposition is needed. These lower- levels quality characteristics are called quality criteria. In a third level of decomposition the quality criteria are associated with a set of directly measurable attributes called quality metrics. Boehm's model of software quality is depicted in Figure 7.

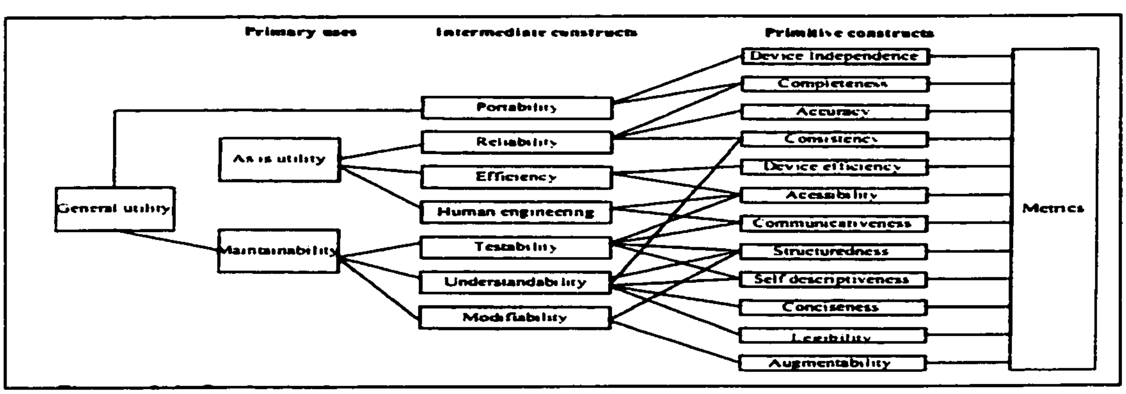


Figure 7: Boehm's model

It incorporates 19 quality factors encompassing product utility, product maintainability, and product portability. The criteria in McCall and Boehm models are not independent; they interact with each other and often cause conflict.

GQM model

Proposed by Victor Basili, the main idea behind the GQM is that measurement should be goal oriented and based on context characterization. GQM defines measurement at three levels (Figure 8):

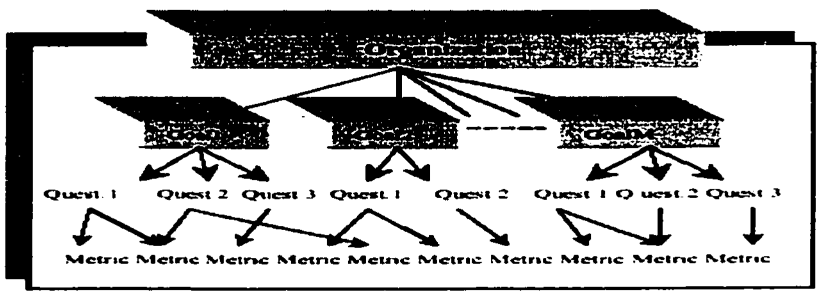


Figure 8: GQM framework for an organization

1. Conceptual level (Goals): Goals are defined for an object based on specific needs, from various points of view and relative to a particular environment.
2. Operational level (Questions): A set of questions is defined for the model of the object that characterizes and assesses a specific goal.
3. Quantitative level (Metrics): A set of metrics based on the model of the object under study is defined for each question in order to answer it in a measurable manner.

HOQ model

HOQ or House Of Quality Model is a component of a large quality framework called Quality Function Deployment (QFD). The QFD model was proposed by Yoji Akao and Shigeru Mizuno in 1970. The purpose of QFD or HOQ is to deploy the voice of user throughout the product development lifecycle. The idea is to translate the customer requirements into the appropriate technical requirements for each stage of software product development. Figure 9 depicts the basic HOQ model.

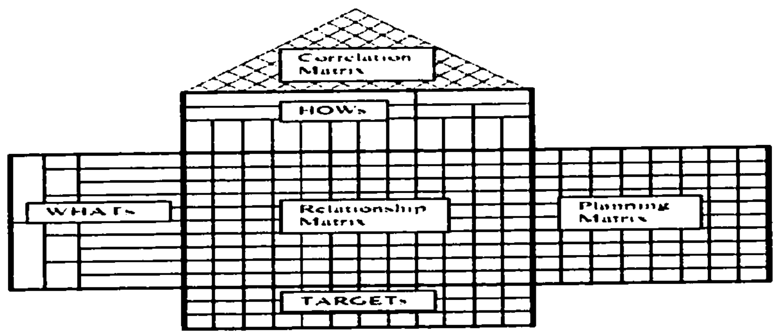


Figure 9: Basic house of quality model

In this figure, WHATs determine the needs or requirements of the users that are collected through several communication techniques like interviewing, feedback etc. HOWs is the determination of the software characteristics that will contribute to these user requirements. Relationship Matrix describes the corresponding relationships between these WHATs and HOWs. This is normally determined through a rating system like numbers or symbols. Symbols may express relationships such as weak, moderate, strong, or no influence etc. The TARGETs section is used to determine the criticality of the relationships. The peak of the HOQ is the Correlation Matrix, which shows the correlation between each of the performance parameters or HOWs of the house. It can be shown with symbols for positive or negative relationships. The Planning Matrix can also be added to the HOQ to determine the future needs or the goals of the product.

This HOQ can be further analyzed and broken down into other HOQ models based on the analyst judgment, to finally reach a measurement model that determine the target values.

ISO 9126 standard

For many years, there was a desperate need for a unique, unambiguous and usable software quality model. In 1991, an international standard was proposed for software quality measurement i.e. ISO 9126: "Software Product Evaluation: Quality Characteristics and Guidelines for their Use". This standard incorporates six quality characteristics: five of them (reliability, usability, efficiency, maintainability and portability) are similar to as in McCall's model and sixth i.e. functionality (Are the required functions available in the software product?) is a new one. These quality characteristics can be further refined into sub characteristics that can have measurable attributes.

Revision of the model in 2000, introduced the concept of quality in use as the seventh software quality characteristic (Figure 10). Quality in use is the combined effect of the six software product quality characteristics and is determined in terms of effectiveness, productivity, satisfaction and safety.

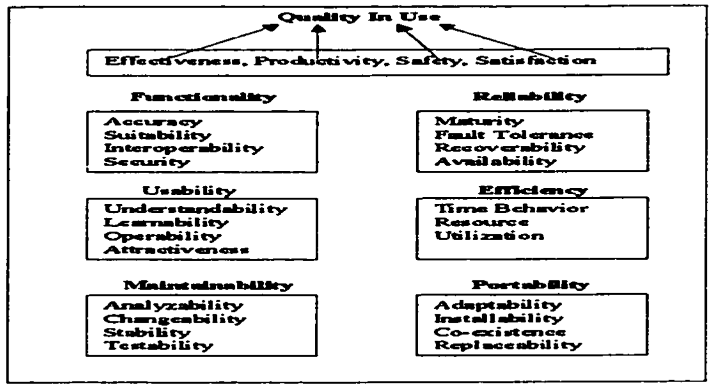


Figure 10: ISO 9126 –2000

IEEE 1061 standard

In 1992, Norman F. Schneidewind proposed the first IEEE standard (Software Quality Metrics Methodology or IEEE 1061 standard) that deals with the quality metrics. According to Schneidewind (1992) IEEE 1061 standard is a methodology for establishing quality requirements and identifying, implementing, analyzing and validating the process and product software quality metrics. He also emphasized that IEEE 1061 does not mandate specific metrics for use. The model proposes a flexible hierarchy shown in Figure 11, where additions, deletions, modifications of quality factors, quality sub factors and metrics are permitted.

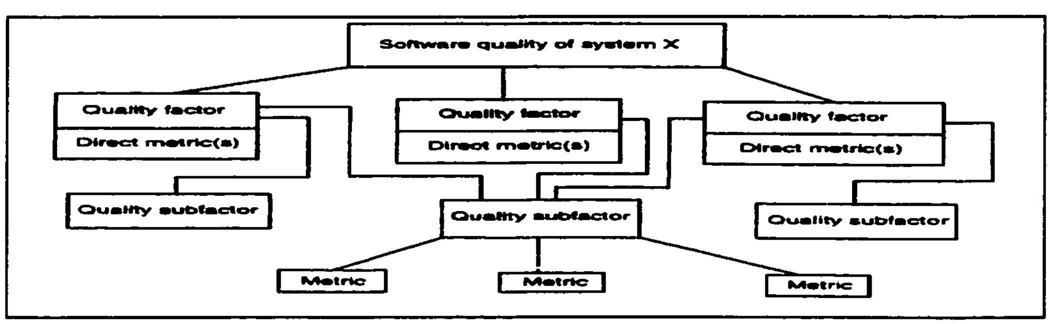


Figure 11: Framework for IEEE 1061

In this figure, at the highest level, software quality requirements are established and the quality factors that represent these software quality views are identified. The direct metric(s) that serve as a quantitative gauge of quality factors are assigned. The values of these direct metric(s) is set to define the acceptable limits of quality. In the middle level, the quality factors are decomposed into measurable sub factors that are further broken down into metrics at the lowest level.

Beside McCall, Boehm, GQM etc., there are a variety of models like AIDE [82], GLEAN [48], SANe [51], DRUM [59] etc. available for measurement of various aspects of the software products.

Chapter 3

QUIM - Quality in Use Integrated Measurement

3.1 Introduction

This chapter is based on the work of (Donyaee, 2001). The purpose to include his work in this thesis is mainly to introduce the concept of QUIM. The contents in this chapter are updated and extended.

(Donyaee, 2001) has already listed the driving forces behind QUIM; he said that QUIM is developed basically for the following reasons:

- The existing software quality models are not addressing the important issue of quality in use of the software products.
- To give precise measurement of quality in use objectively rather than subjectively,
- To reduce the time and expenses of the software product while testing.
- Try to eliminate/ reduce the problems suffered by the software product regarding the user interface.
- To introduce the quality in use practices into the first stages of software development process.
- To facilitate the adaptation of quality in use easy for all developers with different levels of expertise.
- To make the acquiring of quality in use more consistent between developers.
- To develop a dynamic quality model.

- To help the developers in creating more usable software products.
- To create a repository of validated usability metrics.
- To facilitate the integration of usability into the software engineering models.
- To develop a common stage for Human Computer Interaction experts and software engineers.
- To create a model that consists of all well-recognized usability attributes of software.

3.2 Structure of QUIM

QUIM is a multi-layered hierarchical model like other software engineering models. It distinguishes five levels called factors, criteria, metrics, data, and artifacts (Figure 12). The relationship between these levels is an N-M relationship.

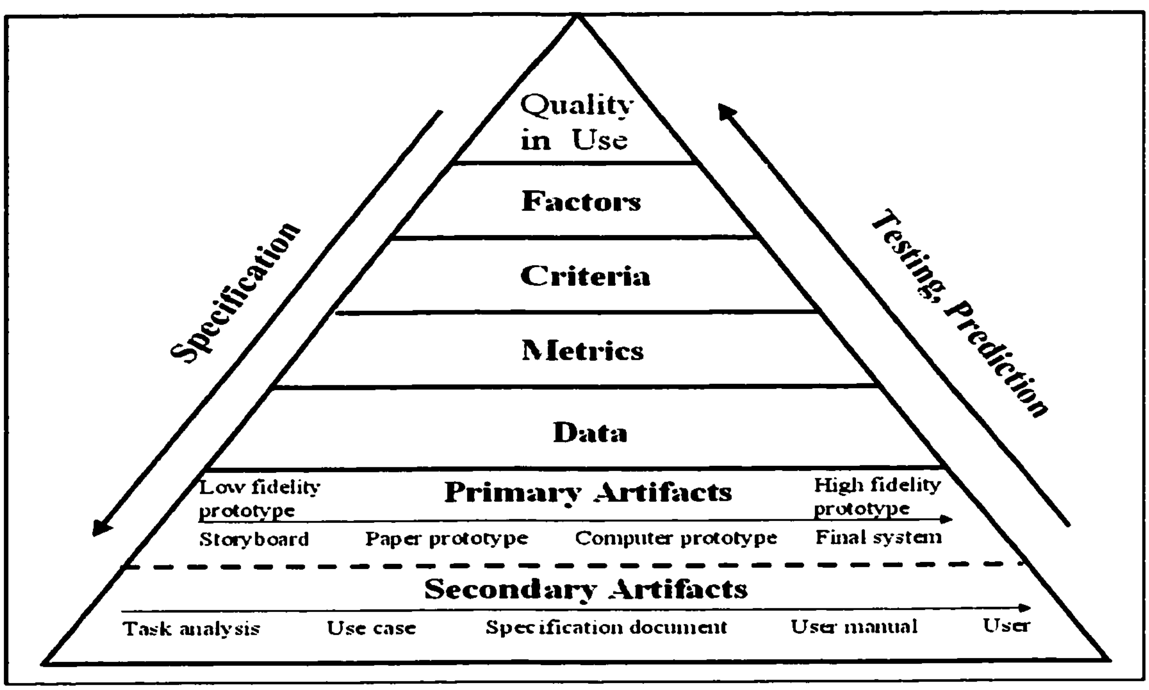


Figure 12: The hierarchy of QUIM

Here is the brief introduction of each of the term depicted in the structure.

- **Quality in Use**

At the top of the hierarchy is the quality in use, which is perception or feeling of the end user about software quality. The user is mainly interested in looking at the external observable aspects of the software product like the speedy performance, good layout etc. The user is not interested in the internal attributes of a software product.

- **Factors**

A quality in use factor is an attribute or characteristic of the user interface that the user sees in terms of the quality in use of the software product. The users define the quality of the software product in their own terms. It is not easy to measure and specify them. A Factor could be refined into the sub factors or Criteria. In QUIM, for the present study we have considered and investigated a set of 10 factors, they are: Efficiency, Effectiveness, Productivity, Satisfaction, Learnability, Safety, Trustfulness, Accessibility, Universality and Usefulness. They will be investigated in this thesis.

- **Criteria**

Similar to other models a criterion is a sub factor or sub characteristic of the software product. Criteria are specified and defined in the language of software developers, so it is hard for most of the users to understand the technical terms involved while defining them. A criterion can be measured by more than one metric. In QUIM, at present there are a total of 27 criteria that can be measured through a number of metrics.

- **Metrics**

As defined in previous chapter, it is a function whose output is a numeric value that summarizes the status of specific user interface characteristic. In QUIM, we have identified around 130 usability metrics, some of them are functions and described in terms of formula, others are just simple countable data.

- **Data**

The lowest layer of QUIM is the list of data that is used to calculate metrics. As it has already been explained in previous chapter that we are concerned about two types of data, countable and calculable. The examples of both types of data is presented as follows:

- Countable

For example: number of individual items on the screen, number of colors used or time spent while performing a particular task. The data collected from questionnaires also falls into this category.

- Calculable

For example: A metric given by (Bevan, Macleod, 1994) for task effectiveness (TE) is:

$TE = Quantity \times Quality / 100$

Where, Quantity is the percent of task completed and Quality is the percent of goal achieved.

The percent of task completed and percent of goal achieved are the calculable data that are used for computing this metric.

The other layers in QUIM are depicting the data collection methods that have already been listed in the previous chapter.

There are two directions for using the model, one is top-down approach and other is bottom-up.

- **Specification**

  In this top-down approach, firstly the usability goals are set and then the model helps in identification of responsible criteria, related metrics, and the required data in achieving the desired goals. The software developer can use these metrics to create a specification or test plan.

- **Testing and Prediction**

  The second approach is bottom-up which can be used for prediction. This approach helps in predicting how the software product goals vary with change in values of data set. It helps in predicting how increase in one metric value will affect the overall software product objective.

3.3 QUIM highlights

- QUIM is an integrated model and it supports quality activities in all the phases (like specification, designing, testing, maintenance etc.) of software development lifecycle.

- QUIM bridges both the software engineering models and HCI practices. Software engineering models like McCall and IEEE 1061 etc. provide the engineering view of the quality problem, whereas HCI practices which needs expert evaluation provide the subjective or heuristic view of software. In QUIM the engineering approach tells us the way to achieve the goal, if the results are not satisfactory then it suggests for redesigning the current model.
- QUIM is a large repository of well defined Factors, Criteria and Metrics based on quality in use. This model enables developers to view several metrics at once and it helps in reflecting those product characteristics that are important in controlling or evaluating the user interface part of the software development process. Viewing several metrics and criteria at one time also enables the developers to set a priority setting among them in order to accomplish their objective. The measurements help in predicting the future trends and requirements and it provides a complete balanced picture of the required software characteristics in order to achieve the desired goals.
- QUIM can be easily coupled to other software quality models. For example: the Performance Measurement method that is part of MUSiC project (Metrics for Usability Standards in Computing) developed by Esprit (European Information Technologies Program). QUIM can be used for context evaluation and for specifying the evaluation targets set in this method.

3.4 Relationships in QUIM

The relationships among the components of QUIM model hierarchy i.e. factors, criteria and metrics are very complex. The relationships in QUIM are not exactly like a tree, as shown in Figure 13.

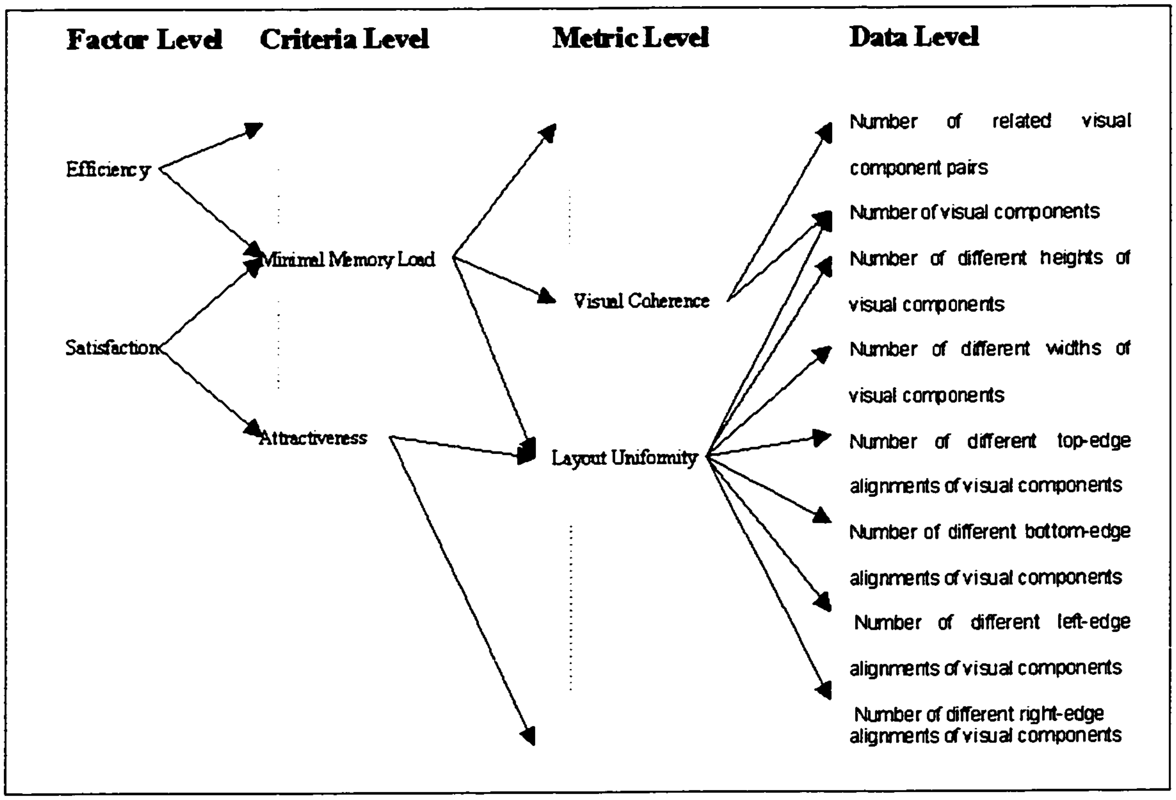


Figure 13: An example of relationships in QUIM

This is because a factor can be affected by more than one criterion that in turn can be measured by more than one metric. It is easier to understand this tree structure through the example shown in Figure 13. In this example, "Minimal Memory Load " is one of the criterions that affect two factors "Efficiency" as well as "Satisfaction". To measure Minimal memory Load there are a number of metrics; " Visual Coherence" and "Layout Uniformity" are shown in the figure". At the data level "Number of visual components" is the common input to these different metrics.

3.5 QUIM Editor

A tool has been developed by the human-centered software engineering (HCSE) group to integrate different usability standards, quality models in one centralized knowledge base. In its latest version 2.0 (Figure 14), it mainly offers two different kinds of functionalities, which are:

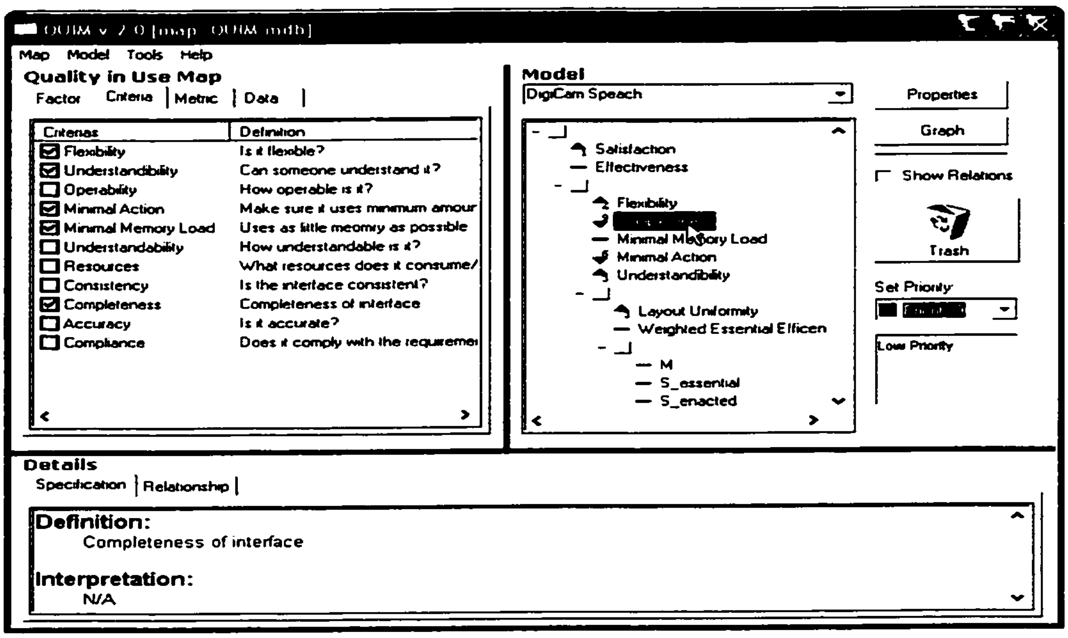


Figure 14: One view of QUIM editor (version 2.0)

- Acting as a quality in use map, it enables an administrator to add, delete or modify the quality in use factors, criteria, metrics and data to the database or repository. In this way it offers an environment to measure the different aspects of quality in use.
- Acting as a model viewer, it allows the users to easily access any usability standard or quality model that can be presented as QUIM 's interactive

measurement model map. It also helps the users to create their own model based on the stored database of quality in use factors, criteria and metrics. It has the tendency to visualize the relationships that exist between quality in use factors, criteria, metrics and data. There is an added functionality of QUIM editor also, which is to prioritize the factors, criteria, metrics and data as well so that most cost effective and appropriate software characteristics that meets the user goals could be recognized and measured first.

Chapter 4

Quality in Use Criteria

In QUIM, the ten factors under study are mapped to the following set of criteria. For each criterion a brief introduction along with the definition is given for better understanding of the concept.

4.1 Time Behavior

As the name implies, this criterion is related to the time spent on performing some usable task. The first and formal goal in any usability measurement of any software product is to observe the time spent by users to accomplish their goals.

The definition of Time Behavior as given in (ISO 9126-1,1998) can be modified in terms of quality in use and the revised definition is:

'The capability of the software product to provide appropriate task time when performing its function, under stated conditions.'

4. 2 Resource Utilization

Every software product consumes resources whether they are human, economical or the temporal ones. Resource utilization or the proper usage of the resources is expected of all the usable software products. (ISO 9126-4,1991) states that resources may include other software products, hardware facilities, materials, (e.g. print paper, floppy disks) and services of operating, maintaining or sustaining staff.

(ISO 9126-1,1998) defines resource utilization as:

'The capability of the software product to use appropriate amounts and types of resources when the software performs its function under stated conditions.'

4.3 Attractiveness

This criterion has more to deal with the external look and feel of the software product. The appearance of the software product particularly the screen design and color aspects are the measures to judge the attractiveness of a software product. Visually pleasant interfaces create users interest in the software product. It is the degree to which the users find a software product pleasant to use.

The definition of attractiveness as given by (ISO 9126-1,1998) is:

'The capability of the software product to be attractive to the user. This refers to attributes of the software intended to make the software more attractive to the user, such as the use of color and the nature of the graphical design.'

A useful metric to measure user interface attractiveness is "Layout uniformity (Constantine, Lockwood, 1999)". This metric assesses the uniformity or regularity of the user interface layout. It measures how well the visual components of interface are arranged.

Layout Uniformity (LU) is defined as:

$$LU = 100 \bullet \frac{(1-(N_h+N_w+N_t+N_l+N_b+N_r)-M)}{6 \bullet N_c - M}$$

Where,

M is an adjustment for the minimum number of possible alignments and sizes needed to make the value of LU range from 0 to 100, given by

$M = 2 + 2 \cdot \lceil 2\sqrt{N_{components}} \rceil$

N_c=Number of components

N_h= Number of different heights of visual components

N_w= Number of different widths of visual components

N_t = Number of different top-edge alignments of visual components

N_b= Number of different bottom-edge alignments of visual components

N_l = Number of different left-edge alignments of visual components

N_r= Number of different right-edge alignments of visual components

$\lceil \rceil$, It is the ceiling function; which means the smallest integer greater than the enclosed value.

Interpretations

0 <= LU <= 100

A review of well-designed dialogues suggests that, in general, a LU value anywhere between 50% and 85% is reasonable.

4.4 Likeability

Likeability is the personal feelings of the user about the software product. User may likes or dislikes a product depending on his/her perceptions about the software product. Every user has a different point of interest in the software product and he/she may see the product quality in terms of his/her own certain sets of attributes.

Rubin (1994) states that attitude or likeability refers to the user's perceptions, feelings, and opinions of the product, usually captured through both written and oral interrogation.

4.5 Flexibility

Flexibility is the user interface plasticity offered by the software product to its users. A flexible user interface can accommodate a variety of changes for variety of users. A flexible software product can be customized to user's personal choices and it is really what the users now a days expect of most of the software products.

There are many definitions of flexibility like:

It is the extent to which the software can accommodate changes or the variety of ways that users can obtain satisfactory results from the application.

'Adaptation to some specified percentage variation in tasks and/or environment beyond those first specified.' (Shackel, 1991 and Preece et al., 1994)

(Wixon, Wilson, 1997) define it as:

'The extent to which a product can be applied to different kinds of tasks.'

In terms of user interface, flexibility involves providing the opportunity for users to tailor interactive elements to fit their personal preferences.

4.6 Minimal Action

Every user wants to accomplish his/her goals as soon as possible and with as little effort as possible. Minimal Action is needed by all users of a software product. Software professionals have to keep in mind this very important criterion to make a product easy to use. Minimal Action not only makes users happy but it influences the overall product quality also.

As the name implies, it is the capability of a software product to help users achieve their tasks in minimal steps.

(Lin et al.,1997) describe minimal action as :

'The extent to which user needs to take minimal effort to achieve a specific task.'

4.7 Minimal Memory Load

Usability studies reveal that the software product should enable users to do their jobs without pressurizing their minds. This is one of the heuristics in the usability evaluation of software products. (Tom et al., 2002) define three primary human memory issues as:

- number of items that have to be remembered,
- the time frame for which the items have to be remembered, and
- similarity among the remembered items.

Minimal memory load means that the user should not be overburdened and should be able to accomplish his/her task easily.

(Lin et al.1997) state that it is the extent to which user needs to keep minimal amount of information in mind to achieve a specified task.

4.8 Operability

Many usability professionals define operability as one of the criteria for judging the usability of a software product. Operability is a measure of whether operating on a software product is easier or not. It has more to do with functionality offered by the software product.

(ISO/IEC 9126,1991) defines it as:

'Attributes of software that bear on the users' effort for operation and operation control.'

4.9 User Guidance

User guidance is regarded as a pervasive and integral part of interface design .A study by Magers (1983) has demonstrated that good user guidance can result in faster task performance, fewer errors, greater user satisfaction, and will allow accomplishment of information handling tasks otherwise impossible for novice users.

It means that the interface should provide context-sensitive user help facilities and also provide meaningful feedback when errors occur.

'It indicates how the interface helps the user to use the application.' (Lin et al., 1997)

4.10 Understandability

Users can use an application more effectively when it is easier to understand because they know a greater portion of the application's functionality. This also leads to more efficiency, because the users can use functionality that achieves goals faster, with fewer steps and less errors. Understandability and learnability are closely related. The easier an application is to understand, the easier it is to learn and relearn.

(Boehm et al., 1978) defines it as:

'The degree to which the purpose of the system or component is clear to the evaluator.'

'Attributes of software that bear on the users' effort for recognizing the logical concept and its applicability.' (ISO/IEC 9126,1991)

(ISO/IEC 9126 -1, 1998) states that understandability is the capability of a 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.

This attribute has more to do with provision of the right functionality; e.g. matching the software to the user's needs.

4.11 Consistency

Consistency is the desirable property of any software product. A consistent software product is the one that has an appealing look and feel and is easier for the user to operate because of the ease of remembrance and similarity of terminology on all screens.

4.11.1 Arguments in favor of Consistency

- Consistency is one of those principles that sounds like a great idea when you first come across it, but it is very difficult to apply the principle in any real situation where there is a wide array of conflicting things which you can be consistent. (Gentner, Nielsen, 1996)
- 'Consistency is the key to usable interaction design,' says Jakob Nielsen [71].
- Ben Shneiderman [86] says that the first golden rule for dialog design is to strive for consistency.
- 'Consistency is really something that is in they eye of the beholder, always with respect to some outside measure.'(Carson, 1999)

4.11.2 Consistency Definitions

Here are a few definitions of Consistency that is considered as an important trait of all usable interfaces.

- 'Agreement or harmony of parts or features to one another or a whole.' (Merriam-Webster, 2001)
- 'The degree of uniformity, standardization, and freedom from contradiction among the documents or parts of a system or component.' (IEEE, 1990)
- 'Consistency means that similar user actions lead to similar results.' (Wolf, 1989)
- 'Consistency refers to common actions, sequences, terms, units, layouts, colors, typography and more within an application program...' (Shneiderman, 1992)
- 'Attributes that bear on the visual uniformity of user interface.' (Lin et al.,1997)

4.11.3 Types of Consistency

Grudin (1989) in his article "The Case Against User Interface Consistency" states that there are three types of user interface consistency, which are:

- The internal consistency of a design with itself

  User interface designers deal with internal design consistency. Consistency might be sought in physical and graphic layout, command naming and use, selection techniques, dialogue forms, etc.

- The external consistency of a design with other interface designs familiar to a user.
- An external analogical or metaphoric correspondence of a design to features in the world beyond the computer domain.

4.12 Self- Descriptiveness

Presence of self-description makes users comfortable and they feel safe and secure while operating the software product.

It is the ability of the software product to introduce itself and its purpose, as well as provide users clear and concise assists for its correct operation.

4.13 Feedback

User feedback also affects the usability of the software product because while interacting online, the users may need visual change of the state or may need confirmation of their actions. The software products that provide an immediate, accessible feedback are the successful ones.

In simple terms, it is the response of the system to user actions.

Feedback measurement

A metric example to measure website feedback is "Freshness factor (Sterne, 2001)." Freshness factor is a measure of whether the regular changes are made to the site. It is needed for two reasons: 1) Visitors want to see fresh material, and 2) Updating the site often will ensure that the information is current. It is given by the formula as:

Freshness Factor (FF)= (Average content area refresh rate) / (Average session visit frequency)

Interpretations

1<= FF<=1.5

A freshness factor (FF) less than 1 means visitors see stale content. They visit more often than the content changes.

A freshness factor (FF) greater than 1.5 means the content changes more often than visitors show up to view it. That means resources are not consumed properly.

4.14 Accuracy

Accuracy is related to the correctness in the task output as judged by the user of the software product. Accuracy is a wish list feature of any software product.

(ISO/IEC 9126-1,1998) defines it as:

'The capability of the software product to provide the right or agreed results or effects. '

4.15 Completeness

Completeness means how much of the work has been finished. It is a criterion that is used to measure the effectiveness of any software product.

According to (Constantine, Lockwood, 1999) completeness is simply the percent of total assigned work completed within the allotted time.

In simple terms, it is the extent to which the user can complete a specified task.

4.16 Fault Tolerance

For a product to be usable it should have the fault tolerance capability.

Here, we want to differentiate fault tolerance from safety. Fault tolerance implies that the software product can recover from or in some way tolerate the error and continue correct operation, whereas safety implies that the software product either continues correct operation or fails in a safe manner. Fault tolerance and safety both require good error detection; it is the response to errors that differentiates the two approaches. A safe failure is an inability to tolerate the fault.

Fault Tolerance Definition

Software fault tolerance is the ability for software to detect and recover from a fault that is happening or has already happened in either the software or hardware in the system in which the software is running in order to provide service in accordance with the specification.

'The capability of the software product to maintain a specified level of performance in cases of software faults or of infringement of its specified interface.' (ISO/IEC 9126-1,1998)

4.17 Resource Safety

It is one of the criteria that lead to the safety attribute of software product. It deals with the safety of human resources. This criterion mainly handles those aspects of safety that are not addressed by other safety related criteria.

It can be defined as the capability of software product to handle resources properly without any hazard or mishap to people in its context of use.

4.18 Readability

Readability, or how easily the contents of web pages can be read, is seen as one of the criteria in QUIM because it is an important issue for all users. The majority of the user time is spent on reading the online information when they access any website. A large number of factors affect the readability of online information including the relative contrast between text characters, background color, page width etc. 'Readability' differs from 'Legibility', which is the ease with which content can be read i.e. how clear the text is visually.

Readability is the ease with which visual content can be understood or it is the degree to which the meaning of text is accessible.

Readability measurement

An example of a metric, which can be used to measure readability of hypertext documents, is "Flesch Reading Ease Score" [77].

It rates text on a 100-point scale; the higher the score, the easier it is to understand the document. It is given by the following formula:

Flesch Reading Ease score = 206.835 – (1.015 x ASL) – (84.6 x ASW)

Where, ASL = average sentence length (the number of words divided by the number of sentences)

ASW = average number of syllables per word (the number of syllables divided by the number of words)

Interpretations

For most standard documents, aim is to score approximately 60 to 70.

4.19 Controllability

It is one of the usability heuristic and it is used to measure the user control of the software product while using it.

In simple terms, it is the degree to which the users feel that they are in-charge of the software product.

4.20 Navigability

Navigation is often taken for granted but plays a big role in web site usability. A site's usability is greatly affected by whether or not a user gets lost on a site or if he/she can easily retrieve the information that they are seeking. An organization's business fully depends upon the capability of navigation mechanism within a website. Navigation helps in getting site visitors to view more than just the home page. If navigation choices are unclear, visitors may select to hit the "Back" button on their first (and final) visit to a web site.

A few basic definitions of Navigability can be defined as:

- It is the degree to which a user can move around in the application.
- It is the ability to maneuver within a site.
- It is the ability of an interface to focus attention on the appropriate materials and to lead one through the material.

McGovern (2000) in his article about web site navigation says that the purpose of navigation is to:

1. Present readers with the user-friendliest path through the classification so that they can find the content they want quickly.
2. Ensure readers always know where they are on the site.
3. Allow readers to move quickly and logically through the web site.
4. Give readers the proper context of the document they are reading.
5. Highlight for the reader parts of the classification that the organization wants to promote.

Navigability measurement

One of the metric, which can be used to measure web navigability, is "Self-referential pages (Author)". This metric gives us a measure of the number of pages on a website that have links back to themselves and it is given by:

X=A/B

Where,

A= Number of pages having links to themselves

B= Total number of pages of a website

Interpretations

X<=0

The closer to 0 is the better.

4.21 Simplicity

The word "simplicity" can be interpreted through three dimensions - functionality reduction, understandability and ease of use of application. The central idea behind simplicity is that users will feel more pleasure in their experience and have more positive reactions to a software product.

'Simplicity for the web user means lack of complexity or lack of obstruction in accomplishing the defined goals.' (Nielsen, 1999)

It means to eliminate the extraneous and enhance the user experience, while at the same time not sacrificing the quantity of information.

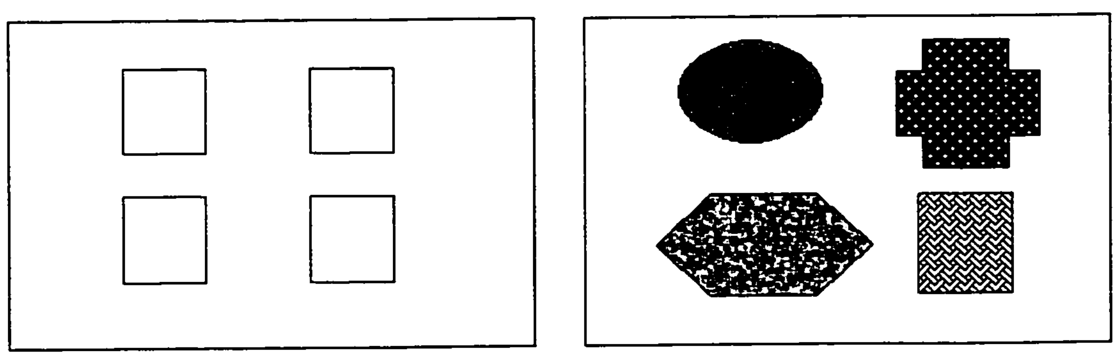
4.21.1 Types of Simplicity

The design principles set by Cognetics Corporation [98] states that several types of simplicity contribute to a well-designed user interface, which are:

- *Visual simplicity* is achieved by showing only the most important controls and objects. Screen layout should follow good visual design practices. Use white space as a visual element to define perceptual areas.
- *Verbal simplicity* comes from the use of direct, active, positive language.
- *Task simplicity* is achieved when related tasks are grouped together, and only a few choices are offered at any one time.
- *Conceptual simplicity* is accomplished by using natural mappings and semantics, and by using progressive disclosure.

Simplicity measurement

"Economy" is a metric proposed by Ngo and Teo [69] and it gives us some measure of the simplicity of user interface. Economy is a measure that tells us how economical the screen is, as shown in Figure 15.



Screen Economy (Good) Screen Economy (Bad)

Figure15: Comparison of screen layouts

Screen Economy can be measured by a formula as follows:

$$ECM = 3/(N_{size} + N_{color} + N_{shape})$$

Where,

N_{size}= Numbers of sizes used

N_{color} = Numbers of color used

N_{shape} = Numbers of shapes used

Interpretations

Economy is achieved by using few styles, display techniques, and colors as possible.

0<= ECM <=1

4.22 Privacy

With the global access to worldwide information, the main concern of the users while using the Internet now a day is secure access to their personal information. More and more people are now recognizing the value of personal privacy and they are considering it as one of the criteria before taking any action on the Internet.

It means that the use of the information one provides is guaranteed to be used for no purpose other than what one gave it for, without their approval.

In all, it is the ability of the software product to handle user's personal information and disclosing it to the third party only with the user's consent.

4.23 Security

Throughout the world, one of the principal inhibitors to the growth of eCommerce is concern about the security of doing business on the Internet. Websites must be resistant to malicious attacks by Internet users, safeguarding both site and user confidential data. It has been observed in various eCommerce surveys that fear of credit card fraud is a major deterrent to online shopping.

Security is the basic need to attract online customers for eCommerce. It is the basic building block for establishing trust in eCommerce site.

(ISO/IEC 12207,1995) defines it as:

'The capability of the software product to protect information and data so that unauthorized persons or systems cannot read or modify them and authorized persons or systems are not denied access to them.'

4.24 Insurance

To trust software product users need some guarantee of the use of their personal information. They feel safer and secure if the software product accommodates some form of insurance in case of some fraud/ hacking occurs. Users want the software product especially the eCommerce sites to provide some form of insurance in case their decision to buy online changes.

Insurance is the liability of the software product vendors in case of fraudulent use of the users personal information; or the users change their minds while shopping online in eCommerce sites.

4.25 Familiarity

Familiarity is a specific activity-based cognizance based on previous experience or learning of how to use the particular interface. If the site has a familiar feel to it, the users can be surer of what they are doing. They feel safe clicking on links, knowing that something unexpected won't happen. They get the feeling of control.

The dictionary definition of familiarity is:

'Reasonable knowledge or acquaintance, as with a subject or place.' (The

Collins English Dictionary, 1998)

The usability company.com defines it as:

'The degree of correlation between the user's existing knowledge and the knowledge required for effective interaction with a system.'

In terms of user interface, it is the degree to which a user recognizes user interface components and views their interaction as natural; the similarity of the interface to concrete objects the user has interacted with in the past.

In all, for the user interface to be familiar to the user it should use terms and concepts, which are drawn from the experience of users who will make most use of the system.

4.26 Loading Time

According to (GVU7, 1997) survey, 77% of the web users consider loading speed or loading time of web pages as the most annoying factor. It has been reported that users are more likely to lose interest in a site if the download time exceeds 10 seconds (Nielson, 1993). Therefore loading time is also one of the criteria that contribute to website's usability. There are many factors that affect the page loading time like the page size, the server speed, connection speed, rate at which they travel on the network i.e. bandwidth etc. Here, it is cleared that for the present study we are looking at those elements of the web sites that affect the loading time so that their measurement could contribute to total loading time. In this way, our measurement of the loading time is only partial because we are evaluating websites from the user point of view excluding those factors that are out of the scope of the user like server speed, network bandwidth etc.

Loading Time can be defined as follows:

- Load time is the time it takes for a web page to load i.e. how fast a web site responds to a user.
- 'The load time is measured from the moment when a user clicks on a selection to the point when all the new elements of the page have been downloaded to the user's computer. ' Web Criteria [62]
- 'Download time is the amount of time it takes for a Web client machine to receive and display a data file submitted by a Web server after that file was requested by the client.' (Rose, 1998)

4.27 Presentation

The presentation of information on a site is important not only because it makes the first impression on the user, but also because it affects how easily the information can be accessed and documented.

(McClure et al. 1997) have defined presentation issues as consisting of:

1. Where information was located?
2. How graphics were used?
3. What elements on the site were distracting?

4.28 Criteria / Metrics mapping matrix

Figure 16 shows the partial mapping of criteria to metrics (because the total number of metrics that are identified is a long list presented in appendix section of thesis).

CRITERIAS \ METRICS	Task Time [19]	Layout Uniformity [14]	Visual Coherence [14]	Broken Link Count [74]	Task Concordance [14]	Horizontal/Vertical [82] Balance	Number of colors	Local Density [97]	Images with alternative text [74]
Time Behavior	•								
Resource Utilization	•								
Attractiveness		•				•			
Likeability									
Flexibility									•
Minimal Action	•				•				
Minimal Memory Load		•	•						
Operability	•								
User Guidance			•						
Understandability								•	
Consistency		•						•	
Self-Descriptiveness									
Feedback									
Accuracy									
Completeness				•					
Fault- Tolerance									
Resource Safety									
Readability									
Controllability									
Navigability				•					
Simplicity					•				
Privacy									
Security									
Insurance									
Familiarity									
Loading Time							•		
Presentation		•	•			•	•		

Figure 16: Criteria/Metrics mapping matrix

Chapter 5

Quality In Use Factors

5.1 Introduction

Our research scope was limited to studying a list of ten factors:

1. Efficiency
2. Effectiveness
3. Satisfaction
4. Productivity
5. Learnability
6. Safety
7. Trustfulness
8. Accessibility
9. Universality
10. Usefulness

An extensive study of the various papers, published on quality and especially on usability by many worldwide usability professionals and organizations, was conducted. In the first step, we explored whether these factors were really quality in use factors or not. In the initial study, we first added "understandability" as one of the factors in our research list and then later on we realized that it is in fact a criterion that is affecting many other factors like-efficiency, satisfaction, learnability etc.

5.2 Efficiency

As we have seen in chapter 3, Efficiency is one of the main factors in many existing quality models. Moreover, it is also the basic quality in use characteristic.

According to Nielsen (1993), efficiency refers to an expert user's steady-state level of performance at the time when the learning curve flattens out. Nielsen relates this attribute to progress in the learning curve, therefore establishing its relation with the learnability factor.

(Hix, Hartson, 1993) and (Wixon, Wilson, 1997) define efficiency as long-term performance, therefore associating it somehow with expert users as well.

(Constantine, Lockwood, 1999) mention efficiency as the level of user productivity while using the system.

In all these definitions, this attribute does not relate to the efficiency of the computer on its own, but to the efficiency of the user in his/her interaction with computer. All these authors consider only time/speed of performance as the only resource that is expended whereas the resources include many things as defined by ISO 9126 (1991).

(ISO/IEC 9126,1991) provides a more general definition of efficiency as:

> Efficiency is the capability of the software product to provide appropriate performance, relative to the amount of resources used, under stated conditions. The resources may include other software products, hardware facilities, materials, (e.g. print paper, floppy disks) and services of operating, maintaining or sustaining staff.

(ISO/IEC 9241, 1998) states that efficiency is the relation between (1) the accuracy and completeness with which users achieve certain goals and (2) the resources expended in achieving them.

Bevan (1995) suggests that efficiency relates the level of effectiveness achieved to the quantity of resources expended. The efficiency is further based on resources like mental or physical effort, time and financial cost. These resources are used to give measures of human efficiency, temporal efficiency, and economic efficiency respectively.

For example:

Human Efficiency = Effectiveness/ Effort

Temporal Efficiency = Effectiveness/ Task Time

Economic Efficiency = Effectiveness/Total Cost

The total cost is the summation of the labor costs of the user's time, the cost of the resources and the equipment used, the cost of any training required by the user etc.

We can summarise efficiency as the number of tasks per unit of time that the user can perform using the system.

In all, it is concluded that efficiency is the capability of the product to enable users to expend appropriate amounts of resources in relation to the effectiveness achieved in a specified context of use.

Criteria

(ISO/IEC 9126 –1,1998) identifies the following as the main criteria to measure efficiency:

- *Time behaviour*
- *Resource behaviour*

The other criteria that are recognised to measure efficiency include:

- *Minimal Action*: If the task requires minimal number of steps then the overall user efficiency increases.
- *Minimal Memory Load*: The users do not want to remember the hard-cored terminology or difficult commands to operate a software product. The software product that offers minimal demands on the users remembrance power is more efficient because they can operate it easily and more effectively.
- *Operability*: A more easier to operate and control software is more efficient.
- *Understandability*: If the users understand the functionality of the software product, they will operate it more efficiently.
- *Feedback*: The software product that provides the user feedback immediately helps in achieving their objectives more frequently.
- *Navigability*: This attribute is more related to the websites, if they offer well navigation mechanisms then the users can achieve their actions more easily and thus it indirectly contributes to the efficiency of the websites.
- *Loading Time*: This is again a characteristic of websites; if the pages take

less time to load then they the users do their tasks more efficiently.

The mapping of the various criteria to the efficiency is depicted in Figure 17.

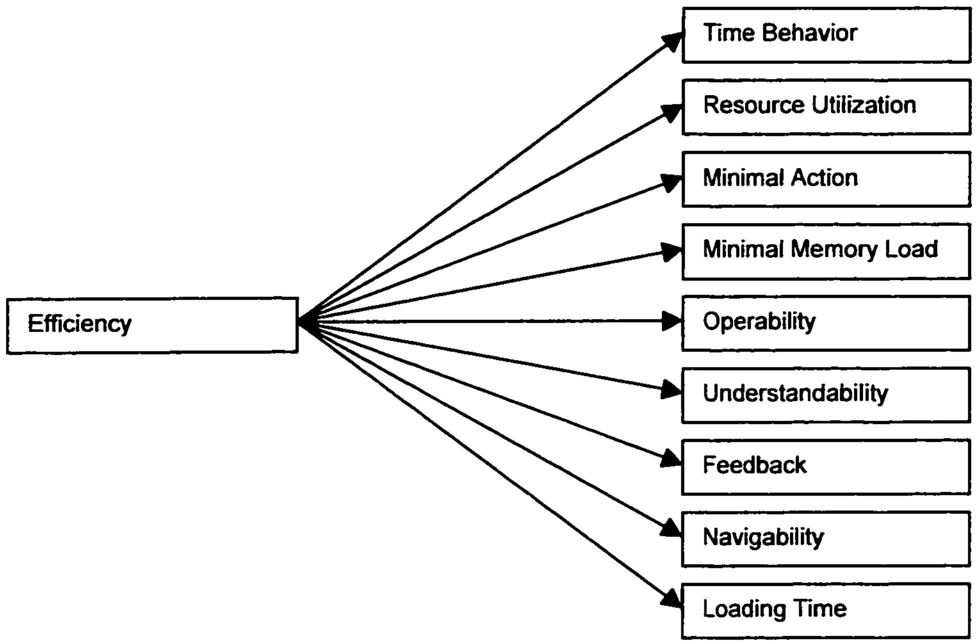


Figure 17: Mapping efficiency to various criteria

Use of efficiency measures

Bevan (1995) says that the efficiency measures can be used to compare the efficiency of:

- Two or more similar products, or versions of a product, for the same context of use (i.e. when used by the same user groups for the same tasks in the same environments).
- Two or more types of users who use the same product for the same tasks in the same environment.

- Two or more tasks when carried out by the same users on the same product in the same environment.

Efficiency measures provide a more general measure of rate of work done in terms of the quantity and quality of output against time.

5.3 Effectiveness

ISO/IEC 9126-4 defines effectiveness as:

> 'The capability of the software product to enable users to achieve specified goals with accuracy and completeness in a specified context of use.'

(Macleod et al., 1997) defines effectiveness as the correctness and completeness with which the goals are achieved in a context. It is similar to the ISO definition; only the accuracy term is replaced by the correctness of the output the software product is providing.

The accuracy or the correctness of the output is a wish list feature of the software product that is sometimes ignored because of the cost involved with it.

To understand the meaning of accuracy and completeness an example is given by (Macleod et al., 1997) as follows:

Suppose the desired goal is to transcribe a 2-page document into a specified format then accuracy can be measured in terms of the number of spelling mistakes and the number of deviations from the specified format, and completeness can be formulated as the number of words of the document transcribed divided by the number of words in the source document.

According to (Macleod et al., 1997), the effectiveness with which the users accomplish their task (i.e. task Effectiveness) while using the product comprises of two components: the quantity of the task that the user completes and the quality of the goals that are achieved and it is given by the following formula:

Task Effectiveness = 1/100 (Quantity x Quality) %

Criteria

In the definition, we have seen that the accuracy and the completeness are the criteria that are affecting the effectiveness of the software product. There are some other criteria also that are indirectly influencing effectiveness like Flexibility, Consistency, Feedback and Navigability (shown in Figure 18).

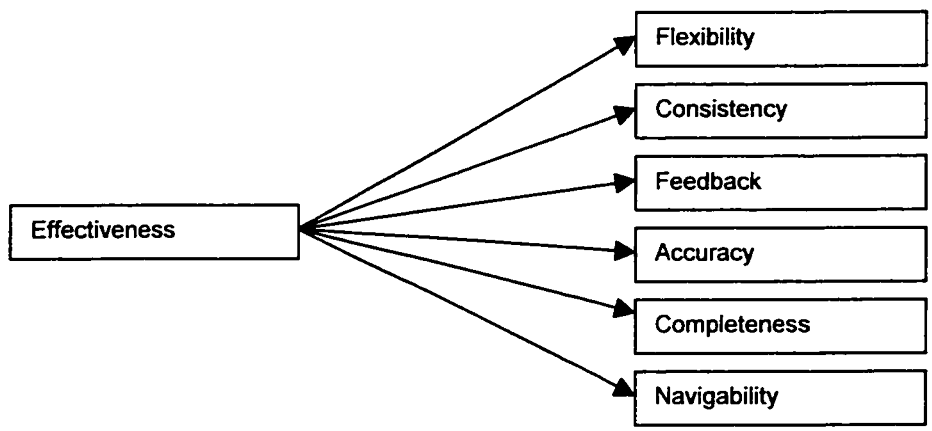


Figure 18: Mapping effectiveness to various criteria

5.4 Productivity

Like effectiveness, productivity is also one of quality in use characteristics of any software product as defined by (ISO/IEC 9126-4,2000). In simple terms, it is rate of output per unit of input.

(ISO/IEC 9126-4,2000) describes productivity as the measure of the level of effectiveness achieved with respect to the resources that are expended.

In this definition it appears to be analogous to efficiency, however they both are separate terms, productivity deals with the useful productive output that is obtained as a result of user interaction with the software product. Not all the user actions are productive, there are generally two types of user task actions one that is productive and the other is unproductive. The productive user task actions are those that contribute to the task output, whereas the unproductive don't. (Macleod et al., 1997) describe three types of unproductive task actions:

1. *Help actions:* The user asks for contextual or online help information/ documentation.
2. *Search actions:* The users explore the software product functionalities without activating them. For example: Opening and then closing a print dialog box.
3. *Snag actions:* These are those user or system actions that unlike the help or search actions don't contribute to directly or indirectly to the task output. These are further classified into negating actions, cancelled actions, and rejected actions. For example: user pastes a picture on the document and then cuts it.

The productive time period is thus the total time spent on performing the task minus the unproductive time.

Our definition of productivity is:

The level of effectiveness achieved in relation to the resources (i.e. time to complete tasks, user efforts, materials or financial cost of usage) consumed by the users and the system while using a software product in a specified context of use.

Criteria

The productivity of any software product can be measured in terms of time behavior, resource utilization properties. In the Figure 19, we have added one more criterion Loading Time, which is the page loading time of websites that directly influence the user or website's productivity.

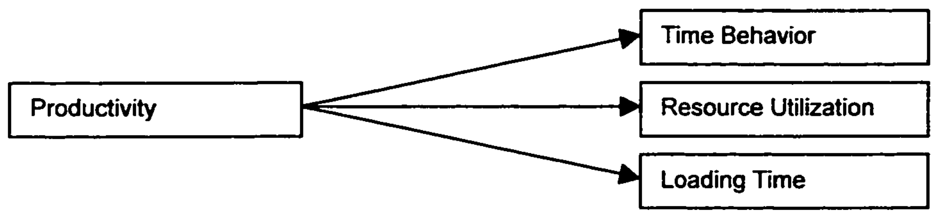


Figure 19: Mapping productivity to criteria

5.5 Satisfaction

User Satisfaction is the predominant factor behind the success of any software product. Many usability experts have considered satisfaction as an important factor for product usability because any usable product should first satisfy the demands of the users. User satisfaction may be an important correlate of motivation to use a product and may affect user performance in some cases. Satisfaction is the most elusive usability attribute that is completely dependant on subjective opinion of users.

Definitions

Satisfaction measures the extent of users freedom from discomfort and their positive attitudes towards the use of the product.

(ISO/IEC 9241-11,1998) describes it as:

'The comfort and acceptability of the work system to its users and other people affected by its use.'

Here, 'comfort' refers to overall physiological or emotional responses to use of the software product (i.e. if the user feels good, warm, and pleased, or tense and uncomfortable). 'Acceptability of use' may measure the overall attitude towards the software product, or the user's perception of specific aspects like: the user feels that the software product supports the way he/she carry out his/her tasks, he/she feels in command of the software product, the software product is helpful and easy to learn etc.

Satisfaction describes the subjective response of user while using a software product in a specified context of use.

5.5.1 Measures of satisfaction

Measures of satisfaction provide a useful indication of the user's perception of the software product usability and the acceptability of the product by the people who use it and other people affected by its use. Measures of satisfaction may be for the overall software product or for the specific aspects of the software product only. Satisfaction can be specified and measured by subjective attitude rating scales such as: liking for the product, satisfaction with product use etc. Subjective measures of satisfaction are produced by quantifying the strength of a user's subjectively expressed reactions, attitudes, or opinions. This process of quantification can be done in a number of ways, for example, by asking the user to give a number corresponding to the strength of their feeling at any particular moment, or by asking users to rank products in order of preference, or by using an attitude scale based on a questionnaire.

Other measures of satisfaction that can be assessed indirectly are: the ratio of favorable to unfavorable users comments during use, number of users preferring the system, number of regressive behaviors etc.

5.5.2 Satisfaction Questionnaires

Questionnaires to measure satisfaction and associated attitudes are commonly built using Likert scale. A Likert scale is simply one based on forced-choice questions, where a statement is made and the respondent then indicates the degree of agreement or disagreement with the statement on a 5 (or 7) point scale. This type of attitude scale, when properly developed, has the advantage that it can be quick to use, has known reliability and does not require special skills to apply.

There are a number of questionnaires to measure satisfaction for example:

1. **Overall satisfaction:** (Brooke, 1996)

   SUS (System Usability Scale) is a 10-item questionnaire that gives an overview of satisfaction with software. It was developed by John Brooke in 1986, and is freely available.

2. **Satisfaction profile:** (Kirakowski, 1996)

   SUMI (Software Usability Measurement Inventory) is a 50-item questionnaire that measures five aspects of user satisfaction (Likeability, Efficiency, Helpfulness, Control and Learnability), and scores them against expected industry norms.

3. **User interface satisfaction:** (Lewis, 1991b)

QUIS (Questionnaire for User Interaction Satisfaction) is similar to SUMI, but measures attitude towards eleven interface factors (screen factors, terminology and system feedback, learning factors, system capabilities, technical manuals, on-line tutorials, multimedia, voice recognition, virtual environments, internet access, and software installation). It does not have industry norms.

4. **Computer usability satisfaction:** (Lewis, 1995a)

Computer User Satisfaction Inventory (CUSI) is a short questionnaire of 22 items. Two subscales of usability were established, called at the time Affect (the degree to which users like the computer system) and Competence (the degree to which users feel supported by the computer system).

5. **After scenario questionnaire:** (Lewis, 1991)

There are a total of 3 questions and it is developed by Lewis and is available freely.

Criteria

The criteria that directly affect the satisfaction of any software product are sketched in Figure 20.

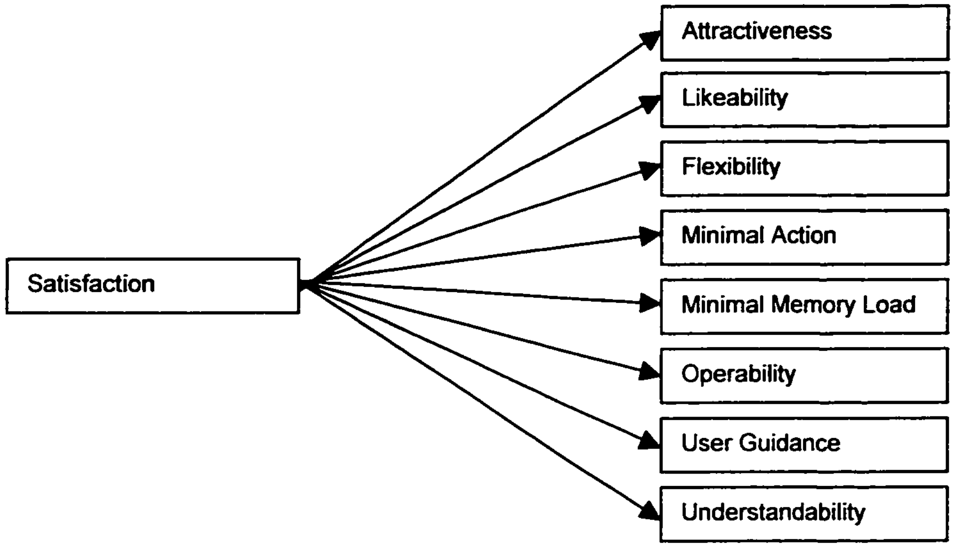


Figure 20: Criteria affecting satisfaction

5.6 Learnability

Learnability is closely related to understandability and understandability measurements are the indicators of the learnability potential of the software product.

Definitions

'The capability of a software product to enable the user to learn its application.' (ISO/IEC 9126-1,1998)

'The ease with which new users can begin effective interaction and achieve maximal performance.' (Dix et al., 1993)

It is the capability of the software product to enable users to feel that they can get to use the software product for the first time, and can learn to use other facilities or access other information once they have started using it.

In all, it is the ease with which a user can master the required features for achieving a certain goal in a certain context of use.

Criteria

It has been studied by many usability professionals that a software product has a good learnability if it fulfills the following requirements:

- The users can easily *understand* its functionality
- It needs *minimal user actions* to do a particular task
- It requires *minimal memory load* on the users i.e. it does not have a hard terminology.
- It obeys the *consistency* rule while designing the user interface so that the users find it easier while exploring different screens.
- It provides context based help whenever required and provides *user guidance* in case of any error message.
- It is *self-descriptive* i.e. it explains its purpose well to the audience.
- It is *simple* in terms of functionality.
- It provides a *familiar touch* to the users so that they feel that it resembles in functionality to other software products that they have already used.

The pictorial representation of the mapping learnability criteria is shown in Figure 21.

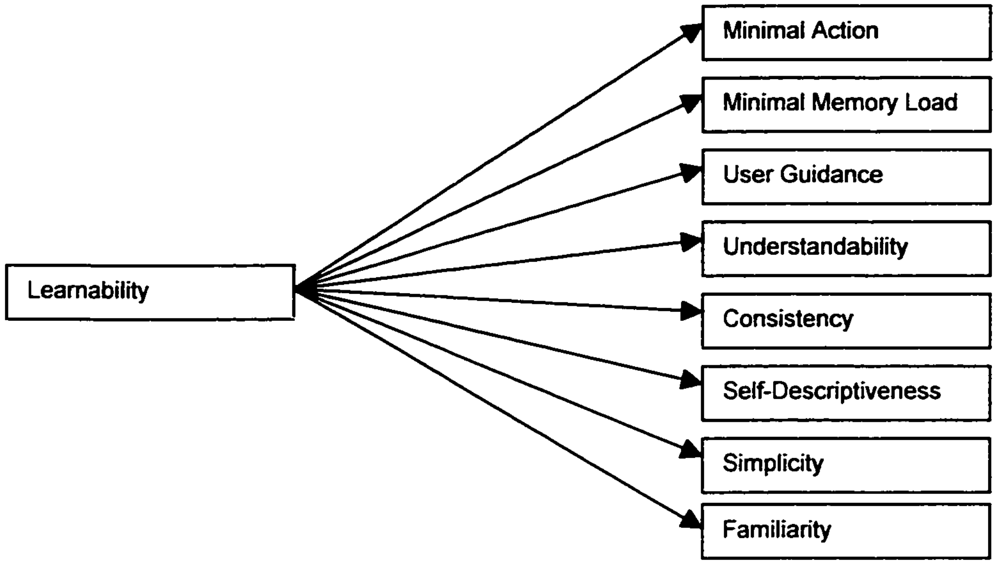


Figure 21: Mapping learnability

5.7 Safety

(ISO/IEC 9126–4,2000) has described safety as another quality in use characteristics of any software product.

(ISO/IEC 9126-4,2000) defines safety as the measure of level of risk of harm to people, business, software, property or the environment in a specified context of use. It includes the health and safety issues of the both the user and those affected by use, and unintended physical or economic consequences as well.

(ISO/IEC 9126-4,2000) states that there are two ways to analyze safety of the scftware products, which are:

- *Operational safety*: It is the ability of the software product to meet the user requirements during normal operation without harm to other resources and the environment.

- *Contingency safety*: It is the capability of the software product to operate outside its normal operation and divert resources to prevent any intended risks.

Our definition of safety is:

The degree to which a software product limits the risk of harm to people in a specified context of use.

Criteria

Figure 22 shows the mapping of various criteria to safety factor. Software product safety can be measured and analyzed through its various characteristics like - consistency, accuracy, completeness, fault tolerance, security and insurance.

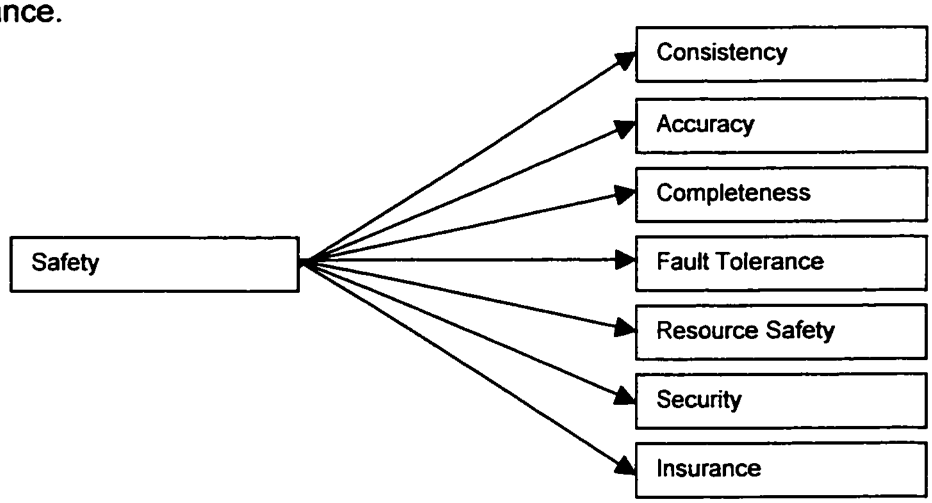


Figure 22: Mapping safety

5.8 Trustfulness

The concept of trust has been widely studied by researchers in many areas. However, it remains a difficult concept to define because of its dynamic, evolving and multi-facet nature. This factor affects every type of software product and it is a complex subject to study; therefore we limited our search to trustfulness of eCommerce websites only. In eCommerce, Trust is a subjective judgment made by the user, based on general experience learned from being a customer. The development of e-commerce cannot reach its potential without trust. In simple terms, trust can be defined as the belief by one party about another party that the other party will behave in a predictable manner (Luhmann, 1979).

Definitions

(Kini , Choobineh, 1998), state that trust, as defined in the Webster dictionary is:

- An assumed reliance on some person or thing. A confident dependence on the character, ability, strength, or truth of someone or something.
- A charge or duty imposed in faith or confidence or as a condition of a relationship.
- To place confidence (in an entity).

Grandison and Sloman [27] define trust as 'the firm belief in the competence of an entity to act dependably, securely, and reliably within a specified context' (assuming dependability covers reliability and timeliness).

The substantive definition of 'trust' in the Oxford English Dictionary [75] is 'confidence in or reliance on some quality or attribute of a person or a thing, or the truth of a statement.'

Trustfulness is the degree of faithfulness a software product offers to its users in a certain context of use.

Criteria

Trustfulness in eCommerce is a very deep research topic, although we tried to seek the criteria responsible for establishing trust in eCommerce. This topic needs more investigation. By thoroughly studying the various papers and research articles on trust in eCommerce (references are provided), the following criteria are supposed to affect the trustfulness.

1. *Security:* Security is number one essential to establishing trustfulness in e-commerce sites. In executing electronic transactions over the Internet, the users or the customers are always concerned about the security of their credit card. Various usability professionals (for example: Friedman et al. [24], Tilson et al. [94], Ahuja [1] and many more) think this issue as the basic building block of trustfulness in eCommerce sites.
2. *Privacy:* An issue related to security is privacy of the user's personal information. In eCommerce websites users are often reluctant to provide their personal information and the websites that asks users too much of personal information or that do not handle users personal information properly are not trustworthy. A 1998 survey by privacy expert Alan Westin found that 81% of Internet users worry about the online invasion of privacy (Harris, Westin, 1998). The websites should define their clear and concise privacy policies so that users could become aware of the distribution of

their personal information to the third parties. There are many trusted third parties like BBBOnline (Better Business Bureau), TRUSTe, VeriSign etc. that are providing the trust seals verifying the security and privacy of the eCommerce websites.

3. *Insurance:* Insurance refers to a social arrangement in which there is a promise to compensate individuals for future harm if it occurs. In e-Commerce, insurance is often offered in terms of financial compensation (for example: fully covering the cost of a credit card purchase that goes awry) or some other arrangement (such as seeking to recover data destroyed by mistake).

4. *Fault Tolerance:* Users and especially customers trust an eCommerce website that has fault tolerance capability for example: if the user enters false value on some field in a transaction form, the website should display an appropriate error message to the user conveying the fault.

5. *Familiarity:* As the name implies it is the name that acts for the fame of the eCommerce sites. Cheskin's report on eCommerce trust study [21] defines it as "brand" that is an important attribute behind the trustworthiness of eCommerce sites. If the customer is familiar to the website or the technology aspects of the website, he/she trust it easily.

6. *Navigability:* The eCommerce trust study report [21] describes navigability as an important component in the establishment of trustworthiness. According to this report if the visitors cannot seek what they need they don't trust that site.

7. *Presentation:* The eCommerce trust study [21] also defines presentation of information on the eCommerce site another needed component in the establishment of trustworthiness. According to this report users see the design attributes that covey them that the site has a professional look and feel and therefore they can trust it.

8. *Self-Descriptiveness:* If a website does not explain its purpose to the users, they will not trust it because they don't know it. The customers may want to know the physical location, contact numbers etc of the company operating an eCommerce site before spending any money online.

9. *Controllability:* The users want some control of their actions while transacting on eCommerce websites. For example they may want to edit or retrace back during the transaction or they may want to cancel it.

11. *Operability:* An eCommerce site should have good operability in terms of the functionality it is offering that will help in the successful consumer transactions leading to trustfulness.

These criteria lead to the mapping shown in Figure 23.

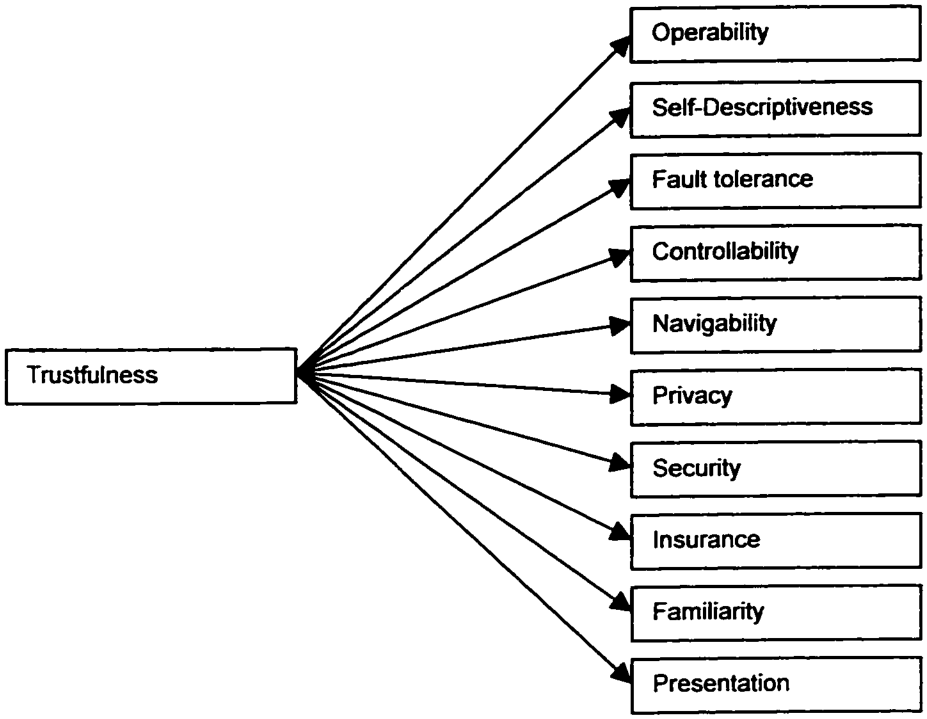


Figure 23: Mapping trustfulness to related criteria

5.9 Accessibility

Now a day more and more people are becoming aware and thinking the accessibility at all levels of society. Earlier its scope was limited to the accessibility of buildings and traditional media. However, this is now being extended to include the accessibility of online information and services made available through web sites. (Gulliksen et al., 2001) say that Accessibility and universal access have been gaining increasing attention recently due to the increasing recognition of the need to promote equal opportunities for all users of interactive systems. Our present research is based on the website accessibility only. These days most of the websites are aimed at impressing and engaging the mainstream surfers only, and they confuse and alienate persons with visual limitations.

Definitions

'The usability of a product, service, environment or facility by people with the widest range of capabilities.' (ISO TS 16071,2000)

According to ISO's definition of accessibility, it is strongly related to the concept of usability as defined in (ISO/IEC 9241-11,1998). Because of its relation to the definition of usability, accessibility becomes a measurable entity; and therefore developers could acquire the goal of increasing the level of accessibility of the products they develop rather than assessing if a product is accessible or not. (Gulliksen et al., 2001)

Accessibility means providing flexibility to accommodate each user's needs and preferences. In an Internet context, accessibility is making computer technology and Internet resources useful to more people than would otherwise be the case. (Valdes, 1998)

Software accessibility can be defined as a trait of software or other electronic information sources whereby it is usable by people with physical, cognitive or emotional disabilities. A software, website or other electronic information source is accessible if someone with a disability is able to use the source's data, information, or services as effectively as someone without a disability. (Slatin, 2001)

Our definition of accessibility is:

'The capability of a software product to be used by permanently or temporarily disabled persons (i.e. vision, hearing, motor, cognitive and language impairment) in a certain context of use.'

5.9.1 Accessibility versus Usability

Although accessibility and usability are related, accessibility comes before usability. Accessibility is by definition a category of usability: software that is not accessible to a particular user is not usable by that person. For a software product to be usable, the user must be able to access its functionality. If the user is able to access the functionality then it is the issue of usability i.e. whether the function can actually be used; otherwise it is the accessibility issue. So, accessibility must be addressed before usability. Accessibility is for a special user group and usability is to consider all users.

5.9.2 Driving forces behind web accessibility

Leo Valdes (2001) has stated that there are many reasons to strive for web accessibility namely:

1. **Commercial:** Website Accessibility helps in increasing the business of organizations as it allows a larger participating audience. Accessible web pages expand a site's potential audience to the millions of disabled persons.
2. **Legal:** It is the legal requirement in some countries like United States, where, the Americans with Disabilities Act requires reasonable accommodation for users with disabilities.
3. **Better Design For All Users:** Accessible designs are often helpful not only to the disabled users but also the mainstream users. For example, screen readers and dictation software, which are meant to empower the visually impaired, can be used for document creation and proofing.

4. **Equity:** Accessibility promotes the equality right of all users whether they are the normal or the persons with disabilities. It helps in achieving societal goals with full and equal participation.

5.9.3 Disability types

Disability can be permanent or temporary depending on the age and condition. The disabled users class is further composed into different types based on the disabling condition as follows:

1. **Blind and low vision users:** A blind person is one who cannot use a visual display at all. These are the users who read Braille displays or listen to speech output (from a screen reader) to get information from their systems. Users with low vision have a wide variety of visual capabilities. A person with low vision can be considered to be someone who can only read print that is very large, magnified, or held very close. Low vision can also be thought when the light is not proper and it is more the context of use.

2. **Colorblind users:** Color vision confusion or color blindness as it is usually known is a common impairment, affecting approximately 8% of males and 2% of females. These individuals see a different range of colors i.e. they confuse colors. Color blindness can be in many variations and degrees of severity. Color blindness is a problem in seeing colors as most others see them. But it is not blindness; it has nothing to do with the eyesight. These people confuse some colors, and may not see some ones at all.

3. **Cognitively disabled users:** Cognitively disabled users are the users who have Memory, Perception, Problem-solving, and conceptualizing disabilities. A cognitively disabled user may have difficulty with finding solutions to problems, lack of expression power, understanding and using the language i.e. he/she may have below average general I.Q. (intellectual quotient). Cognitively disabled users don't have the swift reflexes like the normal users and they have difficulty in understanding the demands and expectations of the environment.

4. **Deaf and hearing impaired users:** It include the category of those users who cannot hear and those users who do not hear because they are talking to others on the phones, paying attention to their tasks, working in a noisy environment.

5. **Mobility-impaired users:** Deng (2001) classifies mobility-impaired users as the persons who are unable to move, manipulate objects and interact with the physical world.

5.9.4 Mapping criteria to accessibility

World Wide Web consortium (W3C) on Web Accessibility Initiative (WAI) has defined several guidelines to incorporate web accessibility. The criteria that are supposed to be responsible for accessibility of websites are recognized through the needs of disabled users defined in these guidelines.

1. **Blind and low vision users:** To measure the user interface issues that specifically relate to blind and low vision users, we have to think about their requirements. Blind users need the alternative text representation for

all non-text elements like graphics, images, icons etc. They also needed the good keyboard navigability because mouse is not useful to blind users as it need hand and eye coordination. Blind users need efficient consistency of the user interface because they use the keyboard access mechanism only. Blind users must rely on their own memory to understand the contents of the screen through screen reading software. Low vision users also need the customizable user interfaces, so that they can transform their requirements into the defaults provided. Readability of the web content can be especially problematic for visually impaired users. Like the blind users low vision users also need the minimal memory load to understand the information contents. Low vision users are also interested in good navigability and clear and concise presentation of the information.

The criteria that specifically address issues related to blind and low vision users are therefore Flexibility, Consistency, Navigability, Minimize Memory Load, Readability and Presentation.

2. **Colorblind users:** The most important issue in measuring the user interface for colorblind users is that user interface should not rely on color alone to convey information. Instead, it should provide redundant means of conveying information. Thus, for colorblind users Flexibility is most important because it helps in providing user customizable font styles and sizes; and foreground, background colors.

3. **Cognitively disabled users:** Cognitively disabled users need plain and

short description of the site information. They should have less memory load on them to understand the site's content. The site should be structured in such a way so that cognitively disabled users could easily achieve their actions. For Cognitively disabled users Self-Description, Minimize Memory Load, Readability, Presentation, Understandability, Operability and Minimal Action are the basic requirements of the user interface.

4. **Deaf & hearing impaired:** Deaf and hearing-impaired users need alternative means for audio presentation. The criterion related with this type of disability is mainly Flexibility.

5. **Mobility impaired users:** Studies have shown that Minimal Action and Navigability are the major issues for the mobility-impaired users.

The diagrammatic representation of the mapping is shown in Figure 24.

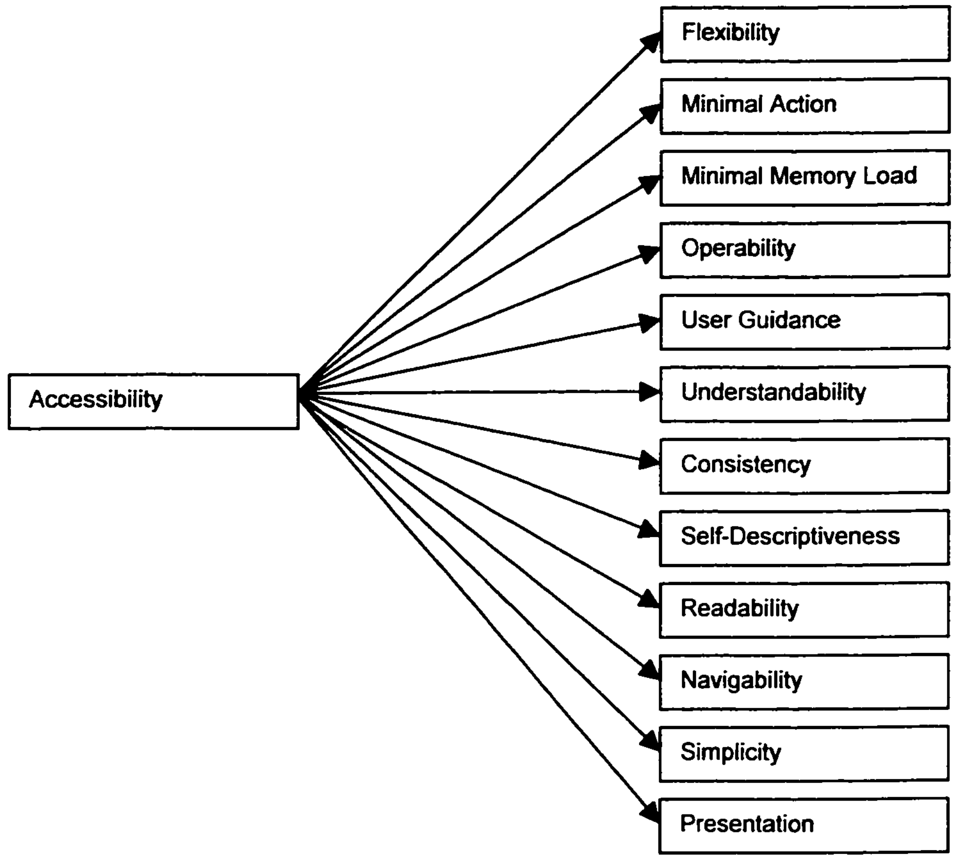


Figure 24: Mapping accessibility

5.9.5 Accessibility measurement

(Ambuhler, Lindenmeyer, 1999) have developed a metric to measure the web page accessibility and according to them an accessibility value of web pages in percent is calculated depending on how much of web pages could actually be used with each browser ("Content Accessibility") and depending on each browser's capability.

They have proposed the following formula:

$$a = \begin{cases} \dfrac{\left[\sum_{i=1}^{6} c \cdot f_B(i)\right] \cdot f_C \, f_i}{16} & s \leq 640 \\[2ex] \dfrac{\left[\sum_{i=1}^{6} c \cdot f_B(i)\right] \cdot f_C \, f_i}{0.0046s+13.037} & s > 640 \end{cases}$$

Where,

a = accessibility value in percent

c = content accessibility, which is assigned as follows:

- Fully accessible = 100%,
- Still useful = 67%,
- Important information is missing = 33%,
- Not accessible = 0%

f_C = "fixed color scheme"-factor: No: [1],Yes:[0.9]

f_i = "text as images"-factor: No: [1],Yes:[0.8]

f_B = browser-factor, given in the Table 3.

Table 3: Browser factor assignment

i	BROWSER	$F_B(i)$
1	MS Internet Explorer V3.x or higher	1
2	Netscape Communicator V3.x or higher	1
3	Spyglass enhanced Mosaic V2.1	2
4	MS Pocket Internet Explorer	3
5	Lynx V 2.8	4
6	PwWebSpeak V2.0	5

s = minimal screen width of the web page in pixels.

Heuristically 640 pixels (VGA) have been chosen as an acceptable screen width without loss of accessibility. The formula leads to decrease of accessibility by 10% if a minimal screen width of 1024 pixels is required.

They have evaluated practically the accessibility values of various websites and have demonstrated their results as:

- Yahoo.com has excellent accessibility with 100% a- value.
- W3C.com has good accessibility with 90% a-value.
- Microsoft.com has a-value of 72% and its accessibility status is OK.

5.10 Universality

'Thomas Jefferson in Reply to American Philosophical Society states that "I feel...an ardent desire to see knowledge so disseminated through the mass of mankind that it may...reach even the extremes of society: beggars and kings." (Shneiderman, 2000, p. 84)

Universality or Universal Usability is a challenging new factor in the field of information technology and it has gained a huge attention because of its value to the whole world. For the present study, we are considering the universality issues of websites only. Universality or Universal Usability factor is even a harder process because of the vast nature of the concept. Sill, we tried to break the concept but more has to be done to fully accomplish the measurement of the Universality (Universal Usability) of software products. Ben Shneiderman, who coined the concept, identified three main challenges to achieve Universal Usability - User Diversity, Technology Variety and Gaps in User Knowledge. We tried to seek the factors responsible for these challenges and gave them the shape of criteria and metrics.

Definitions

> ACM Code of Ethics states that 'In a fair society, all individuals would have equal opportunity to participate in, or benefit from, the use of computer resources regardless of race, sex, religion, age, disability, national origin or other such similar factors.' (Shneiderman, 2000, p. 85)
>
> Universal Usability will be met when affordable, useful, and usable technology accommodates the vast majority of the global population: this entails addressing challenges of technology variety, user diversity, and gaps in user knowledge in ways only beginning to be acknowledged by educational, corporate, and government agencies. (Shneiderman, 2000)

'A focus on designing products so that they can be used by the widest range of people operating in the widest range of situations as is commercially practical.' (Vanderheiden, 2000)

According to Vanderheiden (2000), universal usability is more a function of keeping all the people and all of the situations in mind and trying to create a product, which is as flexible as commercially practical, so that it can accommodate the different users and situations.

Universal usability involves understanding how users attempt to accomplish tasks in a varying context of use.

Our definition of universality is:

'The degree to which a software product accommodates diversity of user cultural/social archetypes in various context of use.'

Here, we want to point out that we have not included the technological aspects as given in other definitions of universality. Moreover our definition defines universality as an aspect that deals more with the cultural/ social archetypes of all users whether they are disabled users or with special needs. The term "context of use" in the definition itself is enough to explain the variety of technology used by the diversity of users (mainstream or the disabled users).

5.10.1 Main challenges in achieving universality

Shneiderman (2000) listed the three main hurdles in the path of Universal Usability:

1. *User diversity*: Accommodating users with different skills, knowledge, age, gender, disabilities, disabling conditions (mobility, sunlight, noise), literacy, culture, income, etc.
2. *Technology variety*: Supporting a broad range of hardware, software, and

network access.

3. *Gaps in user knowledge*: Bridging the gap between what users know and what they need to know.

User diversity - As the name applies, this issue has to deal with the variety of users that can access Internet. This class is further categorized into users with age differences like children and elderly, users with low education and motivation, users using other languages than English, users from other cultures than U.S.etc. It includes the issues of Globalization and Localization. A brief explanation of each of the users group is as below:

1. *Children:* In the era of information technology, children are the part of Internet users. Parents and teachers consider Internet to be a primarily educational / developmental tool for their kids. Children impose their own needs of the site's interface based on their age level of learning. To fulfill the demands of Universal Usability, children are the part of the class of users that has to be taken into account.

2. *Elderly:* Czaja (1997) has studied that attitudes of the elderly toward computers is no different than younger adults, but elderly people face age related difficulties in accessing World Wide Web technology. To achieve universal usability for the elderly, we need to understand what factors have to be considered in the software systems to cope with the age-related changes. Some of the usability problems suffered by the elderly are also the part of the problems encountered by the disabled users.

3. *Users with low education and low motivation:* Universal Usability concept

is for global population and it requires involvement of all users despite of their lack of education or motivation. These users include the users with low income. Illiterate and non-motivated users are the hurdles in the fulfillment of Universal Usability. Low motivated users are those who lack the spirit to access the Internet and it include the educated users as well who are from different professional backgrounds or those who have lost touch with World Wide Web technology.

4. *Users of other languages than English:* Internationalizing the World Wide Web is one of the goals of the Universal Usability. English is not the first language of all the users. To realize the vision of Universal Usability, one important requirement is to enable all languages to be technically available via the Internet, so that users using other languages than English could also be accommodated.

5. *Users from other cultures than U.S.:* Culture is the sum total of learned behavior and standard beliefs of a group of people that are generally considered to be the tradition of that group and which helps in distinguishing it from another human group. The cross cultural study revealed that to expect Universality one has to understand the impact of culture on the understanding and use of the sites. Users from different cultures have different needs, wants, preferences, and expectations.

Technology variety - To achieve universality, we have to deal with the pace the technology changes and the variety of equipment employed by the users. Technology variety encompasses a broad range of hardware devices (like processors, RAM, hard disks, monitors, and input devices etc.), software changes (like different versions of a software) and network access technologies (like modems, cables etc.). In the competitive market there is a continuous change in technology and to cope with the problem of accommodating variety of hardware resources, software changes and network access methods is itself a big problem that has to be addressed. For the present study, this problem is divided into the following categories:

1. *Network access methods*: People access the World Wide Web through different access types like dial up cable modems, DSL, Internet TV etc. The speed of network connection varies for these access methods and this is an important factor to get the full benefits of universality.

2. *Screen resolutions:* Now a day a variety of devices are available to access the Internet, which includes PDAs, mobile phones, hand-held PCs, Web TV etc. The screen resolution of theses devices vary as WebTV is 544 x 372; hand-held PCs is around 240 x 320; popular palm-sized PDA is about 160 x 160 and mobile phones can be as low as 48 x 48 pixels. Universal Usability issue dealing with these types of devices is how to deal with small screen resolutions.

Gaps in user knowledge - The third challenge to Universal Usability is to bridge the gap between what users know and what they need to know. Users have diverse skills and intelligence levels. Users are basically classified as novice and expert users. Expert users can easily use new tools after a few minutes orientation of understanding the novelties while novice users need more time to acquire knowledge about the objects and actions in the application domain and learn the functionality of the user interface.

5.10.2 Mapping criteria to universality

It is said earlier that the criteria and metrics selected are based on the principal difficulties involved in the fulfillment of Universal Usability. So the criteria depicted here are in a way substituting these challenges to Universal Usability.

The very first challenge to Universal Usability is fulfilling the needs of a wide range of users having different ages, cultures and social environments. Here is the brief explanation of the user interface needs required by different users that help in making a website universally acceptable.

1. *Children:* Children should be protected from the private information content on the site. Children need simplistic user interface designs and they are more interested in those sites that have customizable user interface and have appealing look and feel. Privacy, Flexibility, Readability, Attractiveness and Simplicity are the issues that have to be addressed to measure the quality of user interface in case of children.
2. *Elderly:* Elderly persons may suffer from the same problems like other disabled persons. Moreover elderly persons need the user control over the functionality of the interface. Consistency, Navigability, Presentation, Controllability, User Guidance are the criteria to understand for elderly users.
3. *Users with low education and low motivation:* Understandability, Self Description, Navigability, Minimize Memory Load, Consistency and

Flexibility are the related criteria for these kinds of users.

4. *Users from other culture than U.S.:* For these kinds of users, we should measure the user interface in terms of Simplicity, Presentation criteria.

5. *Users using first language other than English:* The measurement of Cross Language Information Retrieval (CLIR) systems is done basically on the availability of the flexibility of the interface they provide. So the main criterion under consideration is Flexibility.

The second challenge to Universal Usability is Technology Variety, which is further addressed through Network connection speed and Screen width.

1. *Network connection speed:* It directly affects the Page Loading time of websites.

2. *Screen width:* Readability, Simplicity, Presentation, Navigability are the criteria needed to test the screen variations in different devices.

Gaps in user knowledge: User knowledge gaps are measured by using the measures of Consistency, Understandability, Self-Description, Flexibility, Minimal Action and Feedback.

Therefore, the factor universality can be measured through the criteria shown in Figure 25.

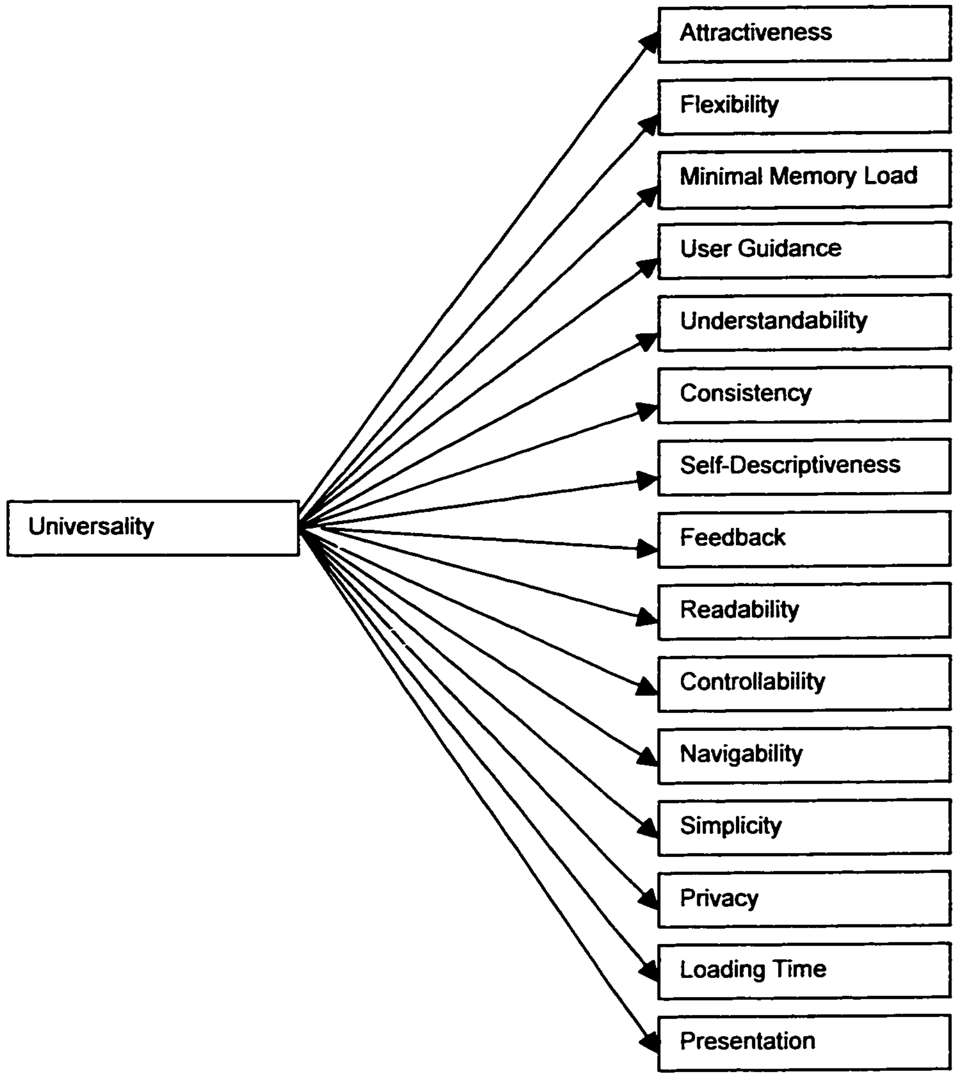


Figure 25: Criteria affecting universality

5.11 Usefulness

A software product may be easy to use but may not be relevant to the actual needs of a user. Therefore, it is essential to measure its practical usefulness. Usefulness depends to a large degree on the features and functionality offered by the software product. The rationale behind choosing usefulness as one of the factor under study is that it is a concept that is often seen inside the usability perspective. Human Computer Interaction professionals are now recognizing its own value and are now a day thinking of it as an independent quality attribute of software product. Our aim is to show its own essence in terms of the quality of the software product and to define it in terms of metrics.

Definitions

Nielsen (1993) suggests that usability and utility together form the usefulness of a system, which is an important attribute affecting product acceptability (Figure 26). 'Nielsen makes this explicit: "...utility is the question of whether the functionality of the system in principle can do what is needed, and usability is the question of how well users can use that functionality."'(Nielsen 1993)

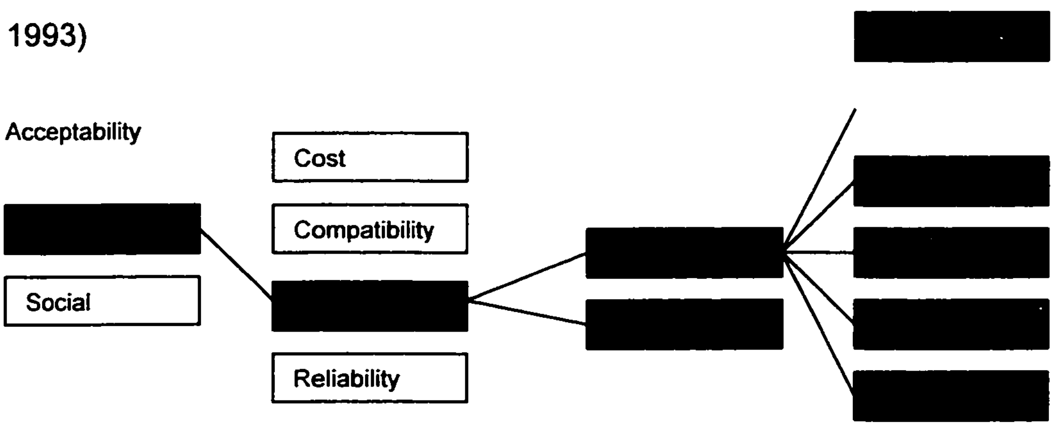


Figure 26: Product acceptance by J. Nielsen (1993)

Here, utility is the ability of the functions to help the user carry out a set of tasks. Usability is a concept that focuses on the problems of how users utilize these functions.

Davis (1993) defines usefulness as: 'the degree to which an individual believes that using a particular system would enhance his or her job performance.' In this definition, 'Performance' refers to the output of the activity and the productive aspects of the process. The meaning is the same as in Nielsen's (1993) conceptual hierarchy, i.e. it joins utility and aspects of usability.

'Usefulness concerns the degree to which a product enables a user to achieve his or her goals, and is an assessment of the user's motivation for using the product at all.' (Rubin, 1994)

Our definition of usefulness is:

> The degree to which a software product actually helps to solve users real practical problems in a certain context of use.

Usefulness versus Usability

It is important to distinguish between these two seemingly similar concepts.

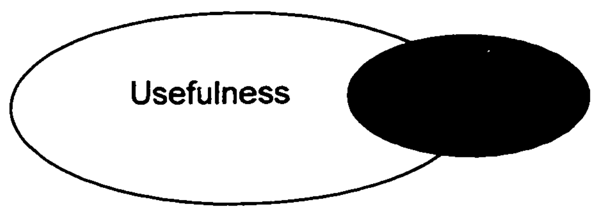


Figure 27: usefulness versus usability

Usefulness implies that the software product has practical utility, which is a measure of how directly a software product supports user's own task model. On the other hand, usability is the ease-of-use of the software product, which implies that how easy it is to figure out what actions are needed while performing some task. A product can be usable but not useful, whereas a useful product is not necessarily usable. Usefulness is a mapping between what the user wants to do and what he/she must do within the constraints imposed by the interface. Wobbrock (2000) states that no matter how usable the product is engineered to be, extreme usability can never result in inherently greater usefulness (Rather poor usability can rather hamper usefulness), this is because high usability cannot create a need where there is none, nor can change the fundamental solution the product is attempting to provide. He also points out that useful product is meaningless if it is not fulfilling its intended context of use

Bohmann (2000) has identified four types of correlations between usefulness and usability:

- High usefulness, highly usable: High usefulness and highly usable software products imply that the optimal user performance will increase in terms of low task time and low task error.
- High usefulness, low usability: It means that users will be able to perform tasks, but with more task errors and task time.
- Low usefulness, highly usable: Low usefulness means that users can perform a limited set of tasks in easy ways. However, users may need to do more work if they do their tasks manually or using other software products.
- Low usefulness, low usability: This scenario states that users are expected to produce many task errors and take long time to perform a task.

Criteria

The criteria that can be used to measure usefulness of a software product are shown in Figure 28.

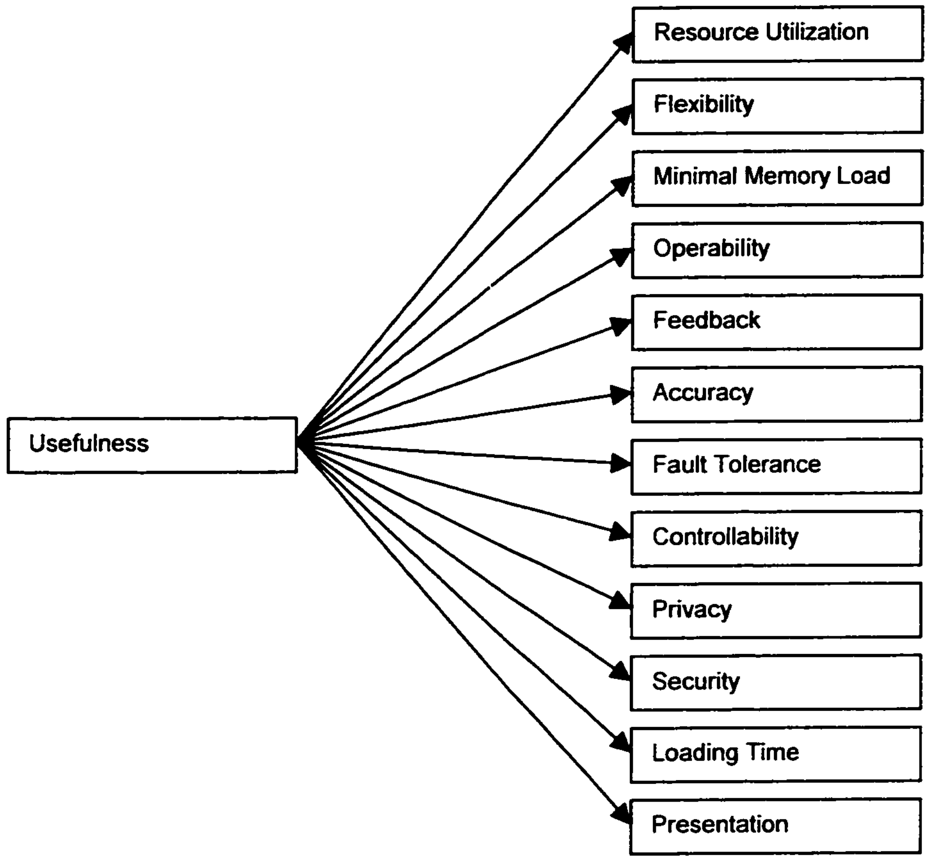


Figure 28: Mapping usefulness to various criteria

5.12 Factors/Criteria relationships matrix

Figure 29 shows the relationships that exist between 27 criteria and 10 factors we identified. This picture also shows the relative importance of the criteria and factors. From the figure, universality is the most important factor in terms of the number of criteria that can be used to measure it. On the other hand, minimal memory load is the criterion that is affecting a number of factors.

FACTORS / CRITERIAS	Efficiency	Effectiveness	Satisfaction	Productivity	Learnability	Safety	Trustfulness	Accessibility	Universality	Usefulness	
Time Behavior	•			•							2/10
Resource Utilization	•			•						•	3/10
Attractiveness			•						•		2/10
Likeability			•								1/10
Flexibility		•	•					•	•	•	5/10
Minimal Action	•		•		•			•			4/10
Minimal Memory Load	•		•		•			•	•	•	6/10
Operability	•		•				•	•		•	5/10
User Guidance			•		•			•	•		4/10
Understandability	•		•		•			•	•		5/10
Consistency		•			•	•		•	•		5/10
Self-Descriptiveness					•		•	•	•		4/10
Feedback	•	•							•	•	4/10
Accuracy		•				•				•	3/10
Completeness		•				•					2/10
Fault- Tolerance						•	•			•	3/10
Resource Safety						•					1/10
Readability								•	•		2/10
Controllability							•	•	•	•	4/10
Navigability	•	•					•	•	•		5/10
Simplicity					•			•	•		3/10
Privacy							•		•	•	3/10
Security						•	•			•	3/10
Insurance						•	•				2/10
Familiarity					•		•				2/10
Loading Time	•			•					•	•	4/10
Presentation							•	•	•	•	4/10
	9/27	6/27	8/27	3/27	8/27	7/27	10/27	13/27	15/27	12/27	

Figure 29: Factors / Criteria relationships matrix

Chapter 6

Conclusion and Future Investigations

Building a useful and consistent repository of factors, criteria and metrics for measuring quality in use, was not an easy work. We encountered several problems while surveying the literature that can be summarized as follows:

- Some of the factors we identified such as universality and trustfulness are new. Only little work has been done yet to define and describe these factors precisely. Moreover, they have different meanings and they seem to be a combination of many other existing factors. Also, finding relevant information about them was even more problematic.
- It was very hard to integrate the different perceptions. This is because the decomposition of factors to criteria varies depending upon various judgments and interpretations.
- The most problematic was to seek useful interpretation for the metrics so that baseline measures could be derived from them. The information was scattered throughout the World Wide Web and it was hard to collect and organize the relevant information to interpret the metrics values.

This research is a first step of a long-term project that aims to build and validate measurement knowledge map. In this thesis, we collected 10 factors, 27 sub factors or criteria and 128 metrics to measure those criteria. This collection of factors, criteria and metrics will provide some of the characteristics of quality in use of the software products. Our repository is dynamic in the sense that it can adapt itself to the requirements of any organization. This repository is equally beneficial to all. We have tried to solve the conflicts in usability definitions as any person can think of usability in terms of these factors. For example a person "A" says that a usable software product is one that is efficient, productive in its operation and also implies user satisfaction and learnability; person "B" can have different view of usability of a software product defined in terms of Trustfulness, Safety, Usefulness. So we have provided a framework where anybody can define and measure his/her own view of usability.

Our future plan encompasses the following activities:

1. Validation: The first and foremost step in near future will be to validate the repository of factors, criteria and metrics. We are thinking to validate it via online surveys, user testing questionnaires so that we could gain statistical information about our repository.
2. Factor exploration: The next step will be to analyze more deeply the list of factors. At present, we have studied some factors only partially like trustfulness (which was investigated for eCommerce sites only), Universality. We want to explore more of them, so that we could see their impacts on other software products.
3. Extension: The next step can be to extend the list of factors. We will try to answer if our list is exhaustive or there are other factors that have to be considered. A potential candidate factor is "comprehension". This phase

will explore more quality in use characteristics of any software product so as to make our repository an integrative and universally accepted standard.

References

Papers and Journals

1. Ahuja, V. (2000). *Building Trust in Electronic Commerce* [Internet]. Available from <http://dlib2.computer.org/it/books/it2000/pdf/f3061.pdf? > [Accessed December 5th, 2002].

2. Ambuhler, R. and Lindenmeyer, J. (1999). Measuring Accessibility. *Poster from 8th International WWW Conference Proceedings*. pp. 48-49.

3. Babiker, E. M. Fujihara, H. Boyle, C. D. B. (1991). A Metric for Hypertext Usability. *ACM Proceedings of the 9th annual international conference on system documenattaion* [Internet journal]. pp. 95-104. Chicago, Illinois, US. Available from <http://portal.acm.org/citation.cfm?id=122793&coll=portal&dl=ACM&ret=1#Fulltext> [Accessed January 3rd, 2003]

4. Bernard M. and Mills M. (2000). So, What Size and Type of Font Should I Use on My Website? In Software Usability Research Laboratory, *Research into readability, and preference in fonts for online reading* [Internet]. Witchita State University. Available from < http://wsupsy.psy.twsu.edu/surl/usabilitynews/2S/font.htm> [Accessed January 10th, 2003].

5. Bevan, N. and Macleod M. (1994). Usability measurement in context. *Behaviour and Information Technology no.*13, part nos. 1 and 2.

6. Bevan, N. (1995). Measuring usability as quality of use. *Software Quality Journal*. Issue 4. pp. 115-140.

7. Bevan, N. (1997). Quality and usability: A new framework. In van V. E. and McMullan, J. (eds) *Achieving software product quality*. Tutein Nolthenius, Netherlands.

8. Bevan, N. (1999). Quality In Use: Meeting User Needs for Quality. *Journal of System and Software. Serco Usability Services* [Internet]. Available from <http://www.usability.serco.com/papers/qiuse.pdf> [Accessed November 11th, 2002].

9. Boehm, B. W. Brown, J. R. Kaspar, Hans. Lipow, Myron. Macleod, G. J. and Merritt M. J. (1978). *Characteristics of Software Quality*. New York. North-Holland Publishing Company.

10. Bohmann, K. (2000). *Useful Features are Not Always Usable* [Internet]. Available from <http://www.bohmann.dk/articles/useful_features_are_not_always_usable.html> [Accessed January 28th, 2003].

11. Brooke, J. [1996]. SUS- A "quick and dirty" usability scale [Internet]. Available from <http://www.cee.hw.ac.uk/~ph/sus.html > [Accessed December 29th, 2003].

12. Carson, R. (1999). *Generalization In User Interface Design*. Draft# 2. Available from <http://web.media.mit.edu/~carsonr/pdf/generalization.pdf> [Accessed December 24th, 2002].

13. Common Industry Format for Usability Test Reports (2001). Version 2.0. In *Industry Usability Reporting project* [Internet]. Available from <http://hci.stanford.edu/cs377/nardi-schiano/AW.CIFUTR.pdf> [Accessed November 12th, 2002].

14. Constantine, L. L. Lockwood, L. A. D. (1999). *Software for Use: A Practical Guide to the Models and Methods of Usage-Centred Design.* New York, Addison-Wesley.

15. Czaja. Sara, J. (1997). *Handbook of Human-Computer Interaction.* Second edition. Elsevier Publications.

16. Davis, F. D. (1993). User acceptance of information technology: system characteristics, user performance and behavioral impacts. *International Journal of Man-Machine Studies* no. 38. pp. 475-487.

17. Deng, Y. (2001). Accommodating mobility impaired users on Web [Internet]. Available from <http://www.otal.umd.edu/uupractice/mobility/> [Accessed November 15th, 2002].

18. DIN 55350-11(1995). Concepts in quality and statistics; basic concepts in quality management, quality control and quality assurance.

19. Dix. Finlay. Abowd. Beale (1993). *Human Computer Interaction* [Internet]. Available from <http://www.dcs.napier.ac.uk/marble/Usability/UsabilityMetrics.html> [Accessed Oct 23rd, 2002].

20. Donyaee M. (2001). Towards an Integrated Model for Specifying and Measuring Quality in Use. In the Webster library, Concordia University. Montreal, Canada.

21. eCommerce Trust study (1999). In *Cheskin Research and Studio Archetype/ Sapient's* [Internet]. Available from <http://www.cheskin.com/think/studies/eCommtrust99.pdf> [Accessed January 5th, 2003].

22. Fenton, N. E. Whitty, R. (1995). *Software Quality Assurance and Measurement: A Worldwide Perspective*. London. International Thomson Computer Press.

23. Fenton, N. E. Pfleeger, S. L. (1997). *Software Metrics: A Regorous and Practical Approach.*International Thomson Publishing Company.

24. Friedman, B. Kahn, P.H. Jr. and Howe, D.C. (2000). Trust Online. *ACM Communications* no. 43, Issue 12. pp. 34-40.

25. Gentner, D. Nielsen, J. (1996). The Anti-Mac Interface. *Communications of the ACM* no. 39, Issue 8 (August).

26. Glinert, E. P. (1990). *Nontextual Programming Environments, Principles of Visual Programming Systems*. Prentice Hall Inc. pp. 195.

27. Grandison, T. and Sloman, M. A Survey Of Trust In Internet Applications. In SURVEYS IEEE COMMUNICATIONS, *The Electronic Magazine of Original Peer-Reviewed Survey Articles.*

28. Grudin, J. (1989). The case against user interface consistency. *Communications of the ACM* no.32, Issue 10(October). pp. 1164-1173.

29. Gulliksen, J. Lutsch, C. and Harker S. (2001). Accessibility Through Standardisation. *1st International Conference on Universal Access in Human-Computer Interaction*. New Orleans, Louisiana, USA.

30. GVU7 (1997). *GVU's 7th WWW User Survey* [Internet]. Available from <http://www.gvu.gatech.edu/user_surveys/survey-1997-04/> [Accessed December 14th, 2002].

31. Harris, L. and Westin, A. (1998). E-commerce and privacy: What net users want. In *Privacy and American Business*. Hackensack, NJ.

32. Hayne, C. (1996). Software Engineering for Usability. In General Data Communications Multimedia R&D Centre.

33. Hix, D. Hartson, H.R. (1993). *Developing User Interfaces: Ensuring Usability Through Product and Process*. John Wiley and Sons.

34. IBM Privacy Survey (1991)[Internet]. Available from <http://www-1.ibm.com/services/files/privacy_survey_oct991.pdf> [Accessed December 14th, 2003].

35. IEEE Std 729 (1983). *IEEE Standard Glossary of Software Engineering Terminology*.

36. IEEE Std 1061(1998). *Standard for a Software Quality Metrics Methodology*.

37. IEEE (1990). *IEEE Standard Computer Dictionary: A Compilation of IEEE Standard Computer Glossaries*. New York.

38. Ince, D. (1994). *ISO 9001 and software quality assurance*. Berkshire, England, McGraw- Hill.

39. ISO 8402 (1986). *Quality – Vocabulary. International Organization for Standardization*. Geneva.

40. ISO/IEC 9126(1991). *Information Technology – Software quality characteristics and metrics.*

41. ISO/IEC 12207(1995). *Information Technology - Software life cycle processes.*

42. ISO/IEC 9126-1(1998). *Information Technology – Software product quality – Part 1: Quality Model* [Internet]. Available from <http://www.cs.uiowa.edu/~sdkim/22c180/ISO-9126-1.pdf> [Accessed January 23rd, 2003].

43. ISO 9241-11(1998). *Ergonomic Requirements for Office Work with Visual Display Terminals – Part 11: Guidance on Usability*. International Organization for Standardization.

44. ISO/IEC 14598-1(1999). *Software Product Evaluation: General Overview.*

45. ISO TS 16071(2000). Ergonomics of human-system interaction – guidance on software accessibility. In *Technical Specification*. Switzerland: International Organization for Standardization.

46. ISO/IEC 9126-4(2000). *Software Engineering- Product Quality - Part 4: Quality In Use Metrics* [Internet]. Available from <http://www.sqi.gu.edu.au/sc7-mirror/private_files/07n2268.pdf>[Accessed January 23rd, 2003].

47. ISO/IEC 9126-2(2001). *Software Engineering- Product Quality- Part 2: External Metrics*. [Internet]. Available from

<http://www.cs.uiowa.edu/~sdkim/22c180/ISO-9126-2.pdf>[Accessed January 24th, 2003].

48. Kieras, D. E., Wood, S. D. Abotel, K. and Hornof A. (1995). GLEAN: A computer based tool for rapid GOMS model usability evaluation of user interface designs. *ACM Symposium on User Interface Software and Technology.*

49. Kini, A. Choobineh, J. (1998). Trust in Electronic Commerce: Definition and Theoretical Considerations. *31st Annual Hawaii International Conference on System Sciences* no. 4. pp.51-61.

50. Kirakowski, J. [1996]. Software Usability measurement Inventory [Internet]. Available from <http://www.ucc.ie/hfrg/questionnaires/sumi/> [Accessed December 29th, 2003].

51. Kirakowski, J. Barry, N. (1996). Usability assessment, version 1.2.In *INUSE project* [Internet]. Available from <http://info.lboro.ac.uk/research/husat/inuse/assess.doc> [Accessed January 5th, 2003].

52. Kirchner, P. (1959). Measurements and Management Decisions. In Churchman, C. West and Ratoosh, P., ed. *Measurement: Definitions and Theories.* New York, John Wiley & Sons Inc., 1959, pp. 64-82.

53. Lewis, J.R. (1991a). Psychometric evaluation of an after-scenario questionnaire for computer usability studies: the ASQ. *SIGCHI Bulletin* no. 23, 1 (January). pp. 78-81.

54. Lewis, J.R. (1991b). Questionnaire For User Interface Satisfaction [Internet]. Avaialble from <http://lap.umd.edu/QUIS/> [Accessed January 5th, 2003].

55. Lewis, J.R. (1995). IBM Computer usability satisfaction questionnaires: psychometric evaluation and instructions for use. *International Journal of Human-Computer Interaction* no.7. pp. 57-78.

56. Lin. Choong, H.X. Y, Y. and Salvendy, G. (1997) .A Proposed Index of Usability: A Method for Comparing the Relative Usability of Different Software Systems. In *Behaviour and Information Technology*. Volume no.16, Issue no. 4. pp. 267-278.

57. Luhmann, N. (1979). *Trust and Power*. London. John Wiley and Sons.

58. Macleod, M. Bowden, R. Bevan, N. and Curson, I. (1997). The MUSiC Performance Method. In *Behaviour and Information Technology* [Internet]. Available from <http://www.usability.serco.com/papers/muspmm97.pdf> [Accessed January 12th, 2003].

59. Macleod, M. Rengger, R .*The development of DRUM: A software tool for video-assisted usability evaluation* [Internet]. Available from <ftp://ftp.npl.co.uk/pub/hci/papers/DRUM.rtf> [Accessed January 10th, 2003].

60. Magers, C.S.(1983). An experimental evaluation of on-line HELP for non-programmers. In *Proceedings of CHI'83 Human Factors in Computing Systems* . New York, Association for Computing Machinery. pp. 277-281.

61. Mahajan, R. Schneiderman, B. (1997). Visual and Textual Consistency Checking Tools for Graphical User Interfaces. *IEEE Transactions on Software Engineering*.pp. 772-735.

62. Max, and the Objective Measurement of Web Sites. In *a white paper on WebCriteria profile* [Internet]. Available from <http://advance.aiga.org/timeline/artifacts/WebCriteria.pdf>[Accessed December 5th, 2002].

63. Mayhew, D.J. (1992). *Principles and Guidelines in Software User Interface Design*. Englewood Cliffs, NJ, Prentice Hall.

64. McCall. J.A. Richards, P.K. and Walters, G.F. (1977). Factors in Software Quality. In Rome Air Force Development Center, *RADC TR-77-363.* Griffis Air Force. Rome, New York.

65. McClure, C. R. Wyman, S. K. Beachboard, J. B. and. Eschenfelder, K. R. (1997). Developing System-based and User-based Criteria for Assessing Federal Websites. *Journal of the American Society for Information Science* (34). pp. 78-88.

66. McGovern G. (2000). *Navigating a Web site* [Internet]. Available from <http://www.clickz.com/design/site_design/article.php/829491> [Accessed January 2nd, 2003].

67. Merriam Webster dictionary (2001)[Internet]. Available from <http://www.m-w.com/home.htm> [Accessed December 9th, 2002].

68. Mills, E.E. (1993). Software Metrics. In *SEI Curriculum Model SEI-CM-12-1.1.* Carnegie Mellon University, Software Engineering Institute.

69. Ngo, D. C. L. and Teo, L. S. *A Mathematical Theory of Interface Aesthetics* [Internet]. Available from <http://members.tripod.com/vismath4/ngo/ > [Accessed Oct 10th, 2002].

70. Nielsen, J. (1993). *Usability engineering*. Boston. Academic Press Inc.

71. Nielsen J. (1997). Top ten mistakes in web design. In *Alertbox Column* [Internet]. Available from <http://www.useit.com/alertbox/9706b.html>[Accessed January 10th, 2003].

72. Nielsen J. (1999). *Designing Web Usability: The Practice of Simplicity*. Indiana. New Riders Publishing.

73. NIST's WebSAT Evaluation Rules for Readability [Internet]. Available from <http://zing.ncsl.nist.gov/WebTools/WebSAT/readability.shtml>[Accessed November 15th, 2002].

74. Olsina, L. Godoy, D. Lafuente, G. and Rossi, G. (1999). Specifying Quality Characteristics and Attributes for Websites. *ICSE 99 Web Engineering Workshop* [Internet]. Los Angeles, US. Available from <http://citeseer.nj.nec.com/olsina16specifying.html> [Accessed January 29th, 2003].

75. Oxford English dictionary [Internet]. Second edition. Available from <http://dictionary.oed.com/> [Accessed December 19th, 2002].

76. Preece, J. Rogers, Y. Sharp, H. Benyon, D. Holland, S. Carey, T. (1994). *Human - Computer Interaction*. Wokingham, UK, Addison Wesley Publishing Company.

77. Readability scores [Internet]. Available from <http://www.comp.glam.ac.uk/Blackboard/ISM20/Readability scores.doc> [Accessed December 23rd, 2002].

78. Rose, G. M. (1998). *Understanding the Impacts of Download Time on Electronic Commerce: The Download Time Brand Impact Model.* P.H.D. Dissertation Proposal. Georgia State University.

79. Rubin, J. (1994). Handbook of Usability Testing. New York, John Wiley & Sons, Inc.

80. Schneidewind, N. F. (1992). Methodology For Validating Software Metrics. *IEEE transactions on Software Engineering* no. 8, Issue no 5.

81. Sears, A. (1993). Layout Appropriateness: A Metric for Evaluating User Interface Widget Layout. *IEEE Transaction on Software Engineering* no. 19 (July). pp. 707-719.

82. Sears, A. (1995). AIDE: A step toward metric-based interface development tools. *Proceedings of the 8th annual ACM symposium on User interface and software technology* [Internet journal]. pp. 101-110. Available from <http://portal.acm.org/citation.cfm?id=215704&dl=ACM&coll=portal> [Accessed April 7th, 2002].

83. Section 508 rules [Internet]. Available from <http://www.niehs.nih.gov/websmith/508/rule.htm> [Accessed January 2nd, 2003].

84. Seffah, A. Donyaee, M. Theoretical Framework for Quality in Use

Measurement.

85. Shackel, B. (1991). Usability – context, framework, design and evaluation. In *Human Factors for Informatics Usability* [Internet]. pp 21-38. Cambridge University Press.

86. Shneiderman B. (1992). *Designing the user interface: Strategies for Effective Human-Computer Interaction.* Second Edition. MA. Addison-Wesley Pub. Co.

87. Shneiderman B. (2000). Universal usability. *Communications of the ACM* no. 43, Issue 5 (May). pp. 84-91.

88. Slatin, J.M. (2001). Focus on participation: Toward a more accessible web [Internet]. Available from <http://www.gslis.utexas.edu/~l385t21/AU_WP_Cost_Justifying_Accessibility.pdf >[Accessed December 10th, 2002].

89. Spool, J. M. (1998). All the Right Stuff. In User Interface Engineering 5 *Eye for Design*. North Andover, MA. pp. 2-3.

90. Spool, J. M. (1999). *Web Site Usability: A Designers Guide*. Morgan Kaufmann Publishers.

91. Sterne J. (2001). Making Metrics Count: You know your Website is serving pages. But is it serving your company? In *Business 2.0* [Internet]. Available from <http://www.business2.com/articles/mag/0,1640,14594,FF.html[> [Accessed December 13th, 2002]

92. Stevens, S. S. (1959). Measurement, Psychophysics, and Utility. In Churchman, C. W. and Ratoosh, P., ed. *Measurement: Definitions and Theories*. New York, John Wiley & Sons Inc. pp.18 – 63.

93. Tiller, E. Green, P. (1999). Web navigation: How to Make Your Website Fast and Usable. *NIST's 5th Conference Proceedings on Human Factors & the Web* [Internet]. Available from <http://zing.ncsl.nist.gov/hfweb/proceedings/tiller-green/index.html> [Accessed December 11th, 2002].

94. Tilson, R. Dong, J. Martin, S. and Kieke, E. (1998). Factors and Principles Affecting the Usability of Four E-commerce Sites. *Conference proceedings from 4th Conference on Human Factors & the Web* [Internet]. NJ, USA. Available from <http://www.research.att.com/conf/hfweb/proceedings/tilson/index.html> [Accessed November 21st, 2002].

95. *The Collins English Dictionary* (1998). HarperCollins Publishers.

96. Tom, B., Darren, G. and Scott D. W. (2002). *Usability for the Web.* Morgan Kaufmann Publishers.

97. Tullis (1984). *Predicting the usability of alphanumeric displays.* PhD Dissertation. Dept of Psychology, Houston, TX, Rice University. pp. 46-61.

98. User Centered Design Principles. In Cognetics Corporation [Internet]. Available from <http://www.cognetics.com/services/ucd/design_principles.html> [Accessed August 12th, 2002].

99. Vanderdonckt, J. Gillo, X. (1994). Visual Techniques for Traditional and Multimedia Layouts. *ACM Proceedings of 2nd Workshop on Advanced Visual Interfaces. pp. 95-104.*

100. Vanderheiden G. (2000). Fundamental principles and priority setting for universal usability. *Proceedings of the ACM conference on Universal Usability.* pp.32-37.

101. Valdes, Leo. (1998). Accessibility on the Internet. Version 1.1. In *Report on a Priority Theme* [Internet]. Available from <http://www.un.org/esa/socdev/disacc00.htm> [Accessed December 18th, 2002].

102. Website Accessibility Checklist. In *University of Minnesota* [Internet]. Available from <http://www.crk.umn.edu/people/services/DisabilServ/WebsiteAccessibilityCheckList.htm> [Accessed January 13th, 2003].

103. Web Usability [Internet]. Available from <http://www.cs.mu.oz.au/371/web-usability.pdf> [Accessed December 11th, 2002].

104. Wilson, R. (1998). Seven Debilitating Diseases of Business Websites (and their cures). In *Web Marketing Today* [Internet]. Available from <http://www.wilsonweb.com/articles/7diseases.htm> [Accessed January 3rd, 2003].

105. Wixon, D. Wilson, C. (1997). The usability Engineering Framework for Product Design and Evaluation. In *Handbook of Human-Computer Interaction*. North-Holland, Elsevier. pp. 653-688.

106. Wobbrock, J. O. (2000). *Useful verses Usable* [Internet]. Available from <http://www.cs.cmu.edu/~jrock/soapbox/useful.html> [Accessed January 28th, 2003].

107. Wolf, R. (1989). Consistency as process, Coordinating User Interfaces for Consistency Checking. London, Academic Press Inc. pp. 89 -92.

108. Yamada, S. Hong, J.k. Sugita, S. (1995). Development and Evaluation of Hypermedia for Museum Education: Validation of Metrics. *ACM Transactions of Computer Human Interface* Volume no. I2, Issue no. 4.

Appendix

Metrics in QUIM

1. Metrics to measure Time Behavior

1.1 Task Time (Dix et al. et al., 1993)

It is a measure of the time taken to complete a task.

Interpretations

The lesser is this value is the better.

1.2 Time spent on errors (Dix et al., 1993)

Interpretations

The closer to 0 is the better.

1.3 Selection time (Author)

It is the time taken to click on a selection (because selecting a link requires a mental decision and a physical action).

Interpretations

Optimal value for this metric has yet to be determined.

2. Metrics to measure Resource Utilization

2.1 Human Efficiency (Bevan, Macleod, 1994)

It is a measure of the human efforts (either mental or physical) expended in relation to the effectiveness achieved and is given by:

Human Efficiency (HE) = Effectiveness/ Effort expended

Interpretations

The more HE is the better.

2.2 **Temporal Efficiency** (Bevan, Macleod, 1994) or **Task Productivity** (ISO/IEC 9126-4,2000)

It is a measure of the user time spent to achieve certain level of effectiveness or is the measure of how productive is the user, and is formulated as:

Temporal efficiency (TE)= Effectiveness/ Task Time

Interpretations

0<=TE

Lesser the time spent is the better.

(ISO 9126-4) points that if task completion has been measured, task productivity can be measured as task completion/task time.

2.3 **Economic Efficiency** (Bevan, Macleod, 1994) or **Economic Productivity** (ISO/IEC 9126-4,2000)

What is the amount of economic resources expended in relation to the level of effectiveness achieved (or how cost effective is the user)?

Therefore,

Economic efficiency (EE)= Effectiveness/Total Cost

Where,

Total Cost = Labor costs of user's time + Cost of resources and equipment used + Cost of training required by the user etc.

Interpretations

0<= EE

The higher the value of EE is the better economic efficiency.

2.4 Relative user productivity (ISO/IEC 9126-4,2000) **or Relative User Efficiency** (Bevan, Macleod, 1994)

How productive is a user compared to an expert?

Relative user productivity X = A / B

A = ordinary user's task productivity

B = expert user's task productivity

Interpretations

$0 <= X <= 1$

The closer to 1.0 is the better

2.5 Essential Efficiency (Constantine, Lockwood, 1999)

It is a measure of how closely a given user interface design approximates the ideal expressed in the essential use case model.

Essential Efficiency (EE) $=100 \cdot S_{essential} / S_{enacted}$

Where,

$S_{essential}$ = Number of user steps in the essential use case narrative

$S_{enacted}$ = Number of steps needed to perform the use case with a particular user interface design (rules for counting the number of enacted steps are in the reference)

There is an*other metric* related to this one, which is

2.6 Weighted essential Efficiency (Constantine, Lockwood, 1999)

It measures the overall efficiency of a design for an entire mix of tasks and is given by:

$$EE_{weighted} = \sum_{\forall i} p_i \cdot EE_i$$

Where,

p_i = Probability or weighted importance) of task i

EE_i = Essential efficiency for task i

Interpretations

It has been noticed that EE does not exactly cover the range of values from 0 to 100%. (Constantine, Lockwood, 1999) points that for very poor interface implementations, it can become very small but cannot reach 0% in practice. However, in practice EE could exceed 100% when a clever and highly efficient design supports a poorly worked out essential use case.

2.7 Completion Rate Efficiency or Completion Rate/Mean Time-On-Task (Common Industry Format, 2001)

It specifies the percentage of users who were successful (or percentage goal achievement) for every unit of time. It is the central measure of efficiency.

Interpretations

As the time on task increases, one would expect users to be more successful. A very efficient product has a high percentage of successful users in a small amount of time.

3. Metrics to measure Attractiveness

3.1 Layout uniformity (Constantine, Lockwood, 1999)

As the name suggests, this metric assesses the uniformity or regularity of the user interface layout. It measures how well the visual components of interface are arranged.

Layout Uniformity (LU) is defined as:

$$LU = 100 \bullet \frac{(1-(N_h+N_w+N_t+N_l+N_b+N_r)-M)}{6 \bullet N_c - M}$$

Where,

M is an adjustment for the minimum number of possible alignments and sizes needed to make the value of LU range from 0 to 100, given by

$$M = 2 + 2 \bullet \lceil 2\sqrt{N_{components}} \rceil$$

N_c=Number of components

N_h= Number of different heights of visual components

N_w= Number of different widths of visual components

N_t = Number of different top-edge alignments of visual components

N_b= Number of different bottom-edge alignments of visual components

N_l = Number of different left-edge alignments of visual components

N_r= Number of different right-edge alignments of visual components

$\lceil \; \rceil$, It is the ceiling function; which means the smallest integer greater than the enclosed value.

Interpretations

0 <= LU <= 100

A review of well-designed dialogues suggests that, in general, a value of LU anywhere between 50% and 85% is reasonable.

3.2 Horizontal / Vertical balance (Sears, 1995)

It evaluates how well balanced the screen is both vertically and horizontally and it is given by:

$$\text{Balance} = 200 \cdot \frac{W1}{(W1+W2)}$$

W1 = Weight of side one

W2 = Weight of side two

Where,

Weight of a side = Number of pixels used × side's distance from the center

Center = Halfway between the left edge of the left-most widget and the right edge of the right-most widget.

Interpretations

An optimal score for this metric is 100, and it means the screen is perfectly balanced.

3.3 User Subjective Rating or Attractive interaction (ISO/IEC 9126-2, 2001)

How attractive is the interface to the user?

Attractive Interaction (X) = A

A= Questionnaire to assess the attractiveness of the interface to users, after experience of usage

Interpretations

Higher the final score is the better.

3.4 **Interface Appearance Customizability** (ISO/IEC 9126-2,2001)

What proportion of interface elements can be customized in appearance to the user's satisfaction?

Interface Appearance Customizability (X) = A / B

A= Number of interface elements customized in appearance to user's satisfaction

B= Number of interface elements that the user wishes to customize

Interpretations

$0 <= X <= 1$

The closer to 1.0 is the better.

4. Metrics to measure Likeability

4.1 **Satisfaction scale** (ISO/IEC 9126-4,2000)

How satisfied is the user?

X = A

Where,

A = questionnaire producing psychometric scales

(Examples of psychometric questionnaires can be found in reference).

4.2 **Satisfaction questionnaire** (ISO/IEC 9126-4,2000)

How satisfied is the user?

X = A

A = questionnaire responses

Interpretations

The more positive response is the better.

4.3 **Discretionary Usage** (ISO/IEC 9126-4,2000)

Does user like to use software frequently?

User's operational frequency (X) = A/B

A= number of times that the user uses the specific software functions

B = number of opportunities to use the specific software functions

Interpretations

$0<X$

The more is the better.

4.4 **Percent of favorable / unfavorable user comments** (Dix et al. , 1993)

Interpretations

The more is this ratio is the better likeability.

4.5 **Number of users preferring your system** (Dix et al., 1993)

Interpretations

More is the better.

For the following metrics, the desirable value is 0.

4.6 **Number of regressive behaviors** (Dix et al., 1993)

4.7 **Number of times the user is disrupted from a work task** (Dix et al., 1993)

4.8 **Number of the users expressing frustration** (Dix et al., 1993)

5. Metrics to measure Flexibility

5.1 **Flexibility Functions** (Donyaee, 2001)

It is a measure of the number of functions offered for changing the interface environment.

X=A/B

Where,

A= Number of interface functions that allow the user to change the default values

B= Total number of functions offered to the user

Interpretations

$0<=X<=1$

The closer to 1.0 is the better.

5.2 Number of ways to perform a task (Author)

Interpretations

There is lower limit only for this metric, so the interpretation could be made as:

More than 1, the better is the flexibility.

5.3 Images with alternative text (Olsina et al., 1999)

It is a measure of the number of images that have alternative text available.

X= A/B

Where,

A= Number of images with alternative text.

B= Total number of images

Interpretations

0<= X <=1

The closer to 1.0 is the better.

Similar metrics can be:

5.4 Graphic icons with alternative text (Olsina et al., 1999)

5.5 Audio/Video files with alternative text (Olsina et al., 1999)

A metric considering the summation of these three metrics can be:

Percentage of the presence of alternative text for non-text elements

It is a measure of the percentage of presence of alternative text for non-text elements (like images, graphic icons, audio/video files etc.)

X=(A/B)*100

Where,

A= Number of non-text elements with alternative text

B= Total number of non-text elements.

Interpretations

0<= X <=100

5.6 Number of languages supported by the site (Author)

Interpretations

For international purposes, the more the value of this metric the better is the flexibility.

6. Metrics to measure Minimal Action

6.1 Task Time (Dix et al. , 1993)

Time taken to complete a task

Interpretations

Lesser is the better.

6.2 Number of commands used (Dix et al., 1993)

Number of commands (X) used to perform a task

X=A

Where,

A=Number of commands used to perform a task

Interpretations

The closer to 1 is the better

6.3 Number of repetitions or failed commands (Dix et al., 1993)

Interpretations

The closer to 0 is the better.

6.4 Task Concordance (Constantine, Lockwood, 1999)

It is an index of how well the distribution of task difficulty using a particular interface design fits with the expected frequency of various tasks.

TC (Task Concordance) is given by the formula as:

TC =100 • D/P

Where,

D= Discordance score, which is the number of pairs of tasks ranked in the correct order by enacted length less number of pairs out of order (For more information on enacted length, please see reference)

P = Number of possible task pairs

If every task has a different difficulty or enacted length, then

$P= N (N-1)/2$

Where, N= Number of tasks being ranked

Interpretations

TC ranges from the –100% to +100%.

+100% TC means that the more frequent tasks are always shorter than the less frequent tasks in a design.

-100% TC means that more frequent tasks are taking more steps i.e. the design is basically backwards.

0% TC or close to it means that the design is essentially random or unrelated to the tasks to be supported.

6.5 **Layout Appropriateness** (Sears, 1993)

It measures the appropriateness of a given layout and is computed by weighting the cost of each sequence of actions by how frequently the sequence is performed.

$$\text{Layout Appropriateness (LA)} = 100 \cdot \frac{C\_optimal}{C\_proposed}$$

Where,

C_optimal = Cost of the LA optimal layout

C_proposed = Cost of the proposed layout

And Cost (C) for a specific layout is given as:

$$C = \sum_{\forall i \neq j} F_{i,j} \cdot D_{i,j}$$

Where,

$F_{i,j}$ = Frequency of transition between visual components i and j

$D_{i,j}$ = Distance between visual components i and j

Interpretations

High LA means a layout where visual components that are most frequently used in succession are packed together, reducing the expected time (cost) of completing a mix of tasks.

7. Metrics to measure Minimal Memory Load

7.1 **Visual Coherence** (Constantine, Lockwood, 1999)

It measures how well a user interface keeps related things together and unrelated things apart. It is based on the principle that well structured interfaces group together components that present closely related concepts. Total Visual Coherence (VC) of a design for an interaction context is computed by

$$VC = 100 \cdot \left(\frac{\sum_{\forall k} G_k}{\sum_{\forall k} N_k \cdot (N_k - 1)/2} \right)$$

With

$$G_k = \sum_{\forall i,j \mid i \neq j} R_{i,j}$$

Where,

N_k = Number of visual components in group k

$R_{i,j}$ = Semantic relatedness between components i and j in group k,

$0 \leq R_{i,j} \leq 1$

In practice $R_{i,j} = 1$ if components i and j belong to the same semantic cluster i.e. they are substantially related.

$R_{i,j} = 0$ if components i and j are not related

Interpretations

$0 <= VC <= 100$

7.2 **Task visibility** (Constantine, Lockwood, 1999)

It measures the fit between the visibility of features and the capabilities needed to complete a given task or set of tasks. (Constantine, Lockwood, 1999) says that Task Visibility is a metric based on Visibility Principle, which means that user interfaces should show users exactly what they need to know or to use to complete a given task. In other words, it measures the percent of necessary features (objects or elements) to complete a task or a use case that are visible to the user.

Task Visibility (TV) is given by:

$$TV = 100 \cdot \left(1/S_{total} \cdot \sum_{\forall i} V_i\right)$$

Where,

S_{total} = Total number of enacted steps to complete use cases

V_i = Feature visibility (0 to 1) of enacted step i

Interpretations

TV ranges from 0 to 100%.

100% TV means that everything needed for a step is visible directly on the user interface as seen by the user at that step.

0% TV can be there only under very exceptional circumstances. For example: remote access to highly sensitive information on a high security interface, where the user must have the pre knowledge of the command order.

7.3 **Layout Uniformity** (Constantine, Lockwood, 1999)

Please refer 3.1

7.4 **Interface shallowness** (Yamada et al., 1995)

It measures the degree of heaviness of the cognitive load on users.

$$\text{Interface Shallowness (ISh)} = \frac{n\,(n-1)^2}{n\,(n-1) - \sum_i IDp_i}$$

Where

n = Number of nodes (> 1)

IDpi =Sum of the values of the IDs (Interface Distances) for the shortest path from the root to node i.

The Interface Distance (IDs) is defined as follows:

$IDs_{i,j}$ (Interface Distance)= 0, if nodes i and j are displayed linearly i.e. at the same level, otherwise 1

7.5 **Number of icons** (Donyaee, 2001)

Interpretations

The more is the number of icons, more is the user's memory load to recognize and distinguish between them. So the good idea is to keep the number of icons between some thresholds.

7.6 Number of acronyms, which are not found in a general-purpose dictionary (Donayee, 2001)

It is a measure of the number of abbreviated words used whose meaning is not clear to the user.

Interpretations

The closer to 0 is the better.

8. Metrics to measure Operability

8.1 Number of commands (Dix et al., 1993)

Number of commands (X) used to perform a task

X=A

Where,

A=Number of commands used to perform a task

Interpretations

The closer to 1 is the better

8.2 Interface shallowness (Yamada et al., 1995)

Please refer 7.4

8.3 Number of frames (Donayee, 2001)

Interpretations

Number of frames should be 0 because the browser can show several pages in different frames at the same time.

8.4 Input-undo ability (ISO/IEC 9126-2,2001)

What proportion of the functions has input-undo ability (X)?

X=A/B

Where,

A= Number of input-undo functions

B= Total number of functions available on the interface

Interpretations

0<= X <=1

The closer to 1.0 is the better.

8.5 Error-undo ability (ISO/IEC 9126-2,2001)

What proportion of the functions has error-undo ability (X)?

X=A/B

Where,

A= Number of error-undo functions

B= Total number of functions available on the interface

Interpretations

0<= X <=1

The closer to 1.0 is the better.

8.6 Customizability (ISO/IEC 9126-2,2001)

What proportion of functions can be customized?

Customizability (X)= A / B

A= Number of functions successfully customized

B= Total number of functions available to user.

Interpretations

0 <= X <= 1

The closer to 1.0 is the better.

8.7 **Productive proportion** (ISO/IEC 9126-4,2000)

What proportion of the time the user is performing productive actions?

Productive proportion (X)= Ta / Tb

Where,

Ta = productive time = task time - help time - error time – search time

Tb = task time

Interpretations

0<= X <=1

The closer to 1.0 is the better.

9. Metrics to measure User Guidance

9.1 **Rate of wizards** (Donyaee, 2001)

It is a measure of the number of wizards used for complex operations.

Rate of Wizard (RW)= N1/N2

Where,

N1= number of provided wizards for complex operations

N2= total number of complex operations

Interpretations

RW closer to 1.0 is the better.

9.2 **Rate of cancel** (Donyaee, 2001)

RC = N1 / N2

N1 = number of provided cancel feature

N2 = total number of operations

Interpretations

The more is the better.

9.3 **Visual coherence** (Constantine, Lockwood, 1999)

Please refer 7.1

9.4 **Rate of error messages** (Author)

It is a measure of the number of error messages displayed each time the user encountered a problem.

X = A/B

Where,

A= Number of error messages displayed

B= Number of problems encountered

Interpretations

X closer to 1.0 is the better.

9.5 **Number of times the interface misleads the user** (Author)

Interpretations

This number should be 0.

10. Metrics to measure Understandability

(ISO/IEC 9126-2,2001) defines the following set of understandability metrics. Actually, it is a list of external product quality metrics, but when it is seen through the Quality In Use perspective, these metrics address this issue also.

10.1 **Completeness of description** (ISO/IEC 9126-2,2001)

What proportion of functions (or types of functions) is understood after reading the product description?

Completeness of description (X) = A / B

Where,

A = Number of functions (or types of functions) understood

B = Total number of functions (or types of functions)

Interpretations

0<=X<= 1

The closer to 1.0 is the better.

10.2 **Function understandability** (ISO/IEC 9126-2,2001)

What proportion of the product functions will the user be able to understand correctly?

Function Understandability (X)= A / B

Where,

A= Number of interface functions whose purpose is correctly described by the user

B= Number of functions available from the interface

Interpretations

0 <= X <= 1

The closer to 1.0 is the better.

10.3 **Frequency of using help and documentation** (ISO/IEC 9126-2,2001)

What is the frequency of use of help (online help or human assistance) and documentation?

If measured during first use, this metric is related to learnability.

X= A

Where,

A = Number of times the help and documentation functions are used.

Interpretations

0<= X

The closer to 0 is the better.

10.4 Time spent using help and documentation (ISO/IEC 9126-2,2001)

X = T, Where T is the time taken in seconds or minutes

Interpretations

The closer to 0 is the better.

Other *metrics* to measure understandability are:

10.5 Number of acronyms, which are not found in a general-purpose dictionary (Donyaee, 2001)

Please refer 7.6

10.6 Overall Density (Tullis, 1984)

It is a measure of the percentage of the display used to present information.

Interpretations

Crowded screens decrease the level of understandability as said by (Mayhew, 1992). Experimental results by (Mayhew, 1992) had shown that Overall Density should not exceed 40%

10.7 **Local Density** (Tullis, 1984)

It measures the percentage of the space used within each individual group of items.

Interpretations

(Mayhew, 1992) had experimentally demonstrated that Local Density should not exceed 62% for good understandability.

10.8 **Relative User Efficiency** (Bevan, Macleod, 1994)

To measure the Learnability of a particular user, the Relative User Efficiency metric gives the position on the learning curve that the user has reached.

Please refer 2.4

11. Metrics to measure Consistency

(Mahajan, Shneiderman, 1997) have given a number of metrics to measure the consistency of dialog boxes as:

11.1 Aspect Ratio

It is given by the formula as:

$$\text{Aspect Ratio (AR)} = \frac{\text{Height of a dialog box}}{\text{Width of dialog Box}}$$

Interpretations

Aspect Ratio (0.5,0.8) is desirable.

Dialog boxes that perform similar functions should have the same aspect ratio.

11.2 Percentage of NonWidget Area

It is the percentage ratio of the nonwidget area to the total area of the dialog box.

$$\text{Percentage of NonWidget Area (X)} = \frac{\text{NonWidget Area}}{\text{Total Area of Dialog Box}} * 100$$

Interpretations

30 < X<= 100

X closer to 100 means good utilization of space and X less than 30 means need for redesigning.

11.3 Widget Density

It is a measure of the crowding of widgets in the dialog box.

It is given by the ratio of the number of top-level widgets to the total area of the dialog box (multiplied by 100,000 to normalize it).

Interpretations

Widget Density greater than 100 means a comparatively large number of widgets is present in a small area.

11.4 Margins

It is the number of pixels between the dialog box border and the closest widget.

Interpretations

For good consistency, in and across different dialog boxes the left, right, top, and bottom margins should all be equal.

11.5 Gridedness

Gridedness is a measure of alignment of widgets.

There are two measures of the Gridedness:

X-Gridedness = Number of stacks of widgets with the same X coordinates (excluding labels).

Y-Gridedness = Number of stacks of the widgets with the same Y coordinates.

Interpretations

High values of X-Gridedness and Y-Gridedness indicate the possibility of misaligned widgets.

Note

A clear definition of alignment along with the example can be taken from (Vanderdonckt, Gillo, 1994).

11.6 Area Balances

It is a measure of degree of spread ness of the widgets over the dialog box. There are two measures:

$$\text{Horizontal Balance} = \frac{\text{Total widget area in the left half of the dialog box}}{\text{Total widget area in the right half of the dialog box}}$$

$$\text{Vertical Balance} = \frac{\text{Total widget area in the top half of the dialog box}}{\text{Total widget area in the bottom half of the dialog box}}$$

Interpretations

High values of balances between 4.0 and 10.0 indicate screens are not well balanced. The limiting value 10.0 represents a blank or almost blank dialog box.

The *other lists of metrics* that can be used to measure consistency are:

11.7 Layout uniformity (Constantine, Lockwood, 1999)

Please refer 3.1

11.8 Layout Complexity (Tullis, 1984)

It is the extent to which the arrangement of items on a screen follows a predictable visual scheme.

11.9 Overall Density (Tullis, 1984)

Please refer 10.6

11.10 Local Density (Tullis, 1984)

Please refer 10.7

11.11 Number of groups (Tullis, 1984)

It is the number of groups of items on the screen. The optimal value needs to be determined.

11.12 Size of group (Tullis, 1984)

It is measure of the number of items per group. The reasonable size of group has to be determined experimentally.

11.13 Number of items (Tullis, 1984)

The number of individual items on the screen.

11.14 Color of Hyperlinked Text

It is a measure of the usage of standard color for hyperlinked text.

Interpretations

The standard color should be used for hyperlinked text so that the user could easily recognize and differentiate the hyperlinked text from the normal underlined text.

Therefore,

Color of Hyperlinked Text = Blue (By Default)

For the same reason the *other two related metrics* are:

Color of the Selected Hyperlinked Text = Red

Color of the Visited Hyperlinked Text = Purple

- If all the three color combinations satisfy for a webpage or hypertext document, then its Hypertext Color Consistency =100%
- If two conditions satisfy, then Hypertext Color Consistency = 67%
- If one color condition satisfy, then Hypertext Color Consistency =33%
- If none, then Hypertext Color Consistency = 0%

11.15 Number of distinct foreground colors

Interpretations

Use of one distinct foreground color throughout all the screens of the software product enhances color consistency.

11.16 Number of distinct background colors

Interpretations

Similar Interpretations can be made for the background color.

11.17 Number of date formats

Interpretations

One date format is required to be used for the software product, otherwise inconsistency may occur and user may get confused. Moreover date formats should obey the cultural aspects of any country where software product is supposed to be used.

11.18 Number of units of measurement per specific item (Author)

Interpretations

It should be 1 and should not change from page to page or screen to screen.

11.19 Number of time formats

Interpretations

There should be one time format depicted at a time and it should obey the culture context of its use.

11.20 Number of font types used

Interpretations

Use of the same font type on all the screens is recommended to get a highly consistent interface.

12. Metrics to measure Self-Descriptiveness

12.1 Self-explanatory error messages (ISO/IEC 9126-2,2001)

In what proportion of error conditions does the user propose the correct recovery action?

$X = A / B$

A =Number of error conditions for which the user proposes the correct recovery action

B =Number of error conditions encountered per task

$0 <= X <= 1$

Interpretations

The closer to 1.0 is the better.

Other metrics to measure self – descriptiveness of websites can be:

12.2 Number of self-explanatory icons on the website (Author)

12.3 Number of self-descriptive statements on the home page (Author)

Interpretations

The presence of self-descriptive icons or statements is a positive aspect of any website.

13. Feedback metrics

13.1 Number of confirmation notices or email confirmations per successful transaction = 1(Author)

13.2 Number of feedback functions per site (Author)

Interpretations

There should be feedback functions (like contact name, address, email etc.) on every page of a website.

13.3 **Freshness factor** (Sterne, 2001)

It is a measure of whether the regular changes are made to the site. It is needed for two reasons: 1) Visitors want to see fresh material, and 2) Updating the site often will ensure that the information is current.

It is given by

Freshness Factor (FF)= (Average content area refresh rate) / (Average session visit frequency)

Interpretations

$1 <= FF <= 1.5$

A freshness factor (FF) less than 1 means customers sees stale content. They visit more often than the content changes.

A freshness factor (FF) greater than 1.5 means the content changes more often than visitors show up to view it. That means resources are not consumed properly.

14. Metrics to measure Accuracy

14.1 **Error frequency** (ISO/IEC 9126-4,2000)

What is the frequency of errors?

Error Frequency (X) = A

A = number of errors made by the user

Interpretations

$0 <= X$

The closer to 0 is the better.

This metric is only appropriate for making comparisons if errors have equal importance, or are weighted.

14.2 **Task effectiveness** (ISO/IEC 9126-4,2000)

What proportion of the tasks is completed correctly?

M1 = A x B

A= Quantity (completeness) = the proportion of task goals represented in the output of task

B= Quality (correctness) = the degree to which the task goals represented in the output have been achieved

Interpretations

0<= M1 <=1

The closer to 1.0 is the better.

15. Metrics to measure Completeness

15.1 **Task completion** (ISO/IEC 9126-4,2000)

What proportion of the tasks is completed?

X = A/B

A = number of tasks attempted

B = total number of tasks

Interpretations

0<= X <=1

The closer to 1.0 is the better.

This metric can be measured for one user or a group of users. If tasks can be partially completed the Task effectiveness metric should be used.

15.2 Percent of task completed per unit time (Dix et al., 1993)

Interpretations

The more is the better.

15.3 Broken Link Count (Olsina et al., 1999)

It is a measure of number of links that lead to missing destination.

Interpretations

It should be 0.

16. Metrics to measure Fault Tolerance

Metrics in the area of software fault tolerance (or software faults) are generally pretty poor. However, the basic metrics are:

16.1 Incorrect operation avoidance (ISO/IEC 9126-2,2001)

How many functions are implemented with incorrect operations avoidance capability?

Rate of Incorrect Operation Avoidance (X) = A / B

Where,

A= Number of avoided critical and serious failures occurrences

B= Number of functions available to user

Interpretations

$0 <= X <= 1$

The closer to 1.0 is better

16.2 **Error-undo ability** (ISO/IEC 9126-2,2001)

What proportion of the functions has error-undo ability?

X=A/B

Where,

A= Number of error-undo functions

B= Total number of functions available on the interface

Interpretations

0<= X <=1

The closer to 1.0 is the better.

16.3 **Rate of Cancel** (Donyaee, 2001)

Please refer 9.2

17. Metrics to measure Resource Safety

17.1 **User health and safety** (ISO/IEC 9126-4,2000)

What is the incidence of RSI (Repetitive strain injuries) among users of the product?

User Health and Safety (X) = A / B

A = number of users reporting RSI

B = total number of users

Interpretations

0<= X <=1

The closer to 0 is the better.

17.2 **Patient safety** (ISO/IEC 9126-4,2000)

What is the risk of hazard to the patient?

Patient Safety (X) = A / B

A = number of patients receiving incorrect treatment

B = total number of patients

Interpretations

$0 <= X <= 1$

The closer to 0 is the better.

18. Metrics to measure Readability

18.1 Color Contrast

Text should have high contrast to be easily readable. This means that the color or brightness of the text should be as different as possible to the background it is placed upon. 90% contrast difference between foreground and background is required in order to read text clearly.

Interpretations

Black text on a white background is the best option.

Background color = white (default)

Color of text = black (default)

18.2 Font

Interpretations

(Bernard, Mills, 2000) have conducted a research on readability of fonts and have evaluated that for good online readability, the font size and type should be:

Font Type= Arial (default)

Font Size = 12 (default)

18.3 Color of Hyperlinked text

Interpretations

The color of the hyperlinked text should be kept standard; as by recognizing the color users can easily understand that this is the hyperlinked text and it helps in readability also.

Color of hyperlinked text = blue (default)

For the same reason the *other two related metrics* are:

Color of the Selected Hyperlinked Text = Red

Color of the Visited Hyperlinked Text = Purple

18.4 Number of horizontal lines across the web page

Interpretations

NIST's WebSAT [73] checklist tells that when a horizontal line is placed across the web page, user might be confused and thinks he/she has reached the end of the web page. So in their guideline for good readability,

Number of horizontal lines across the web page = 0

18.5 Number of animated text objects

Interpretations

The use of animation is currently a contradictory subject. (Spool, 1998) reports that users frequently cover animated objects with their hands since the movement interferes with reading, i.e.

Number of animated text objects = 0

18.6 **Flesch Reading Ease Score** (Readability Scores) [77]

It rates text on a 100-point scale; the higher the score, the easier it is to understand the document. It is given by the following formula:

$$\text{Flesch Reading Ease score} = 206.835 - (1.015 \times ASL) - (84.6 \times ASW)$$

Where,

ASL = average sentence length (the number of words divided by the number of sentences)

ASW = average number of syllables per word (the number of syllables divided by the number of words)

Interpretations

For most standard documents, aim is to score approximately 60 to 70.

18.7 **Page Width**

Interpretations

Page width also contributes to the readability of the web page. Span of a person's eye movement is normally about 3 inches, while most web pages are at least 6 inches. Most of the usability professionals recommend a web page width of <470, 625 >pixels.

18.8 **Number of columns** (Author)

Interpretations

Columns divide the screen into various sections and it decreases the readability to some extent.

The *other metrics* that need interpretations can be:

18.9 Percentage text density

It is the percentage of display used to present the textual information.

18.10 Percentage grouping

It is percentage of the number of groups of information present on the screen.

18.11 Number of words per line

19. Metrics to measure Controllability

19.1 Flexibility Functions (Donyaee, 2001)

Please refer 5.1

19.2 Number of Frames (Donyaee, 2001)

Please refer 8.3

19.3 Input-undo ability (ISO/IEC 9126-2,2001)

Please refer 8.4

19.4 Error-undo ability (ISO/IEC 9126-2,2001)

Please refer 8.5

19.5 Number of emergency exits per error condition (Author)

Interpretations

At least one exit per error condition is required for the proper functioning of the software product.

20. Metrics to measure Navigability

20.1 Self-referential pages (Author)

How many pages of a website have links back to themselves?

$X=A/B$

A= Number of pages having links to themselves

B= Total number of pages of a website

Interpretations

$X<=0$

The closer to 0 is the better.

20.2 Home page reference (Author)

How many pages of the website have links back to home page?

$X= A/B$

A= Number of pages having links back to home page

B= Total number of pages of a website

Interpretations

$0<= X <=1$

The closer to 1.0 is the better.

20.3 Size of the graphic links

Interpretations

For some of the users (mobility-impaired) it is hard to move the mouse with precision, therefore there should be some minimum size of graphic links. The web accessibility checklist as listed on web page of university of Minnesota [102] contributes that size of the graphic links should be at least (0.5 X 0.5) inches.

20.4 Broken Link Count (Olsina et al., 1999)

It is a measure of number of links that lead to missing destination.

X= (A/B)

A= Number of broken links

B= Total number of links of a website

Interpretations

X<=0

The closer to 0 is the better.

(Olsina et al., 1999) have added one more metric called *Elementary Quality Preference (EP)* as:

EP = 1 (or 100%) if X = 0; EP = 0 (or 0%) if X >= X_{max};

Otherwise, EP = (X_{max} – X) / X_{max} if 0 < X < X_{max}

Where X_{max} is some agreed upper threshold such as 0.06

They have interpreted this elementary quality preference into three acceptability levels namely:

- Unsatisfactory (from 0 to 40%),
- Marginal (from 40 to 60%),
- Satisfactory (from 60 to 100%)

20.5 Orphan Pages (Olsina et al., 1999)

It is a measure of number of pages that have no internal link to the site where they are included.

X= (A/B)

A= Number of orphan pages of a website

B= Total number of pages of a website

Interpretations

X<=0

The closer to 0 is the better.

A similar metric can be *Orphan Links*, where it is a measure of number of links that force the users back to where they came from. For good navigability, its value should also be zero.

20.6 Words per Content Link

How many words are there in the content link?

X= Number of words per content link

Interpretations

(Spool, 1999) discovered that 7-12 words in a content link is the ideal for helping users to be successful in finding information.

20.7 Page Size

Interpretations

<10K, 50K> (Tiller, Green, 1999)

20.8 Number of pages per site

Interpretations

An article on "Web Usability"[103] defines number of pages per site according to the site's purpose as follows:

Number of pages	Example genres
1-10	Personal bio, project, course outline
5-50	Scientific paper, conference program
50-500	Book, city guide, product catalogue
500-5,000	Photo library, museum tour
5,000-50,000	University guide, newspaper site
50,000-500, 000	Telephone directory
500,000-5, 000,000	Journal abstracts, parliament record
Over 5,000,000	Library of Congress, NASA archives

20.9 Number of ambiguous links

Interpretations

Ambiguous links like "click here" cause the user to go to the wrong page. So they must be avoided. Therefore,

Number of ambiguous links =0

There are *many more metrics* to measure navigability aspects and they need interpretations as research is still going on to find the optimal values for these metrics like the following:

20.10 Number of commands / mouse clicks to complete the task (Dix et al., 1993)

20.11 Number of links

20.12 Number of graphic elements

20.13 Longest Depth

Depth is the measure of the path traversed in the information hierarchy from one point to other.

20.14 Average links per page

20.15 Number of levels of menu hierarchy (Author)

20.16 Selection time (Author)

Please refer 1.3

(Babiker et al., 1991) have defined the following metric to measure the *navigability of hypertext documents* and they named it as "Access and Navigation Parameter (A) " and proposed the following formula:

$$A = a_1 * O / (a_2 * Ta + a_3 * Ksa + a_4 * Era)$$

Where,

Ta = Performance time

Ksa = Keystroke time

Era = Number of errors in performing a navigation task.

a_1, a_2, a_3, a_4 are weights.

"O" is the orientation contribution and is given by:

$$\text{Orientation } (O) = 1 / (w_1 * To + w_2 * Ero)$$

Where,

To = Time spent by a user in order to return to specific location from his current location

Ero = Number of errors committed by the user

w_1 and w_2 are weights that reflect the importance of each attribute.

21. Metrics to measure Simplicity

21.1 Number of commands/mouse clicks required per task (Dix et al., 1993)

Interpretations

The best score here is 1 in terms of the simplicity.

21.1 Number of panels (Author)

Interpretations

The lower this number is, the better. An optimal upper and lower limit for this metric has yet to be determined.

21.3 Page Width

Interpretations

Many usability professionals agree on the range < 470, 625>pixels.

21.4 Task Concordance (Constantine, Lockwood, 1999)

Please refer 6.4

21.5 Economy [69]

Economy is a measure of how economical the screen is and is given by

$ECM = 3/(N_{size} + N_{color} + N_{shape})$

Where,

N_{size}= Numbers of sizes used

N_{color} = Numbers of color used

N_{shape} = Numbers of shapes used

Interpretations

Economy is achieved by using few styles, display techniques, and colors as possible.

0<= ECM <=1

22. Metrics to measure Privacy

Privacy is one of the latest issues of web usability that have to be dealt with. More research has to be done before providing a standard set of privacy metrics. However, the general elements of the website that are seen by the users while looking for their privacy concerns are:

22.1 Number of validated privacy icons on the home page

Privacy icons like shown below which are validated by the trusted third parties create a positive affect on the attitude of the user.

Interpretations

The more this number is the better privacy.

22.2 Privacy Statement

(IBM, 1991) has surveyed that over 70% of the Internet users believe that it is very important for a website to display a privacy notice before they would be willing to make a purchase.

Interpretations

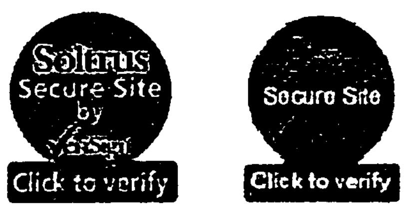
A clear and concise privacy statement on the website is required to enhance privacy of users.

23. Metrics to measure Security

Like the Privacy metrics, security is somewhat internal to the software product and it is hard for the user to judge the website security. Users can simply look at the security icons or security policy statements displayed on the website to evaluate whether the security procedures have been enforced on the website. Therefore, the general elements of the websites that helps the user to recognize the security issues are depicted in terms of metrics as follows:

23.1 Number of validated security icons

Security icons like shown below which are validated by the trusted third parties (for example Veri Sign) create a positive affect on the attitude of the user.



23.2 Security Policy

Interpretations

A clear and well-defined security policy on the website contribute to user's feeling of the website's security.

23.3 Address of URL

Interpretations

The URL address of the website also contribute to security. Users prefer the site's whose address starts with "https" than "http" in terms of the security issues.

24. Metrics to measure Insurance

24.1 Refund / Return Policy

Interpretations

Users feel safe and insured if they see a clearly defined refund/return policy on the website.

25. Metrics to measure Familiarity

25.1 Color of Hyperlinked text

Interpretations

The color of the hyperlinked text should be kept standard, as users are familiar to this color terminology and can easily understand that this is the hyperlinked text.

25.2 Number of commands/tasks whose terminology is familiar to the user (Author)

Interpretations

The more this number is, the better is the familiarity.

26. Metrics to measure Loading Time

As there are many elements that contribute to page loading time, so there is not a complete formula to evaluate the total loading time. The interpretations of these metrics are hard because their values vary from page to page. Therefore, it was not possible to define upper or lower limit for these metrics. These are actually a part of those elements that contribute to the loading time of webpage.

Here is the list of the metrics from which the judgment of the loading time can be made.

26.1 Size of the HTML page text

26.2 Size of background image

26.3 Number of elements on a page that appeared on previous pages

Interpretations

Only unique or new elements are loaded from a web site, others are simply reloaded from faster disk cache.

26.4 Number of images

26.5 Total Image Size

26.6 Number of animations

26.7 Number of frames

Frames are the split screens where multiple pages can open at once.

Interpretations

Use of the frames is a contradictory concept, and to achieve universality of websites, the number of frames should be 0.

Usability Evaluations had shown a number of problems are encountered while using the frames, which are:

- Frames make it more difficult to bookmark pages.
- Frames make it hard to use the browser's backward and forward buttons.
- Printing the screen contents of framed pages is problematic.

- Frames cut the screen into windows that require excessive vertical and/or horizontal scrolling.

26.8 Number of tables per page

Interpretations

The more is the better because splitting the information into several tables helps in faster loading compared to one table content presentation.

26.9 Number of graphic files per page

26.10 Number of colors

Interpretations

256 or less is the better.

26.11 Page Size

26.12 Number of embedded files

26.13 Speed of connection

Interpretations

Users use different modes (like modems, ISDN, FDDI, T1, T3, Cable modem etc.) to connect the Internet, which vary in their speed of connection ranging from <14.4 KB/sec or less to 45 MB/sec or faster>.

(Wilson, 1998) recommends that total size of text and graphics per page should be 50 to 60 bytes, where

Size of HTML page text =5K

Size of background image=5K

Size of masthead or top-of-page graphic with company logo=20K

Size of navigation bar=8K

Size of navigation buttons=16K(where, number of navigation buttons per page =8)

Size of award logos=10K(where, number of award logos =2)

Therefore, Total size =64K

27. Metrics to measure Presentation

27.1 Iconic Window Size (Glinert,1990)

It gives a measure of the maximum number of icons that can be displayed on a screen at a time in non-overlapping manner. It is given by:

$$\text{Iconic Window Size (IWS)} = \left[\frac{MD}{MI} \right]^{1+MA}$$

Where,

MD = The diagonal measure in centimeter or pixels, of the display screen used by the environment at hand

MI = The diagonal measure of the icons used in the environment at in the same units as MD

MA = The aspect ratio of the screen used in the environment at hand

27.2 Horizontal / Vertical Balance (Sears, 1995)

Please refer 3.2

27.3 Layout Uniformity (Constantine, Lockwood, 1999)

Please refer 3.1

27.4 Layout Complexity (Tullis, 1984)

Please refer 11.8

27.5 Layout Appropriateness (Sears, 1993)

Please refer 6.5

27.6 Visual Coherence (Constantine, Lockwood, 1999)

Please refer 7.1

27.7 Flicker Rate

Interpretations

Section 508 rules [83] specify that pages shall be designed to avoid causing the screen to flicker with a frequency greater than 2 Hz and lower than 55 Hz.

27.8 Number of colors

Interpretations

256 or less is the better.

27.9 Number of font types

Interpretations

Only one font type should be used to make a consistent appealing website.

27.10 Freshness factor (Sterne, 2001)

Please refer 13.3

Other metrics whose threshold values are unknown are:

27.11 Number of columns per page

27.12 Number of images

27.13 Number of animations

27.14 Number of symbols used per page

27.15 Number of text elements within icons/symbols