AL-Tahadi University
Faculty of science
Department of computer science
Libya-Sirite G.S.P.L.A.J.

A quality assurance in local software Using McCall's model

By

Ahmed S. Mohamed Abdulla

Supervisor : Dr. Mohamed Ahmad Khlaif

A thesis submitted to the department of computer science

In partial fulfillment of the requirement for the degree of

Master of science

Academic year 2008/2009

Faculty of Science
Department of Computer Science

Title of Thesis

A quality Assurance in Local Software Using McCall's Model

By

Ahmed S. Mohamed Abdulla

Approved by:

Dr.Mohamed Ahmad Khlaif ....
(Supervisor)

Dr.Omer M. Sallabi ....
(External examiner)

Dr.Idris S. El-Feghi ....
(Internal examiner)

Countersigned by:

Dr. Ahmed Farag Mhgoub
(Dean of faculty of science)

جامعة التحدي

Dedication

To the spirit of my parents who still guiding me in my path.

Acknowledgment

With much honor, I would like to express my thanks and gratitude to my supervisor Dr. Mohamed Ahmad Khlaif for his true guidance and support that without it I would never accomplish this work. His moral support, above all, illuminated my way through my study.

I also thank the staff of Arabian Gulf Oil Company for their true cooperation specially Mr. Nouri El Hadad director of computer department. My appreciation to my friend Salem El Fakhry.

Abstract

Building and developing quality software effectively is what every company focuses on.

This thesis focuses on introducing a study on software quality as an effective instrument in managing and organizing the daily operations in many large corporation in Libya. As a product, software might be more convenient to an environment more than the other. Our interest here is to insure the compatibility of the software with our needs and to mentions its compatibility with the universal standard.

Hoping that the studies reflect a true image , and as a reference to develop the local software. we have adopted the questionnaire method in surveying and collecting data based on McCall characteristics to ensure how an efficient local software program is. We have targeted the big corporations as a field to plan our research since they can reflect a true image with their massive daily operations and that can clearly reflect the influence of the quality of software on the long run business, and because these institutions contain all parties able to judge the software as a product (user, developer and manager).

In order to complete our study with a reliable conclusion, we have chosen the (SPSS) as an analyzing program to process our collected data and Cronbach's coefficient to get the most reliable results for our survey.

After applying our study on our local organization, we have obtained an average results . generally , the results was over the average, but still the results was less than the average in some characteristics (reusability, portability, interoperability). We could have better results if we could improve the system in these characteristics.

Table of contents

List of figures

List of tables

Chapter One

Introduction

Quality is the most essential thing in the product, no matter what it might be. Software, as any other product, requires quality which is the criteria in judging the software, whether as an effective tool in work for the user and the manager, or the developer. Generally, we use the word "quality" to imply the number of characteristics that bears on its ability to satisfy the needs of customer. Software quality engineering is the science dedicated to assure the quality in the software. With their researches and improvement studies along almost the last 40 years, Institutions, scientists and studiers all over the world have done their best to enhance the performance of the software to meet all the needs and the various requirement of the user who is considered as the last hand dealing with the software and the applier of the software in real life. The multiple and diverse characteristics of the software quality create a great difficulty in assessing the quality and in developing the product to meet the different needs. It is the main objective of all studies in quality engineering. We have to know the effective qualities of the product and to understand them well to be able to maintain a systematic development and enhancement. Quality characteristics may slightly differ from one researcher to another, but still the main effective elements do not change. Their order of importance may not be same since evaluation is the reflection of many factors. The human estimation and the different needs are involved in this estimation and that call for the need of a true study on the multiple requirement in an efficient software. We will preview the different definitions and points of view in quality to explore the evolution in software quality using McCall's model in our local environment. McCall (1977) with his model of quality, Boehm (1978) , Ghezzi et *al* (1991) kan (2000) and many others along the last three decades have given their approaches, also many institutions like ISO9126-1, IEEE Standard (IEEE Std 729-1983), ANSI Standard (ANSI/ASQC A3/1978) and the German Industry Standard DIN 55350. The differentiation in their approaches helps to cover all the various attributes of quality.

1-1 Historical glance on the software engineering:

Looking at the software engineering from a historical perspective, the 1960s and earlier could be viewed as the functional era, the 1970s the schedule era, the 1980s the cost era, and the 1990s and beyond the quality era. All along these eras, we have learned how to exploit the information technology to meet industrial needs and we began to link it to the daily operations of institutions. Then the massive schedule delays and cost overruns were the aim of studies that have seen the multiple and divertive requirement of the institutions. In the 1980s hardware costs continued to decline, and the information technology permeated every facet of our institution and became available to individuals. As a competition in the industry became keen and low-cost applications became widely implemented, the importance in productivity in software development increased significantly. Various software engineering cost models were developed and used [19].

In the late 1990s, the importance of software quality was also recognized .Then, the 1990s and beyond has witnessed the quality era. With state-of-the art technology now able to provide abundant functionality, customer demand high quality. Starting the mid 1990s two major factors emerged that have proved to have an unprecedented impact on not only software engineering but also on global business environment: business reengineering for efficiency and the internet [19].

The result of the three eras with their contributions leaded us not only to better development of the software as a product, but also to higher level of evaluation and improving of the performance of our projects.

1-2-The problem with our local software systems

The problem reside in the presence of a great number of software programs with which a considerable number of official departments in Libya are operated; But there is no previous studies on the compatibility of these programs to the standard measurement. This random usage of software programs create problem in the usage and in the application of the product in our organizations. That leads us to certain questions:

i. Have we carefully studied these programs before applying them in our organization?

ii. Do these programs satisfy our needs or not?

iii. Do they comply with our market?

1-3 The aim of our study:

First, We aim to apply the international standard of quality evaluation on these local software programs to see how much do they comply to it; And that can be executed through applying one of the pioneer models of evaluating quality, and that is McCall's model, on our local institutions, and that will be through knowing the points of view of users, developers and managers.

Second, Gathering information through a questionnaire diffused on the users, developers and managers to evaluate the used software. These questionnaires will be according to the McCall's model and these collected data will be used in a statistic program to obtain their result.

Third, this study will be as a reference to develop the local software since, in case of sincerity, it will reflect the needs and the nature of the Libyan market and his organizations.

1-4 Our targeted companies and systems

The our targeted companies and systems are:

- GARYOUNIS University
- Arabian Gulf Oil Company
- Electricity Company
- Al WEHDA Bank
- The institution of Work and Labor
- Studies and examinations systems
- Payroll system.
- Others

It's obvious that these organizations have a massive daily operations, the matter that makes it suitable for our survey.

1-5 Why have we adopted McCall's model in our evaluation?

McCall's model of evaluation has survived through almost the last 30 years and that because of its global factors that did not change although receiving some improvements by some institutions. These enhancements did not change the essence of his model. That model defines software product qualities as a hierarchy of factors, criteria, and metrics. A quality factor represents a behavioral characteristic of the system. A quality criterion is an attribute of a quality factor that is related to the software production and design. A quality metric is a measure that captures some aspects of a quality criterion. Quality factors contribute to a complete picture of the software quality. One or more quality metrics should be associated with each criterion.

McCall's model Focused on three categories of software product characteristics as in figure 1.1[16]:

- Its operational characteristics (product operation)
- Its ability to undergo change (product revision)
- Its adaptability to new environments (product transition)

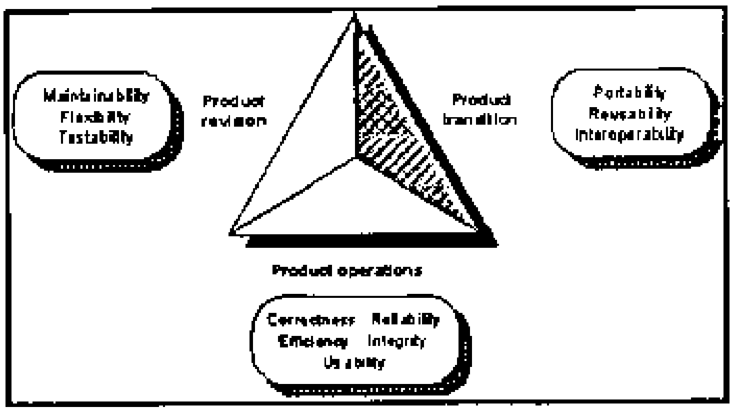

Figure 1.1:Mccall's model characteristics

That is why we have chosen his model as a base to our evaluation. His global approach can be applied in our local environment; also his quality characteristics classification is almost perfect for our survey.

1-6 The difficulty in evaluating software quality attributes

Defining the attributes of software quality and determining the metrics to assess the relative value of each attribute are not formalized processes. Because users place different values on each attribute depending on the products use, it is important that quality attributes should be observable to customers. However, with software there exists not only asymmetric information problems (where a developer has more information about quality than the user or the manager), but also instances where the developer truly does not know the quality of his own product.

However, customer satisfaction is the ultimate validation of the quality. Both product quality and customer satisfaction form the total meaning of quality. So the success of any software in real life and at the long term business is the degrees of satisfaction the software give to customer. Although the characteristics of quality are all important to software efficiency, but it's their order of importance may differs from one customer to another. This relativity comes from our hierarchy of characteristics importance that the customer or the institution itself states. In other words, the kind and degree of satisfaction varies form one customer to another. This variation may create a difficulty to the evaluator, especially if the characteristics needed are immeasurable or even affected to some extent by the human factor.

1-7 Thesis organization

This thesis is divided into five chapters to present our material relating software quality.

We started our study by giving a quick glance on the history of software engineering with its evolution through three decades of study and development, And then we briefly discussed our aim of this study and the methodology that we have adopted after giving some examples of the different methods of evaluation.

As a reason for choosing McCall for our evaluation, we explained his method that will be more clarified later.

Chapter two is completely consecrated to the different definitions of software quality and its characteristics and quality evaluation models concentrating on McCall as we have followed his evaluation of quality in our survey.

Chapter three is a presentation of quality models and factors as a base of our survey. We also presented Customer Satisfaction Surveys and the different points of view in quality by the user, the developer and the manager, with a quick explanation of the software we used (SPSS) in analyze data to getting our results and the CRONBACH alpha Coefficient used to get reliability factor .

Chapter four is a layout of the results and comments supported by the charts and analyzing the results of all characteristics.

Chapter five conclusion and recommendation .

Chapter Two

Characteristics of software quality

This chapter introduces the software quality as a base to evaluate the software efficiency in satisfying the purpose of the software itself, from the users', developers' and managers' point view, passing by the different definitions of quality given by researchers and institutions. Then we will explain the characteristics of the quality as a function in quality performance.

2-1 Software quality

Software quality is a product that has a high influence on the performance of almost all the economical and administerial establishments and companies all over the world. That is why several studies, articles and thesis have been made to define the quality and it's characteristics of different points of views.

Our study is concentrated on the quality and quality characteristics taking into consideration many studies of different organizations and authors each with his notion.

Concerning quality we first look into what quality means in the case of software development. There are many definitions of software quality. A general definition of quality is as follows [17]:

- "Software quality is the existence of characteristics of a product which can be assigned to requirements. We require a quality software product to have certain characteristics that are related to requirements (of the user) and satisfy them" Kitchenham (1989).

- According to Gilles: "quality is transparent when presented, but easily recognized in its absence" .His definition goes beyond the measurable limits to the resulted effects.
- McCall saw the quality as the following "Quality is not a single idea, but rather a multidimensional concept. The dimensions of quality include the entity of interest, the viewpoint on that entity, and quality attributes of that entity" .
- ISO 9126: "Software quality characteristic is a set of attributes of a Software product by which its quality is described and evaluated".
- German Industry Standard DIN 55350 Part 11: "Quality comprises all characteristics and significant features of a product or an activity which relate to the satisfying of given requirements".
- ANSI Standard (ANSI/ASQC A3/1978): "Quality is the totality of Features and characteristics of a product or a service that bears on its ability to satisfy the given needs" .
- IEEE Standard (IEEE Std 729-1983):"The totality of features and characteristics of a software product that bear on its ability to satisfy given needs": For example, [13,21] :

1) Conformance to specification: Quality that is defined as a matter of products and services whose measurable characteristics satisfy a fixed specification-that is, conformance to an in beforehand defined specification.

2) Meeting customer needs: Quality that is identified independent of any measurable characteristics. That is, quality is defined as the products or services capability to meet customer expectations – explicit or not.

2-2. Software quality characteristics

Characteristics play a central role in software quality. For one person, one characteristic of the product can be more important than another characteristic.

According in McCall Characteristics is set of attributes of a software product by which its quality is described and evaluated. A software quality characteristic may be refined into multiple levels of sub-characteristics [17].

2-2-1- McCall's classification of characteristics

Now, we will define the eleven characteristics of McCall for better comprehending their effect on the software performance and needs of the users [9,17,16] :

- Usability: Effort required learning, operating, preparing input, and interpret output of a program.
- Integrity: Extent to which unauthorized access to Software or data is controlled.
- Correctness: The extent to which a program satisfies its Specification and fulfills the customer's needs and objectives.
- Reliability: The extent to which a program can be expected to perform its intended function with the required precision.
- Efficiency: The amount of computing resources and codes required by a program to perform its Intended function.
- Maintainability: Effort required locating and fixing an error in a program.
- Flexibility: Effort required modifying an operational program.
- Testability: Effort required testing a program to ensure its performance of the intended function.

- Reusability: Extent to which a program or parts of a program can be reused in other applications related to the packaging and scope of the functions that the program performs.
- Portability: Effort required transferring the program from one hardware and/or software system environment to another.
- Interoperability: Effort required to couple one system to another.

2-2-2 ISO 9126 classification of characteristics

According to ISO-9126 software quality may be evaluated by the following characteristics as shown in figure 2-1[12]:

- Functionality: A set of attributes that relate to the existence of a set of functions and their specified properties.
- Reliability: A set of attributes that bear on the capability of software to maintain its level of performance under stated conditions for a stated period of time.
- Usability: A set of attributes that bear on the effort needed for use, and on the individual assessment of such use by a stated or implied set of users.
- Efficiency: A set of attributes that bear on the relationship between the level of performance of the software and the amount of resources used, under stated conditions.
- Maintainability: A set of attributes that bear on the effort needed to make specified modifications.
- Portability: A set of attributes that bear on the ability of software to be transferred from one environment to another.

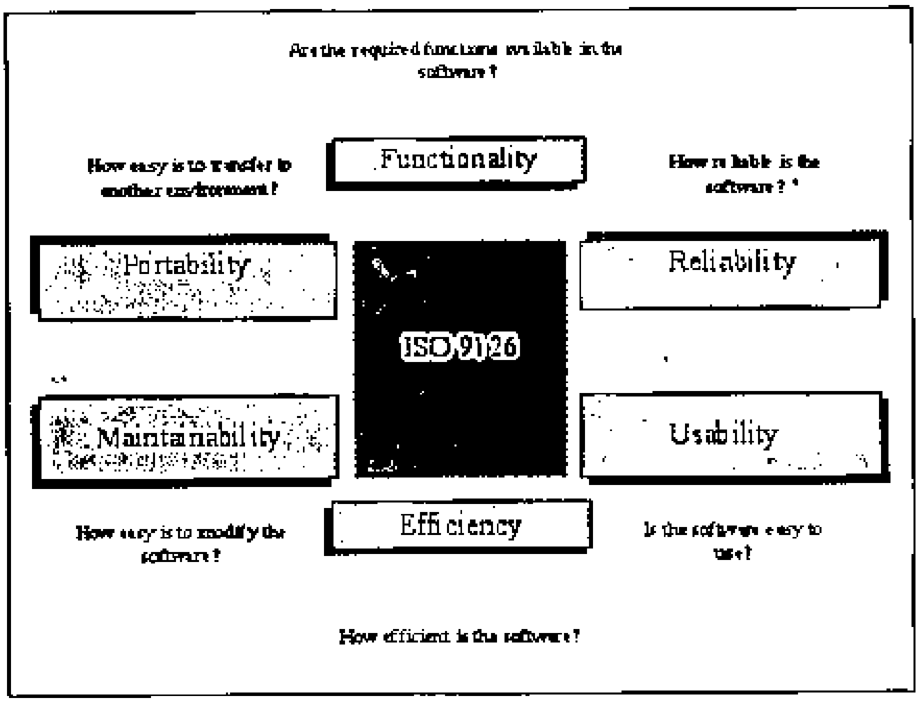
Are the required functions available in the software ?
How easy is to transfer to another environment ?
Functionality
How reliable is the software ?
Portability
ISO 9126
Reliability
Maintainability
Usability
How easy is to modify the software ?
Efficiency
Is the software easy to use?
How efficient is the software ?

Figure 2.1 ISO 9126 MODEL for software quality characteristics

2-3 Quality sub-characteristics

Some characteristic can not refer directly to their criteria, they require a extra intermediate level to be computed. Elements of this intermediate level are Sub-characteristics.

For example, in McCall's model integrity as factor refers to two sub-factors:

Access control , Access audit[9,17,16].

2- 4 Quality model

Evaluation of quality requires models to link measures of software artifacts with external, high-level, quality characteristics. First, we introduce the concept of quality model, then we present the different elements related to quality models .

The quality model consists of several quality attributes that are used as a checklist for determine software quality[8].

The benefit of Quality model is given by the decomposition of the valuable object (both a process, a product or an organization) in a list of its characteristics, sub-characteristics and measures and it is applicable both to predict/assure/verify the achievement of a defined goal about the object before/while/after producing it [15].

Each element must be clearly defined to realize it's importance in qualifying a software product.

2-4-1 Quality model definition

ISO/IEC 9126-1 defines a quality model as a framework which explains the relationship between different approaches to quality[8]. Quality models decomposes in hierarchical elements. An approach to quality is to decompose quality in Factors, Sub-factors, and criteria. Each quality sub-factors is assessed using their criteria. Finally, numerical value are assigned to quality characteristics from their quality sub-factors. Evaluation of a program begins with measuring each quality criteria with numerical value from metrics.

2-4-2Different types of models

These models have been made to illustrate the concepts of their authors in their search for software quality.

These models are classified according to the order of implement of their elements. Here are some of these models.

2-4-2-1 Hierarchical Models

We subscribe to McCall's model of software quality, which defines a three-level hierarchy of software attributes[9]. The first level is a set of generic qualitative attributes for the software product, which are frequently referred to as external quality attributes. The second level is a decomposition of each external software quality attribute into a set of quantitative factors that affect that external attribute. Finally, the third level is a set of computable metrics that we measure from the software product artifacts (such as architecture, design, or code) where each metric affects one or more quantitative factor.

The best known quality models for software are those proposed by Boehm et al. McCall et al. It is possible to distinguish quality models depending on their number of layers:

2 layers (Boehm and McCall), that show a list of characteristics subdivided in a set of sub-characteristics, up to two levels we have a description closer to user's viewpoint.

3 layers, where it is added a detailed level of single sub-characteristic measures, of course closer to a technical perspective. It allows quantifying an evaluation in a extremely subjective manner (subjectiveness is intrinsic in quality models), equally important as the objective one [1].

A software product has product artifacts that are produced along its development lifecycle (such as architecture, design, or code). It is important to distinguish which artifacts we are using in the measurement process. For instance, we can focus on metrics that pertain/are relevant to architectures. This is far different from typically code-level metrics. Architecture metrics can be obtained from software architecture description, which is usually defined using an architectural description language [18].

2-4-2-1-1 ISO 9126(1991) model

With the need for the software industry to standardize the evaluation of software products using quality models, the ISO (International Organization for standardization) proposed a standard which specifies six areas of importance for software evaluation and, for each area, specifications that attempt to make the six area measurable .

One of the advantages of the ISO 9126 model is that it identifies the internal characteristics and external quality characteristics of a software product.

However, at the same time it has the disadvantage of not showing very clearly how these aspects can be measured [15].

The layers of quality model in ISO/IEC are defined as [11]:

- Characteristics
- Sub-characteristics
- Metrics

2-4-2-1-2 McCall's Model (1976-7)

McCall's model for software quality combines eleven criteria around product operations, product revisions, and product transitions. The main idea behind McCall's model is to assess the relationships between external quality factors and product quality criteria [9,16].

McCall started with a volume of 55 quality characteristics which have an important influence on quality, and called them "factors". For reasons of simplicity, McCall then reduced the number of characteristics to the following eleven:

1. Usability.
2. Integrity.
3. Efficiency.
4. Reliability
5. Accuracy.
6. Maintainability.
7. Testability.
8. Flexibility.
9. Interface facility.

10. Re-usability.

11. Transferability.

McCall's Model is used in the United States for very large projects in the military, space, and public domain. It was developed in 1976-7 by the US Air force Electronic System Decision (ESD), the Rome Air Development Center (RADC), and General Electric (GE), with the aim of improving the quality of software products [10].

One of the major contributions of the McCall's model is the relationship created between quality characteristics and metrics, although there has been criticism that not all metrics are objective. One aspect not considered directly by this model was the functionality of the software product. The layers of quality model in McCall are defined as shown in figure 2-2 [9]:

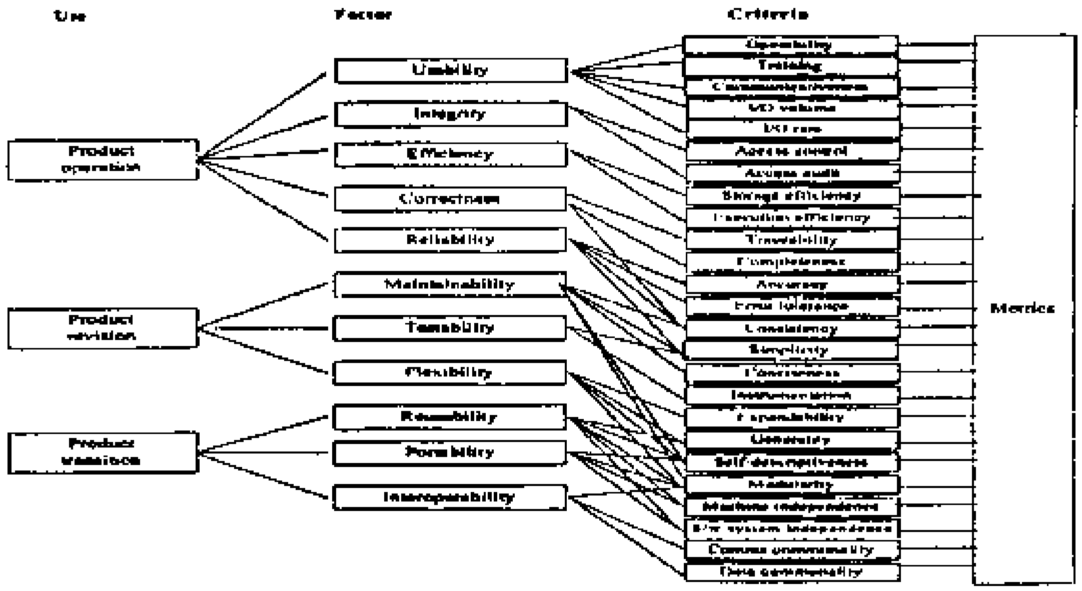

Figure 2-2 McCall's model

2-4-2-1-3 Boehm's Model (1978)

Boehm added some characteristics to McCall's model with emphasis on the maintainability of software product. Also, this model includes considerations involved in the evaluation of a software product with respect to the utility of the program[3]. The Boehm model as shown in figure 2-3 ,is similar to the McCall model in that it represents a hierarchical structure of characteristics, each of which contributes to total quality. Boehm notion includes users needs, as McCall's does; however, it also adds the hardware yield characteristics not encountered in the McCall model.

However, Boehm's model contains only a diagram without any suggestion about measuring the quality characteristics.

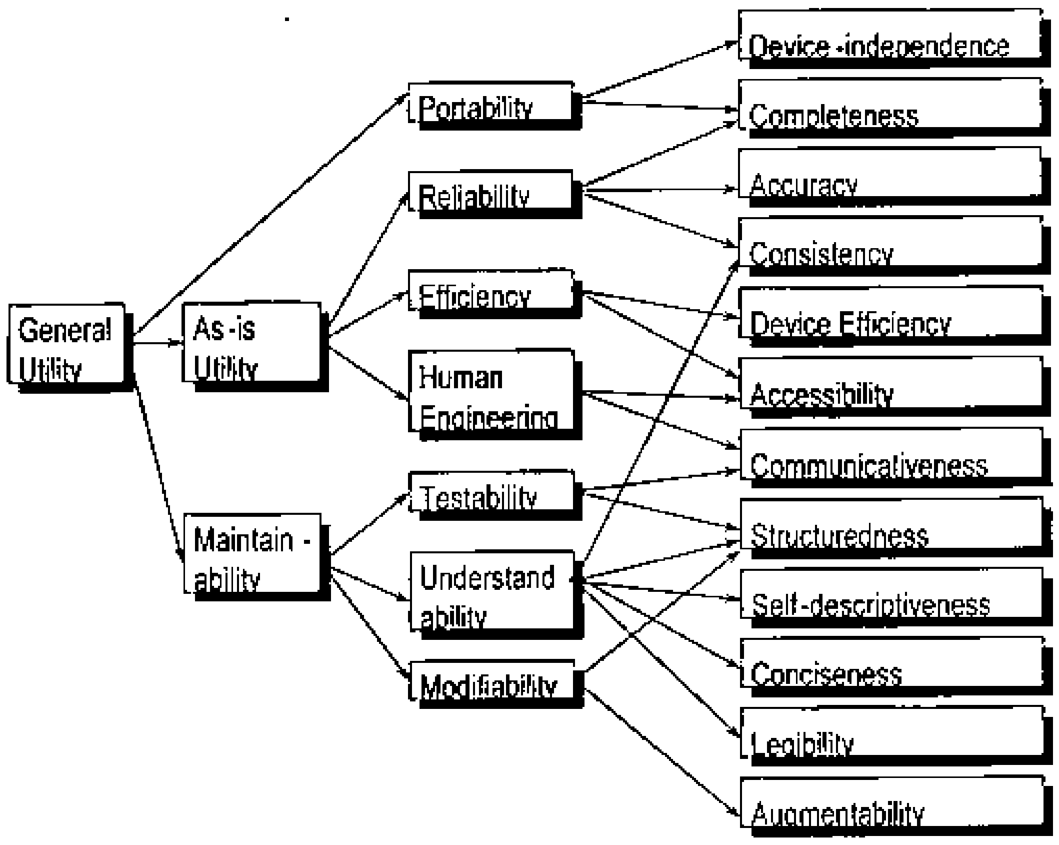

Figure 2.3 Boehm's Model

2-4-2-1-4 FURPS model

A later, and perhaps somewhat less renown model that is structured in basically the same manner as the previous two quality models (but still worth at least to be mentioned in this context) is the FURPS model originally presented by Robert Grady [7]. FURPS stands for: **Functionality** – which may include feature sets, capabilities and security. **Usability** - which may include human factors. **Reliability** - which may include frequency and severity of failure, recoverability, predictability, accuracy, and mean time between failure (MTBF). **Performance** - imposes conditions on functional requirements such as speed, efficiency, availability, accuracy, throughput, response time, recovery time, and resource usage. **Supportability** - which may include testability, extensibility, adaptability, maintainability, compatibility, configurability, serviceability, installability, localizability (internationalization).

The FURPS-categories are of two different types: Functional (F) and Non-functional (URPS). These categories can be used as both product requirements as well as in the assessment of product quality [7].

2-4-2-1-5 Dromey's Quality Model

An even more recent model similar to the McCall's, Boehm's and the FURPS quality model, is the quality model presented by R. Geoff Dromey [5,6]. Dromey proposes a product based quality model that recognizes that quality evaluation differs for each product and that a more dynamic idea for modeling the process is needed to be wide enough to apply for different systems. Dromey is focusing on the relationship between the quality attributes and the sub-attributes, as well as attempting to connect software product properties with software quality attributes[6].

2-4-2-2 Non-hierarchical Models

Another way of laying out the quality model is the non-hierarchical model and we will present two types as examples:

2-4-2-2-1 Bayesian Belief Networks

A BBN2 is a graphical networks whose nodes are the uncertain variables and whose edges are the causal links between the variables, Associated with each node is a set of conditional probability functions that model the uncertain relationship between the node and its parents.

It can be explained in two stages, one stage covers the life-cycle processes of specification, design or coding and the second stage covers testing.

Using the BBN have some benefits as follow :

BBNs enable reasoning under uncertainty and combine the advantages of an intuitive visual representation with a sound mathematical basis in Bayesian probability[14].

With BBNs, it is possible to articulate expert beliefs about the dependencies between various variables and to propagate consistently the impact of evidence on the probabilities of uncertain outcomes, such as future system reliability.

BBNs allow an injection of scientific rigour when the probability distributions associated with individual nodes are simply expert opinions.

A BBN will derive all the implications of the beliefs that are input to it; some of these will be facts that can be checked against the project observations, or simply against the experience of the decision makers themselves.

The ability to represent and manipulate complex models that might never be implemented using conventional methods[14].

2-4-2-2-2 STAR Model

The star model is introduced as follows: The software quality Star is a conceptual model for presenting different perspectives of software quality. The model is based on the acquirer and supplier as defined in ISO/IEC 12207 (1995).

There are three significant elements in the STAR: The procurer (acquirer), the producer (supplier), and the product. The procurer enters in a contract with the producer to create a software product. This contract clearly specifies the quality characteristics of the product. The procurer's perspective of the producer organization is that they use the best project management techniques available and that they engage in first-rate processes to create a quality product. The procurer's perspective of the product is that it must be acceptable by the user community and that it can be serviced and maintained by their professionals [17].

The model considers that the acquirer be the lead party in any contractual arrangement because it is the acquirer's users and technical support professionals who dictate the success or failure of the software product. Also, it is the acquirer who dictates the profile and maturity of the supplier organization.

The model accommodates the producers perspective of software quality and focuses on the maturity of the producer organization as software developers and the development processes that they used to create quality software products [17].

2-4-3 Classification of quality models according to layers

1:N relationship, as in ISO 9126 - every characteristic has its own set of sub-characteristics.

N:M relationship, as in McCall Factor-Criteria Model (FCM)- every sub-characteristic is linked to one or more characteristics [12].

Table 2-1 maps the different terms used in these various software quality models [12]:

LAYER	MC CALL	BOEHM	ISO9126:1991	IEEE 1061	DROMEY
1	FACTOR	high-level charac.	characteristics	Factor	hi-level attribute
2	CRITERIA	primitive-level	sub-characteristics	Sub-factor	subordinate level
3	METRIC	metric	metrics	Metrics	--------------

Table 2-1- Software Quality Models

However it is argued that with these models: It is not sufficient to list some characteristics that imply or contribute to a better quality in building software , but it is necessary to create direct links between quality attributes and product characteristics; There is a little assistance in building quality into software.

2-5 Quality Metric

We need to specify quality metrics to evaluate a given quality criteria. Each quality metric provides a numerical value that can be scaled to measure a quality factor.

Metrics must be completed and detailed sufficiently to be the firm foundation of a quality model.

There is a strange relationship between internal and external quality. External quality is quality as measured by the customer. Internal quality is quality as measured by the programmers [4, 13].

2-6 Software Metrics

Software metrics are used to quantify software, software development resources, and software development processes [20]. Some software characteristics are measurable directly (as LOC-Lines of Code), some other software characteristics can be inferred only from indirect measurements (for example, maintainability), and some software characteristics are mostly related to human perception (for Example, understandability is more dependent to the people vs. the programs).

Software metrics can be classified into three categories:

2-6-1 Product metrics: Describe the characteristics of the product, such as size, Complexity, design features, performance, and quality level.

2-6-2 Process metrics: Can be used to improve software development and maintenance process (i.e., effectiveness of defect removal during development, defect arrival, response time to fix defect).

2-6-3 Project metrics: Describe characteristics of the project and its execution (i.e., number of software developers, stating pattern related to the life cycle of the software, cost, schedule, and productivity).

However, some metrics belong to multiple categories (i.e., the in-process quality metrics of a project are both process metrics and project metrics). More-over, a healthy metrics program focuses on much more than the measurement of programmer productivity. Consider these areas of software development which can benefit from a well-planned metrics program [19] :

- Project management.
- Product quality.
- Product performance.
- Development process.

2-7 The problem of measuring quality

"When you can measure what you are speaking about, and express it in numbers, you know something about it; but when you cannot measure it, when you cannot express it in numbers, your knowledge is of a meager and unsatisfactory kind". (Lord Kelvin).Some kinds of metrics is needed to prove that the quality of the product is good or bad. Quality is difficult to measure, but somewhere some kind of measurement should be possible. Defining the attributes of software quality and determining the metrics to assess the relative value of each attribute are not formalized processes. Because users place different values on each attribute depending on the products use, it is important that quality attributes should be observable to consumers. However, with software there exists not only asymmetric information problems (where a developer has more information about quality than the consumer), but also instances where the developer truly does not know the quality of his own product. Although a general set of standards has been agreed on, the appropriate metrics to test how well software meets those standards are still poorly defined. Publications by IEEE (1988,1996) have presented numerous potential metrics that can be used to test each attribute.

2-8 Relationships among quality characteristics

Some quality characteristics are related to one another, we summarize these interrelationships between quality characteristics of McCall's model as shown in figure 2.4.

Interrelationships between quality characteristics - after Perry using McCall

	C	R	E	I	U	M	T	F	P	R	I
Correctness											
Reliability	_										
Efficiency											
Integrity											
Usability	_	_	*	_							
Maintainability	_	_	*		_						
Testability	_	_	*		_	_					
Flexibility	_	_	*		_	_	_				
Portability			*			_	_				
Re-usability		*	*	*		_	_	_	_		
Interoperability			*	*					_		

* = Inverse blank = Neutral _ = Direct

Figure 2-4 relationship between McCall's characteristics

Chapter Three

Survey on Libyan software systems

Everyone agrees that software quality is the most important element in software development, because high software quality could reduce the cost of maintenance, test and software reusing [3]. But quality has very different meanings for managers, users, developers, therefore many institutes and organizations have to improve software quality.

In this chapter we discuss Customer Satisfaction Surveys and the methodology to collect and analysis data.

3-1 Customer satisfaction surveys

There are various methods to obtain customer feedback with regard to their satisfaction levels with the product(s) and the company [19].

3-1-1 Methods with a moderate range of customer:

There are three common methods to gather survey data [19] : face-to-face interviews, telephone interviews, and mailed questionnaires (self-administered).

The personal interview method requires the interviewer to ask questions based on a pre-structured questionnaire and to record the answers. The primary advantage of this method is the high degree of validity of the data. Specifically, the interviewer can note specific reactions and eliminate misunderstandings about the questions being asked. The major limitations are costs and factors concerning the interviewer. If not adequately trained, the interviewer may deviate from the required protocol, thus introducing biases into the data. If the interviewer cannot maintain neutrality, any statement, movement, or even facial expression by the interviewer could affect the response. Errors in recording the responses could also lead to erroneous results.

Telephone interviews are less expensive than face-to-face interviews. Different from personal interviews, telephone interviews can be monitored by the research team to ensure that the specified interview procedure is followed. The computer-aided approach can further reduce costs and increase efficiency. Telephone interviews should be kept short and impersonal to maintain the interest of the respondent. The limitations of this method are the lack of direct observation, the lack of using exhibits for explanation, and the limited group of potential respondents-those who can be reached by telephone. The mailed questionnaire method does not require interviewers and is therefore less expensive. However, this savings is usually at the expense of response rates. Low response rates can introduce biases to the data because if the respondents are different from the non-respondents, the sample will not be representative of the population. Non-response can be a problem in any method of surveys, but the mailed questionnaire method usually has the lowest rate of response. For this method, extreme caution should be used when analyzing data and generalizing the results. Moreover, the questionnaire must be carefully constructed, validated, and pre-tested before final use. Questionnaire development requires professional knowledge and experience [19].

* Notice

We have adopted the method of ***Luigi Buglione&Alain Abran***[12]: a modified method of data collection might be close to that of the mailed questionnaire except we deliver the pre-structured questionnaire personally. This should demonstrate to the institutions concerned that we are really interested in all parties benefit and that should reflect in a full cooperation and sincere answers to our questions. However, there are problems with values derived in McCall's model, the degree of subjectivity varies substantially from one question to another, even though all responses are treated equally. This variation makes difficulty in assessing the characteristics. Moreover, when appropriate response should be reflected in a richer measurement scale, it is not reasonable to expect a yes-or-no response to the questions. So, we have noticed that this method do not give accurate results that is why we have adopted the ordinal scale to reflect variety of possible answers.

3-1-2 Sampling methods

The three methods, mentioned above, are sufficient if the customers' base is not large; but it would be very costly in case of a large base of customers. Estimating the satisfaction level of the entire customer population through a representative sample is more efficient. To obtain representative samples, scientists' probably sampling method must be used. There are four basic types of probability sampling[19]:

- Simple random sampling
- Systematic sampling
- stratified sampling
- cluster sampling.

3-1-2-1 Simple random sampling: If a sample of size (n)is drawn from a population in such a way that every possible sample of size n has the same chance of being selected; the sampling procedure is called simple random sampling. The sample, thus obtained, is called simple random sample. Sometimes the method is mistaken since the investigator just randomly select individuals that he or she happens to come across.

3-1-2-2 Systematic sampling: is simpler that random sampling if a list is extremely large or long sample to be drawn. There are two types of situations in which systematic sampling may introduce biases:

(1) The entries on the list may have been ordered so that a trend occur.

(2) The list may posses some cyclical characteristics means may suffer of duplication.

3-1-2-3 Stratified sample: we first classify individuals into non-overlapping groups, called **(strata)**, then select simple random sample from each stratum. The strata are based on important variables pertaining to the parameter of interest. For example, customers with complex network system may have a set of satisfaction criteria for software product that is very different from those who have standalone system and simple applications. Therefore, stratified sample should include customer type as a stratification variable.

Notice:**(This is the type that we have chosen as a method of our survey)**

3-1-2-4 Cluster sampling is a simple random sampling in which schools, districts or work plants are used units for clustering sample. These geographical units are used for clustering sampling. It is usually less efficient than simple random sampling [19].

3-2 Quality model as a base of evaluating quality:

The benefit of Quality models is given by the decomposition of the valuable object (both a process, a product or an organization) in a list of its characteristics, sub-characteristics and measures and it is applicable both to predict/assure/verify the achievement of a defined goal about the object before/while/after producing it[12].

For clarity of definition, we use in this study the McCall's model characteristics .

The problem of a common acceptance of more relevant characteristics to consider or to reject is a never-ending story in every community and also in the IT one. Therefore the value of standards is to group as many people as possible at the aim to use and share definitions, concepts and methods.

3-3 Quality as a survey base

3-3-1 The procedure flow

This section presents the design of the procedure to calculate quality, which returns a numerical value (rephrase) integrating the users (U), developers (D) and managers (M) opinions about the quality of the software being measured. As mentioned above, it is important to a company to do regular quality assessments of its products in order to manage them properly. Management must therefore obtain a value to decide based on the opinions of the three major interest groups (U, D, M), which actions must be taken for the future [12]. Figure 3.1 shows the high-level procedure flow.

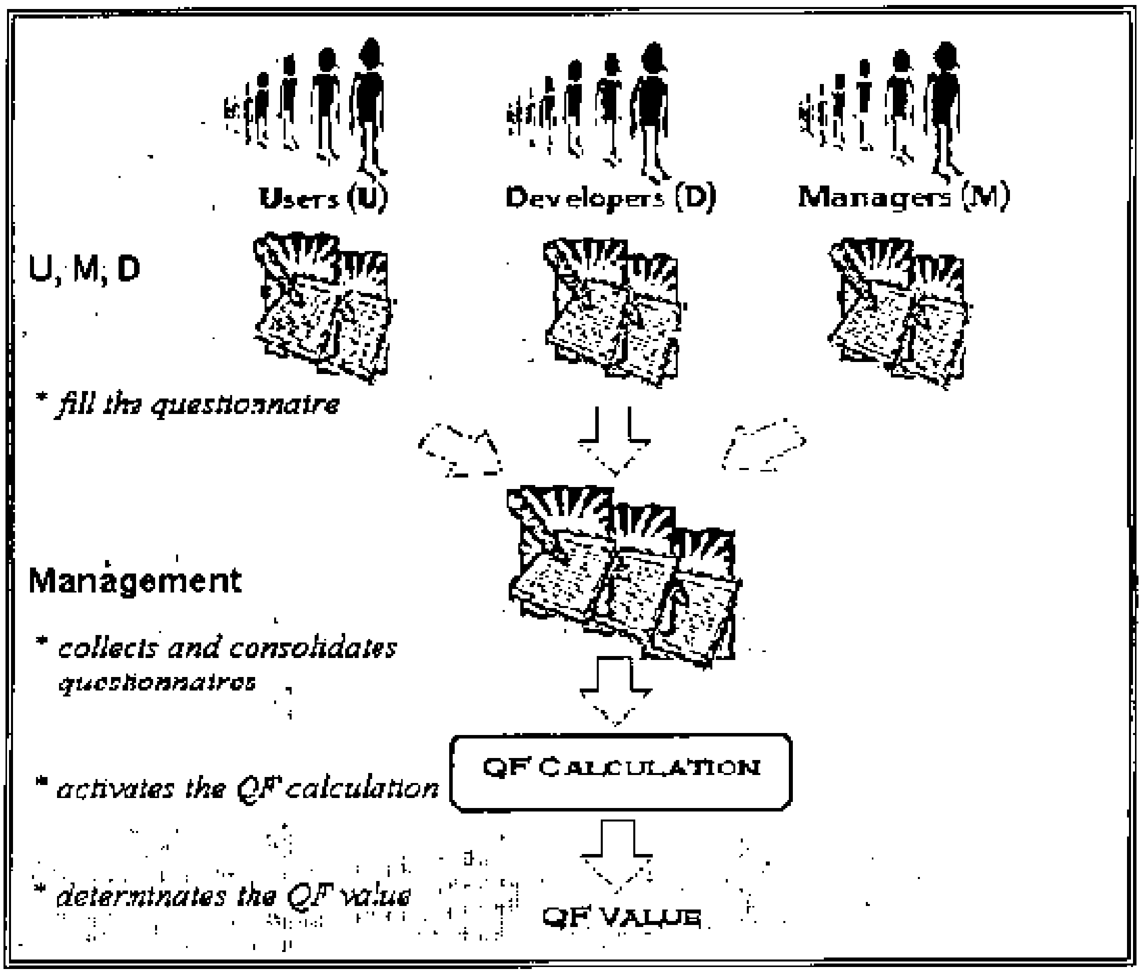
Users (U)
Developers (D)
Managers (M)
U, M, D
* fill the questionnaire
Management
* collects and consolidates questionnaires
* activates the QF calculation
QF CALCULATION
* determinates the QF value
QF VALUE

Figure 3.1 quality factors procedure flow

First, a questionnaire is submitted to users, developers and managers, who express their quality opinions. Then, the questionnaires are gathered and consolidated these opinions and, through the quality calculation procedure, the final quality value is obtained for the software program being assessed. The procedure is fairly simple and easy to automate with a spreadsheet [12].

3-3-2 The Quality factor calculation

We design the procedure to calculate quality, which returns a value which integrated the users , developers and managers opinions about the software product being measured. The measurement instruments needed to calculate quality are: The questionnaire is the technical instrument we use to collect data. Using the samples analysis through the stratified sample technique with Universe sensible elements divided in non-Overlapping groups (**strati**) and then choosing a random sample in every stratus we can have a good sample representatively and reliability of the values found. The questionnaire is structured following Statistics and Social Research Methodology principles, handed over to the interviewed person and filled up by him/her without any help from an interviewer [2,4,12].

Here is a sample of the rank table(table 3.2) used in recording the grades:

MARK	RATING	GLOBAL RATING
3	**Excellent**	**Satisfactory**
2	**V.Good**	
1	**Good**	
0	Poor/ Absent	Unsatisfactory

Table 3.2 rank table

For processing our results, we have chosen the (SPSS) program using the Cronbach's coefficient that gives high percent in the reliability on its results.

3-3-3 Different points of view in software characteristics

3-3-3-1 User's view

Users are mainly interested in using the software, its performance and the effects of using the software. Users evaluate the software without knowing the internal aspects of the software, or how the software is developed [2].

3-3-3-2 Developer's view

Since developers are responsible for producing software which will satisfy quality requirements they are interested in the intermediate product quality as well as in the 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 [2].

3-3-3-3 Manager's view

A manager is typically more interested in the overall quality rather than in a specific quality characteristic. 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].

3-4 Tools of our survey and analysis

As an essential step in our study on quality and the efficiency of software programs , it is logical that we detect their effect in our local Organizations through a questionnaire and personal interview.

We have mentioned previously that we have adopted the method of ***Luigi Buglione&Alain Abran*** [11,12], a pre-structured questionnaire to be filled by the three categories involved in software applications.

The results will be analyzed and evaluated by a selected software : SPSS.

3-4-1 The analysis program (SPSS):

SPSS is a software package used for conducting statistical analyses, manipulating data, and generating tables and graphs that summarize data. Statistical analyses range from basic descriptive statistics, such as frequencies and averages, to advanced inferential statistics, such as regression models, analysis of variance, and factor analysis. (SPSS) also contains several tools for manipulating data, including functions for recoding data and computing new variables, as well as for merging and aggregating datasets. SPSS also has a number of ways to summarize and display data in the form of tables and graphs.

3-4-2 Cronbach's Coefficient (Alpha)

This coefficient is used to determine the internal coordination of the measurements through recognizing the Internal Consistency Reliability (**ICR**).

That test uses the answers collected of all given questions to evaluate the degree of coordination and to evaluate the extent in which every question measures the same conception.

Chapter Four

Results and Comments

We will present the coefficient that we have chosen in our survey, Cronbach's Coefficient (Alpha) that gives a reliable results in measuring. Then we will present the form of our questionnaire followed by the results of our survey supported by the charts under each a comment of the result gained.

4-1 Cronbach's Coefficient (Alpha)

This coefficient is used to determine the internal coordination of the measurements through recognizing the Internal Consistency Reliability (ICR).

That test uses the answers collected of all given questions to evaluate the degree of coordination and to evaluate the extent in which every question measures the same conception.

These questions are related to each other and the most common test to measure the (ICR) is what we call Cronbach coefficient where the value of stability, applied on a primary specimen (no of cases =25,no of items=38), reaches 93.4% and that indicates the accuracy of the test and the stability of its results that it is almost infallible.

In other words, the possibility of dependency on the measurements indicates the stability and the unity in the scale make it reliable to use in our questionnaire.

4-2 Questionnaire form as a sample of the record

Here enclosed a copy of the questionnaire form as a sample of the record of quality evaluation:

جامعة التحدي
كلية العلوم
قسم الحاسوب

يهدف هذا الاستبيان إلى جمع بيانات لإتمام رسالة ماجستير حول دراسة جودة البرمجيات المحلية

نرجو الإجابة على هذه الأسئلة

***اسم المنظومة** --------------

***بيانات شخصية**

1- العمر ☐ 18-25 ☐ 26-35 ☐ 36-45 ☐ اكبر من 45

2- المستوي العلمي ☐ دون الثانوي ☐ ثانوي ☐ جامعي ☐ فوق الجامعي

3- عدد سنوات العمل بالمؤسسة ☐ أقل من سنة ☐ من 1- 5 ☐ أكثر من 5 سنوات

4- عدد سنوات العمل على المنظومة ☐ أقل من سنة ☐ من 1- 5 ☐ أكثر من 5 سنوات

5- عدد سنوات الخبرة في مجال الحاسب الآلي ☐ أقل من سنة ☐ من 1- 5 ☐ أكثر من 5سنوات

6-العمل في المؤسسة ☐ مستخدم **USER** ☐ مطور**DEVELOPER** ☐ مدير **MANEGER**

*** بيانات حول جهاز الحاسوب المستعمل**

1- سرعة الحاسب ----------

2- نظام التشغيل -----------

3- هل تستخدم الفأرة (الماوس) في المنظومة ☐ نعم ☐ لا

4- هل تستخدم لوحة المفاتيح في المنظومة ☐ نعم ☐ لا

• ضع علامة☑ لكل من الخصائص الفرعية التي في رايك ذات علاقة أكثر لجودة النظام لديك

الخاصية الرئيسية	الخاصية الفرعية	لا اعرف	ضعيف	جيد	جيد جدا	ممتاز
سهولة الاستخدام Usability	سهولة تشغيل النظام Operability					
	تعلم النظام سهل Training					
	سهولة التخاطب مع شاشات النظام Communicativeness					
	النظام يستوعب كل المدخلات والمخرجات I/O volume					
	سرعة معالجة البيانات المدخلة وتفسير المخرجات I/O rate					
التكامل (سلامة النظام) Integrity	التحكم في الوصول للبرامج والبيانات Access Controlمثلا هل توجد كلمة سرPassword					
	تدقيق وفحص الوصول Access Audit هل يتم تسجيل معلومات عن كل الأشخاص الذين دخلوا للنظام					
الكفاءة أو الأداء Efficiency	استغلال جيد لوسائط التخزين Storage Efficiency					
	سرعة تنفيذ النظام Execution Efficiency					
صحة النظام Correctness مدى ملائمة النظام للمقاييس ومتطلبات الزبون	سهولة تتبع النظام للتأكد من تلبية النظام لمتطلبات الزبون Traceability					
	النظام أنجز كل ماهو مطلوب منه ويعالج كل المدخلات المحتملة Completeness					
	استخدام تصميم وتوثيق موحد خلال تصميم وتطوير البرنامج Consistency					
النظام يقوم بكل المهام وبالدقة المطلوبة Reliability	دقة النظام في الحسابات والتحكم في عمليات Accuracy					
	النظام يسمح بتجاوز الأخطاء التي لا تؤثر في النظام Error tolerance					
	النظام يمنع حدوث أخطاء قد تؤدي إلي فشل النظام Consistency					
	النظام موثوق فيه ويمكن فهمه ببساطة Simplicity					
الجهد المطلوب لتحديد وتصحيح الخطأ في النظام Maintainability	عمليات الصيانة تتم وفق إجراءات محددة وموثقة consistency					
	سهولة تحديد أخطاء النظام وإصلاحها Simplicity					
	النظام مختصر وموجز ولذلك صيانة النظام تتم بسهولة conciseness					
	خطوات برنامج المصدر-الكود تقدم وصفا متكاملا عن عمل النظام Self descriptiveness					
	مكونات النظام مستقلة ولا يوجد اعتماد أي مكون على الأخر فتصحيح الأخطاء تتم على اي جزء بدون أن تتأثر بقية أجزاء النظام modularity					

الخاصية الرئيسية	الخاصية الفرعية	لا أعرف	ضعيف	جيد	جيد جدا	ممتاز
اختبار النظام Testability	بساطة وسهولة اختبار النظام Simplicity					
	استخدام طرق أو أدوات برمجية لاختبار عمليات النظام وتحديد الأخطاء Instrumentation					
الجهد المطلوب لتعديل النظام Flexibility	النظام قابل للتطوير Expandability					
	النظام عام ويمكن التعديل فيه بسهولة ليقوم بوظائف مشابهة Generality					
	النظام يعطي وصفا متكاملا عن كل العمليات التي يقوم بها Self descriptiveness					
	مكونات النظام مستقلة عن بعضها البعض فتعديل أي جزء لا يؤثر على عمل بقية أجزاء النظام modularity					
قابلية إعادة الاستخدام Reusability	النظام عام ويمكن استخدامه أو استخدام جزء منه في نظام أخر Generality					
	أجزاء النظام موصوفة ذاتيا Self descriptiveness					
	النظام مستقل عن الآلة ويمكن إعادة استخدام أي جزء من البرنامج على جهاز حاسوب مختلف Machine Independent					
	النظام مستقل عن البرمجيات ويمكن إعادة استخدام أي جزء من البرنامج على نظام تشغيل مختلف S/W Independent					
	مكونات النظام مستقلة وظيفيا عن بعضها البعض ونقل أي جزء من النظام إلى النظام نفسه أو إلى نظام أخر يتم بسهولة modularity					
قابلية نقل النظام Portability	النظام يصف نفسه ذاتيا ولا يحتاج إلى مجهود عند نقله إلى بيئة أخرى Self descriptiveness					
	النظام مستقل عن الآلة ويمكن استخدام النظام على جهاز حاسوب مختلف Machine Independent					
	النظام مستقلا عن البرمجيات ويمكن استخدام النظام على نظام تشغيل مختلف Software Independent					
	النظام قابل للنقل ويتكيف مع البيئة الجديدة modularity					
قابلية الاتصال مع النظم الأخرى Interoperability	النظام يتصل مع النظم الأخرى بواسطة اتصالات ووجهات شائعة الاستخدام Communications Commonality					
	النظام يستخدم نفس هيكلية البيانات ونفس النوع مع النظم الأخرى التي يتبادل معها البيانات Data Commonality					

4-3 Results of local software quality characteristics

In this section we demonstrate the obtained results of our survey using McCall's characteristics (0=poor , 1=good , 2=very good , 3=excellent):

4-3-1 Usability

Grade	Subcharacteristic
1.84	Operability
1.73	Training
1.73	Communicativeness
1.61	I/O volume
1.63	I/O rate

Table 4.1 Sub characteristics of usability

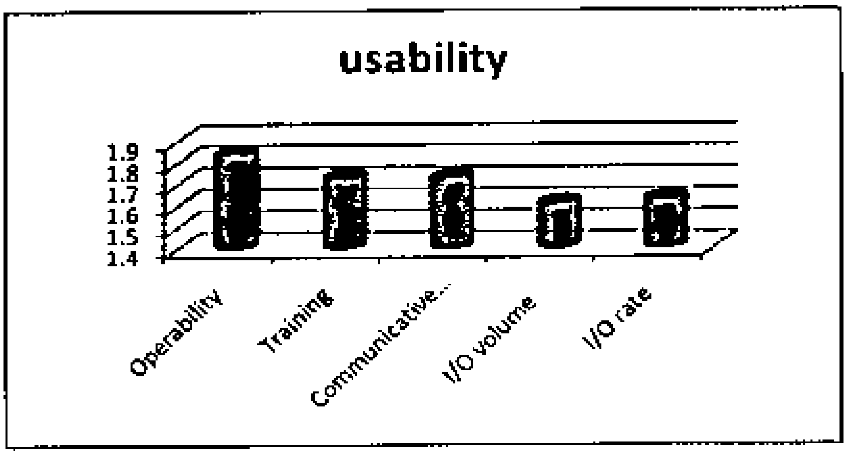

Figure 4.1 Sub characteristics of usability

From figure 4.1, we notice that the usability is not on the required level ,because some of the institutes in Libya such as Banks and the Arabian Gulf Oil Company, use DOS operating systems which lacks the user interface.

4-3-2-Integrity

Grade	Subcharactristic
1.77	Access Control
1.49	Access Audit

Table 4.2 Sub characteristics of integrity

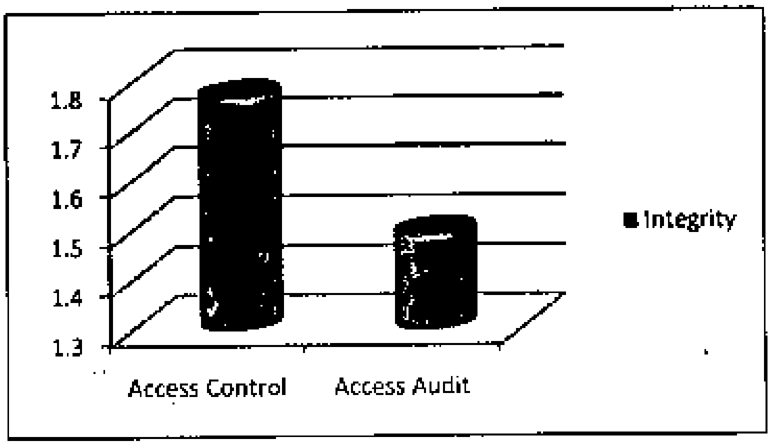

Figure 4.2 Sub characteristics of integrity

From figure 4.2, we notice that the integrity is not on the required level ,because some developers don't pay attention to protect their systems by using weak databases instead of the most protected databases like (ORACAL and SQL SERVER).

4-3-3 Efficiency

Grade	Subcharactristic
1.37	Efficiency Storage
1.44	Execution Efficiency

Table 4.3 Sub characteristics of Efficiency

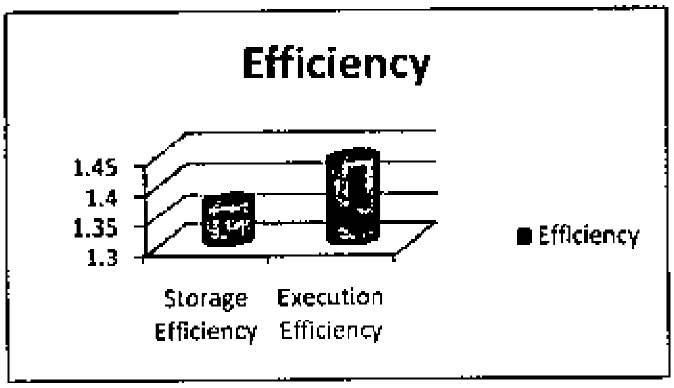

Figure 4.3 Sub characteristics of Efficiency

From figure 4.3, we notice the efficiency is not on the required level ,because some developers don't use databases which automatically compresses the data to make more space (storage efficiency) .

4-3-4 Correctness

Grade	Subcharactristic
1.46	Traceability
1.45	Completeness
1.03	Consistency

Table 4.4 Sub characteristics of correctness

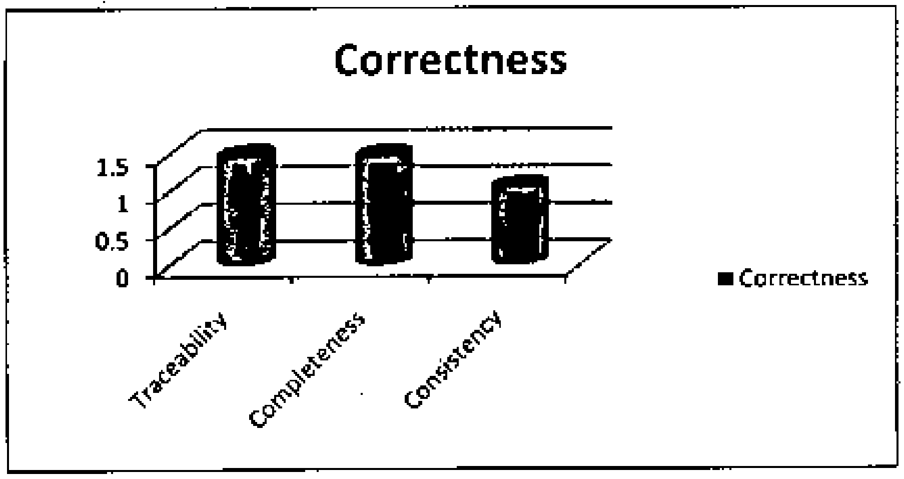

Figure 4.4 Sub characteristics of correctness

From figure 4.4, we notice that local software suffer from weakness in design and documentation, and local software is suffering of compatibility with demands of client, it means that some software is not compatible with the client requirement.

4-3-5 Reliability

Grade	Subcharactristic
1.44	Accuracy
0.87	Error tolerance
1.09	Consistency
1.61	Simplicity

Table 4.5 Sub characteristics of reliability

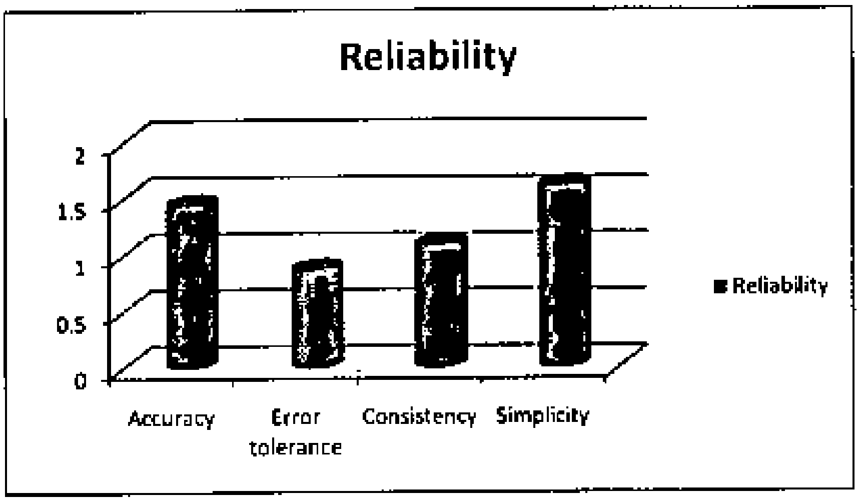

Figure 4.5 Sub characteristics of reliability

From figure 4.5, we notice that the most of the local software is not reliable due to continues changes of the management decisions and requirements , this effects the reliable of the local software .

4-3-6 Maintainability

Grade	Subcharactristic
1.32	Consistency
1.24	Simplicity
1.18	Conciseness
0.89	Self descriptiveness
1.01	Modularity

Table 4.6 Sub characteristics of maintainability

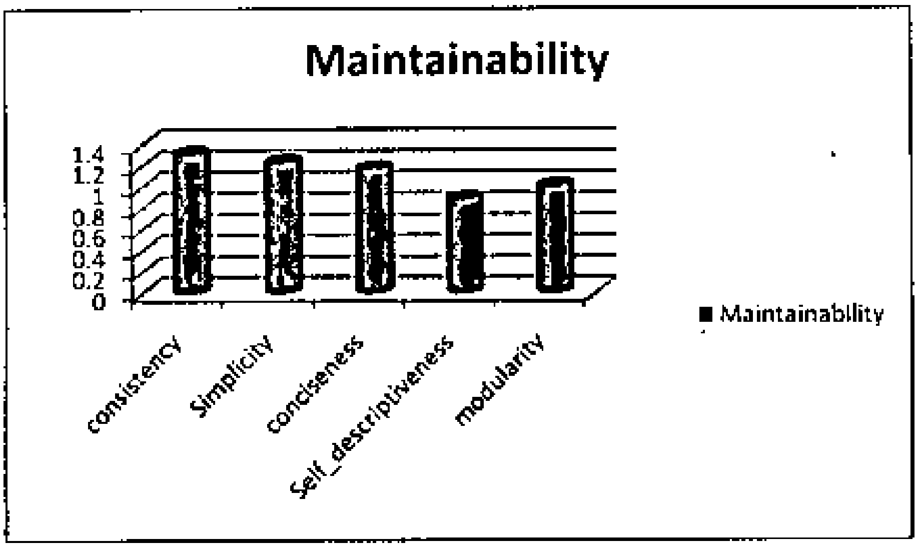

Figure 4.6 Sub characteristics of maintainability

From figure 4.6, we notice that local software need time and effort for maintenance and that means they are not well designed for the future maintenance.

4-3-7 Testability

Grade	Subcharactristic
1.25	Simplicity
0.93	Instrumentation

Table 4.7 Sub characteristics of reliability

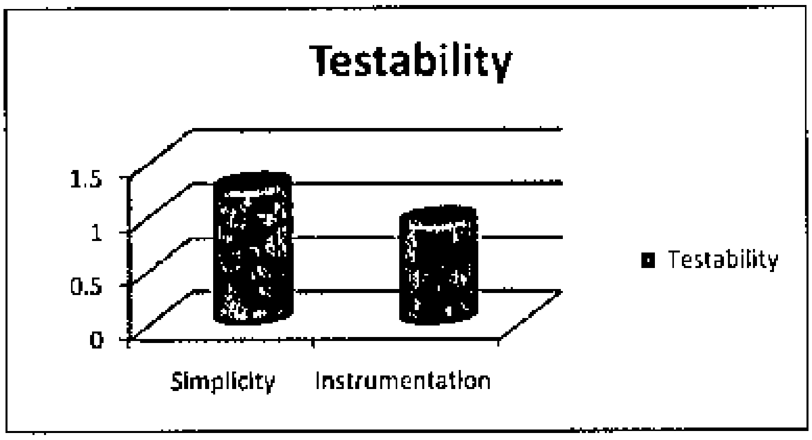

Figure 4.7 Sub characteristics of testability

From figure 4.7 , we notice that local software do not use programming tools for testing the software to determine errors , but it is simple and easy somehow in testing.

4-3-8 Flexibility

Grade	Subcharactristic
1.47	Expandability
1.17	Generality
1.15	Self descriptiveness
1.03	Modularity

Table 4.8 Sub characteristics of flexibility

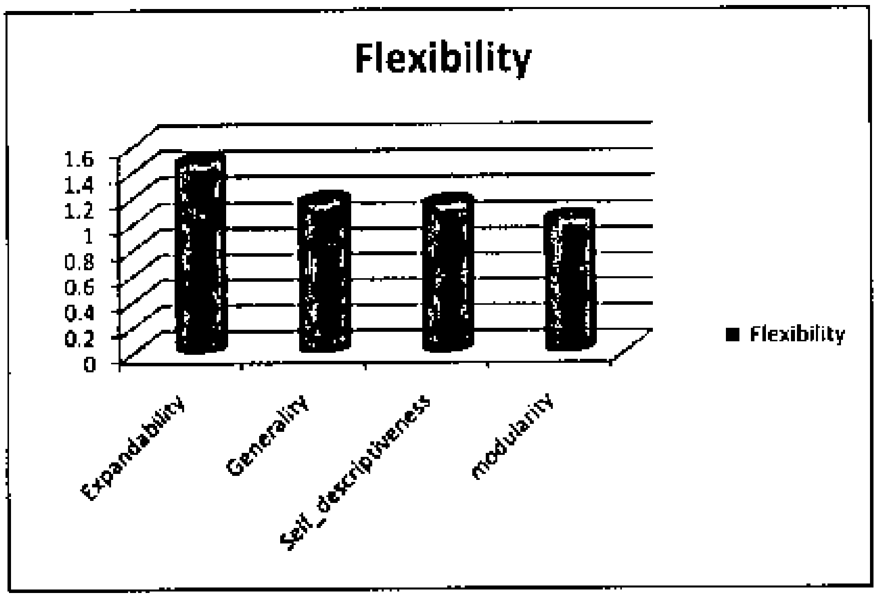

Figure 4.8 Sub characteristics of flexibility

From Figure 4.8, we notice flexibility was better in expendability while less in other sub-characteristics.

4-3-9 Reusability

Grade	Subcharactristic
0.81	Generality
0.99	Self descriptiveness
1.03	Independent Machine
0.79	S/W Independent
0.84	Modularity

Table 4.9 Sub characteristics of reusability

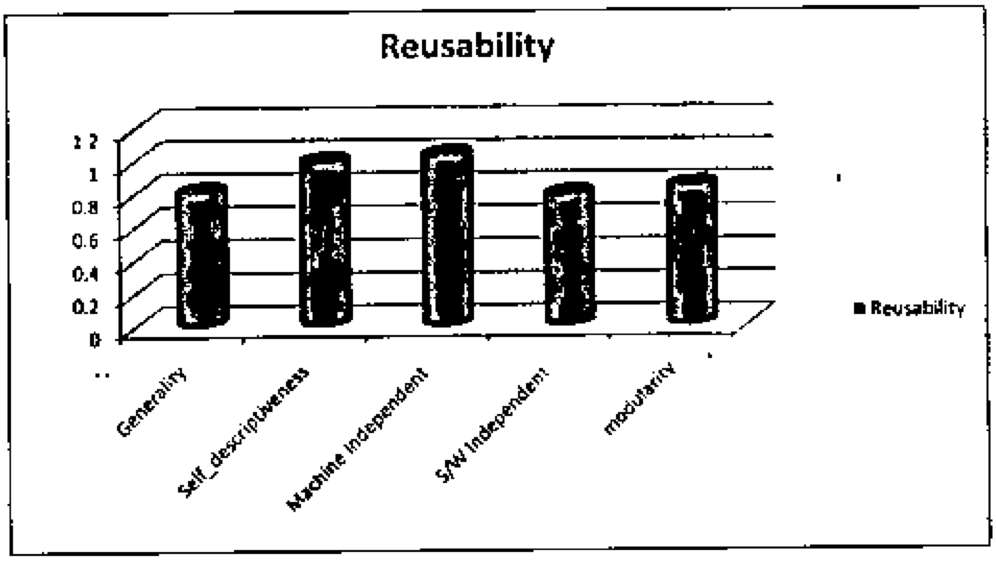

Figure 4.9 Sub characteristics of reusability

From figure 4.9, we notice that reusability is weak in all its sub-characteristics that imply the difficulty in reusing the system or part of the system in the system or another system because some developers don't use modern programming languages that support reusing code .

4-3-10 Portability

Grade	Subcharactristic
0.76	Self descriptiveness
1.01	Machine Independent
0.69	S/W Independent
0.81	modularity

Table 4.10 Sub characteristics of portability

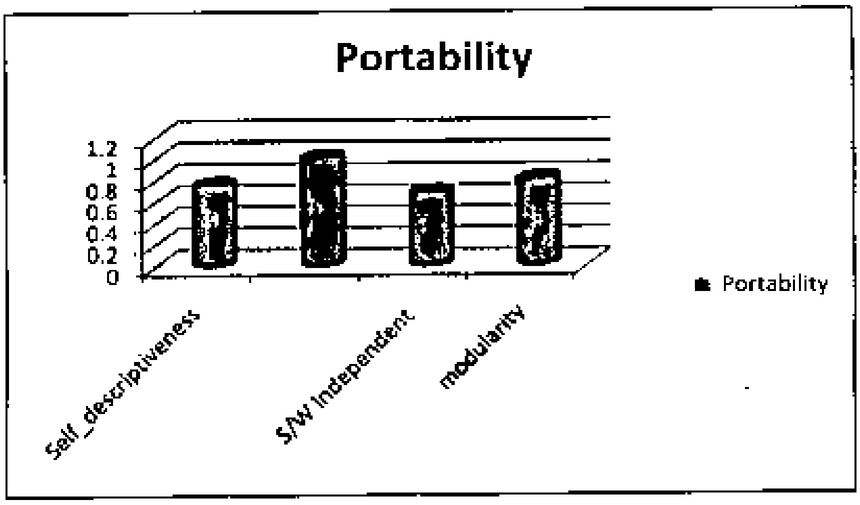

Figure 4.10 Sub characteristics of portability

From figure 4.10, generally, portability is the weak point in Libyan local software specially in software independent, that means difficulty in transferring the software to another environment communication between system and the other.

4-3-11 Interoperability

Grade	Subcharactristic
0.84	Communication Commonality
0.74	Data Commonality

Table 4.11 Sub characteristics of interoperability

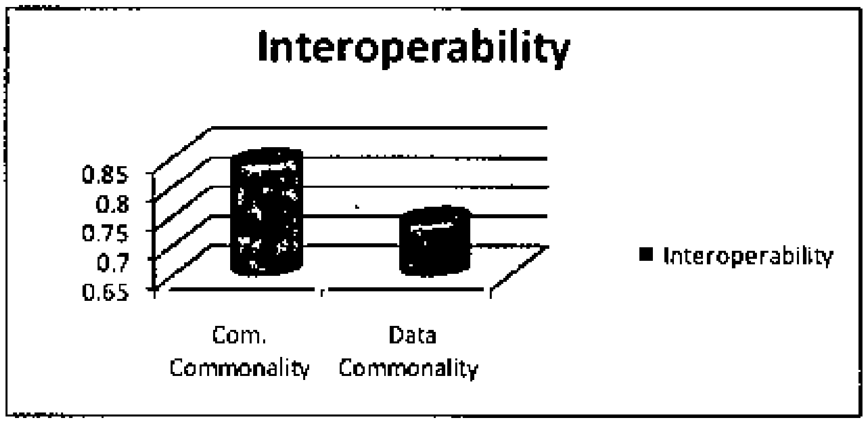

Figure 4.11 Sub characteristics of interoperability

From figure 4.11, we notice that interoperability is weak point specially in data communication between system and the other .

4-3-12 Software quality overall

CHARACTARISTIC	Grade
Usability	1.68
Integrity	1.61
Efficiency	1.48
Correctness	1.45
Reliability	1.44
Maintainability	1.26
Testability	1.23
Flexibility	1.32
Reusability	0.95
Portability	0.81
Interoperability	0.84
AVG	1.275168118

Table 4.12 software quality characteristics

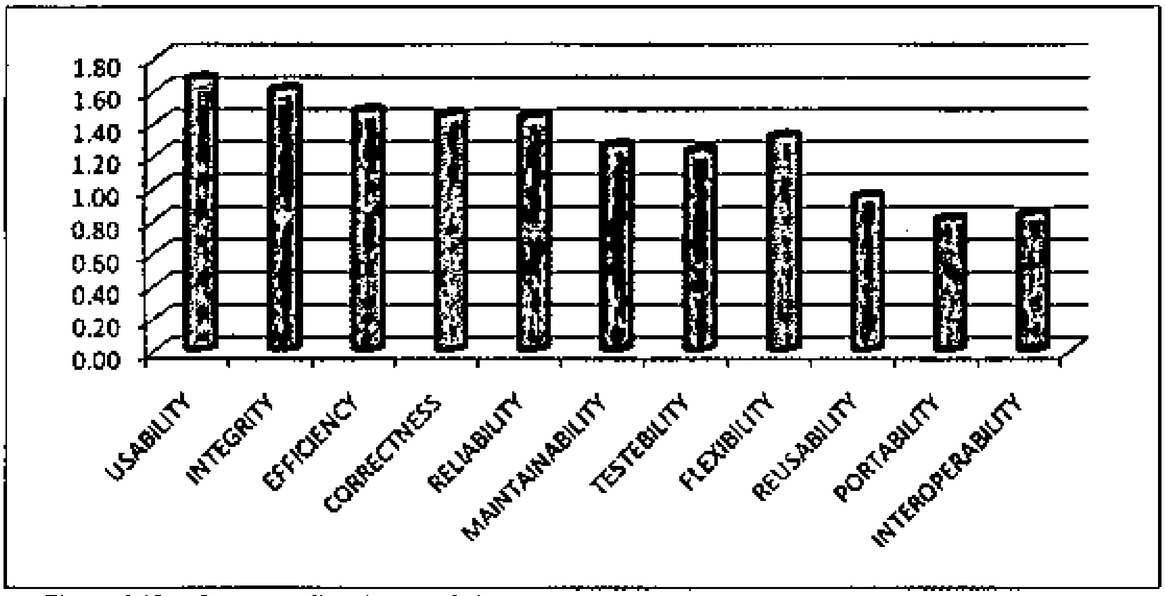

Figure 4.12 software quality characteristics

From figure 4.12 we notice that quality was not in the required level, no characteristic has reached the excellent or even the very good level. Most of characteristics were good except the last three factors which were in a poor level.

We now explain the different viewpoint of the users, developers and managers about local software quality.

CHARACTARISTIC	MANAGER	DEVELOPER	USER
Usability	1.71	1.60	1.73
Integrity	1.58	1.62	1.64
Efficiency	1.62	1.45	1.37
Correctness	1.79	1.32	1.25
Reliability	1.67	1.50	1.15
Maintainability	1.42	1.33	1.05
Testability	1.38	1.31	1.00
Flexibility	1.63	1.17	1.16
Reusability	0.98	1.02	0.85
Portability	0.75	0.87	0.82
Interoperability	0.85	0.90	0.76
AVG	1.394736842	1.27443609	1.156331423

Table 4.13 viewpoint software quality characteristics

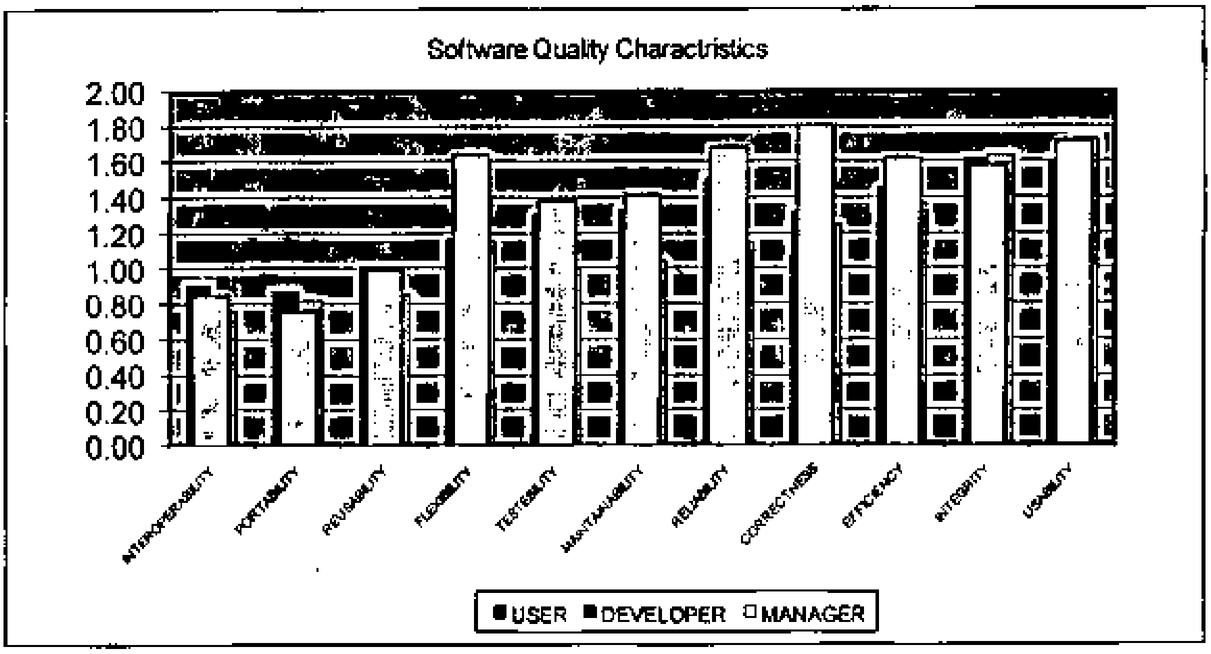

Figure 4.13 viewpoint software quality characteristics

Studying this chart, we notice that quality was not in the required level, no characteristic has reached the excellent or even the very good level. Most of characteristics were good except the last three factors which were in a poor level.

And also we noticed that developers were better with the characteristics: (reusability, portability, interoperability) because they concern them more than the users and the managers.

The opinions of the users, developers and managers were different, also the results were not the same for all characteristics. In our targeted organizations, the opinions of managers were better than those of the users and developers and that because of many reasons:

- The managers have more years of experience than the developers and the users on the software .
- Scientific degree of managers are more than users and developers.
- Years of experience in the field of computers are equal with the developers.

Generally , I support the point of view of managers because the final approval on the software is made by them. While users had weak answers because they are in experience than managers and developers.

4-4 Results of experience of users ,developers and managers

4-4-1 Years experiences on the computer

	More than 5 years	From 1 to 5 years	Less than 1 years
User	45.54	36.63	14.85
Developer	76.19	23.81	0.00
Manager	76.92	15.38	0.00

Table 4-14 Years experiences on the computer

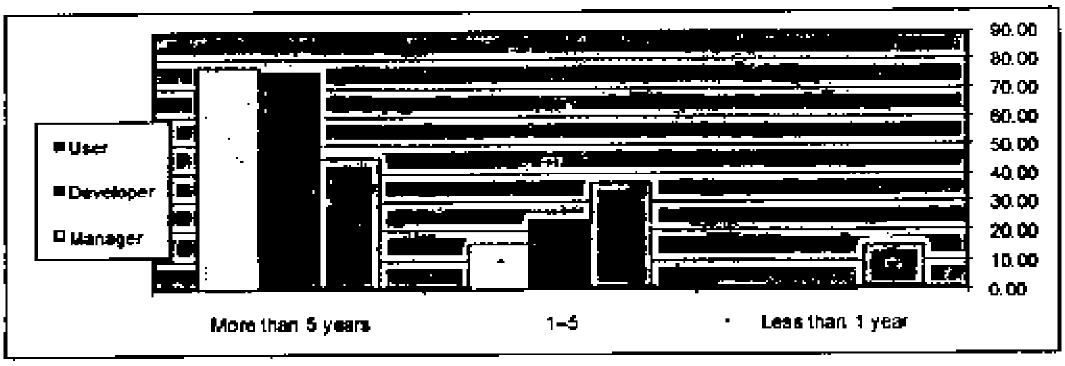

Figure 4-14 Years experiences on the computer

From figure 4.14 we notice that the developers and managers have more experience in the computer than users.

4-4-2 Years experiences on the system

	More than 5 years	From 1 to 5 years	Less than 1 year
User	30.69	32.67	33.66
Developer	42.86	33.33	19.05
Manager	69.23	30.77	0.00

Table 4-15 Years experiences on the system

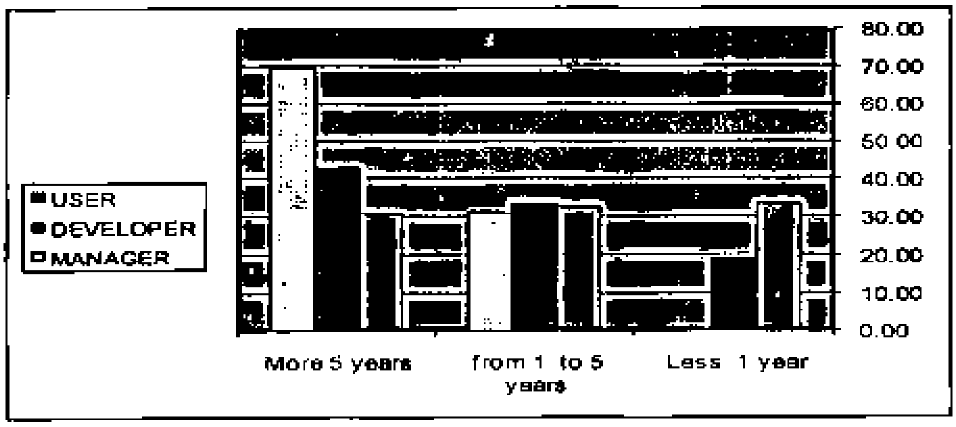

Figure 4-15 years experiences on the system

From figure 4.15 we notice that the managers has more experiences in the system than developers and users.

4-4-3 Years experience in the company

	More 5 years	From 1 to 5 years	Less 1 year
User	49.50	32.67	17.82
Developer	76.19	9.52	14.29
Manager	76.92	23.08	0.00

Table 4-16 years experiences on the company

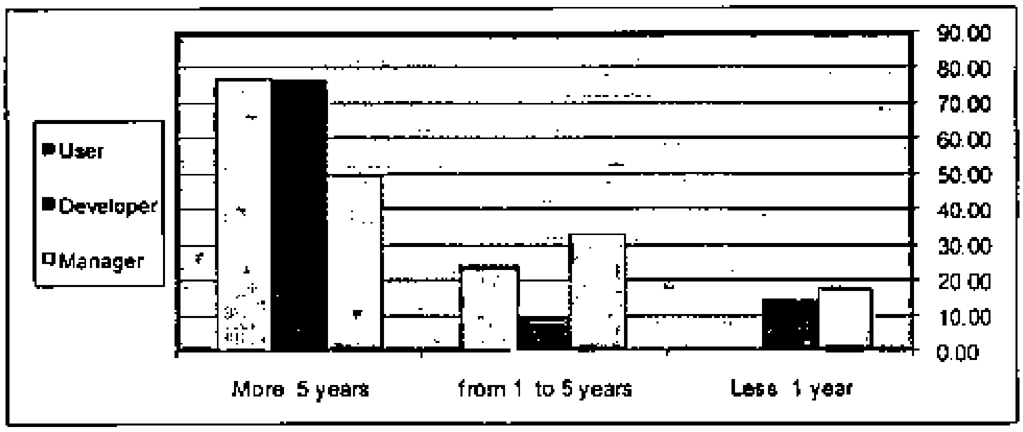

Figure 4-16 years experiences on the company

From figure 4.16 we notice that the developers and managers have more experience in the company than users.

4-4-4 Education qualification

	Post graduate	Graduate	High school	Junior high school
User	4.95	58.42	23.76	10.89
Developer	9.52	76.19	9.52	4.76
Manager	23.08	61.54	15.38	0.00

Table 4-17 education qualification

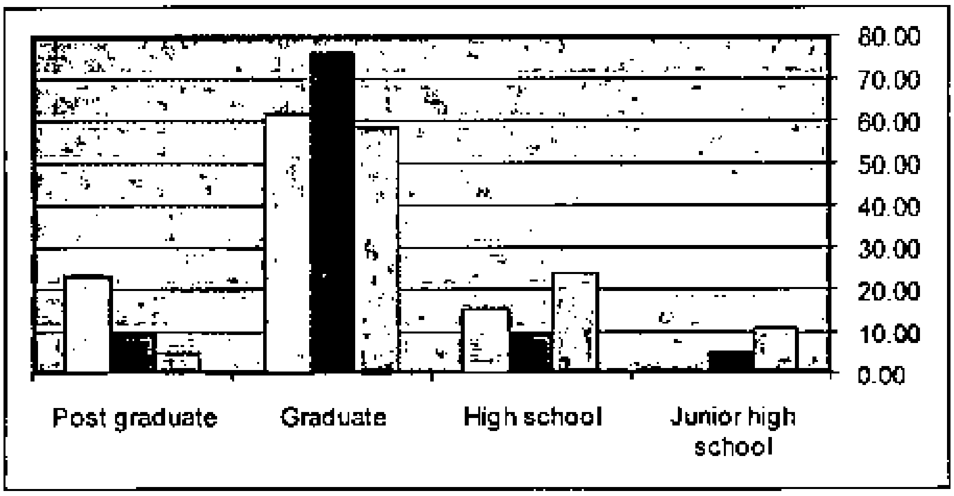

Figure 4-17 education qualification

From figure 4.17 we notice that the developers and managers have high level education more than users.

4-5 Results from personal interviews

Through our interviews ,Generally, according to our study, local software quality is average.

We have noticed that most of our local organizations have no documentation for the systems to be able to solve the problems of maintaining and developing.

Also, we have noticed that reusability had weak results because most developers use the structural programming instead of object oriented programming(OOP).

Object-oriented technology is one of the most widely used paradigms for developing software systems. Researchers assert that OO practice assures good quality software.

OOP helps more in maintaining and developing the system and that can save more time , effort and costs.

We have noticed that portability has weak results because some programmers use programming language that does not support this characteristic. This problem can be solved by using programming languages that support this characteristic (portable programming languages) like: Java that uses virtual machine or .Net languages that uses common language runtime (CLR).

We have noticed that interoperability has weak results and that can be solved by using modern technologies as Web service, that is a software system designed to support interoperable machine to machine interaction over a network. It has an interface described in a machine processable format (specifically WSDL). Other systems interact with the Web service in a manner prescribed by its description using SOAP messages, typically conveyed using HTTP with an XML serialization in conjunction with other Web-related standards [22].

Also we noticed that some developers are not aware of methodologies in software engineering.

Chapter Five

Conclusion and recommendation

5-1 Conclusion

Studying the quality of software and its evaluation.is not a subject that you can contain in one study. It deserves many studies to cover it with all its embranchment . we have tried to cover an essential part of it hoping to get to an important step in developing and improving the performance of the software programs in Libya. It's the survey on the relative importance of the software quality. When we say relative, we imply to the difference in quality characteristics order from one consumer to another. These relativity create the importance of the survey and its integrity in reflecting the suitability of the software to each consumer environment with all its variables and requirements. This survey is a mutual interest between all the parties involving in using, developing or producing the product. Investigating the quality of the software may be the first step in reaching high performance software since the application of the software in real life is the true judge of the product. Enhancing and generalizing the survey method for better obtaining data could be a major step in this process. It is more logical that the checklist made to reflect the quality and high performance requirements , should be made periodically to adopt with the change in the projects operation and the rapid development in the field of computer technology. The proposed methodology starts from the need to obtain an integrated software quality measurement, which answer the technical, economic and social needs that form the total structure of every organization operations, which however is not often presented in a unitary view. Starting from a questionnaire filled out individually, gathering the opinions of users, developers and managers about the quality perceived in a certain product, it is possible to obtain a single numerical value useful at the management level to take economical decisions for the future. Our checklist is based on the current McCall's model and the procedure is in line with upcoming revisions of this standard.

The environments used for the surveys are the Libyan organizations as the axis of our interest.

We have reached a conclusion, after applying our study and analyzing the data collected that Libyan local software not in the required level.

5-2 Recommendation

As we mentioned the quality of software and the questionnaire made to search the high performance , it is recommended that we educate the institution of the high importance of this questionnaire for everybody and that it is not a record of their operations or any other kind of detection that may break the secrecy of the company and its business. That will help to eliminate the hesitation in giving true statements or any other kind of fear might be generated. On the other hand, it is advisable that the institutions ensure the compatibility of the software with the universal standard before applying the software on their operations to avoid the negative effect on the long run business.

Instructing university and high institutions students the methodology of software engineering and accentuate on its importance in designing and programming software. Documentation must have its importance in designing any software for the future developing and maintenance.

Establishing a local division for a local standard which comply with the universal standard and function as a control on local organizations producing and developing software to assure the stability of the software standard.

I recommend the future work in the same field to add the database and the programming language as two additional elements in their questionnaire since they are closely related to the software quality. Also they must expand the base of search to cover the different regions of Libya.

References

1. BASILI V.R. & ROMBACH H.D.,The TAME Project: Towards Improvement-Oriented Software Environment, IEEE Transaction on Software Engineering, Vol: 14, No. 6, pp. 758-773 (June 1988)

2. BASILI V.R. & WEISS D.M., A Methodology for Collecting Valid Software Engineering Data, IEEE Transaction on Software Engineering, Vol. SE-10, No. 6,pp. 728-738 (November 1984).

3. Boehm, B. W., Brown, J. R., Lipow, M. Quantitative Evaluation of the Software Quality, Proc. of 2nd Intl. Conf. on Software Engineering, pp. 592-605 (1976):.

4. Donald G. Firesmith. Common concepts underlying safety, security, and survivability engineering. Carnegie Mellon Software Engineering Institute-Technical Note CMU/SEI- 2003-TN-033, (December 2003).

5. Dromey, R. G., "Concerning the Chimera [software quality]", IEEE Software, no. 1, pp. 33-43, (1996).

6. Dromey, R. G., "A model for software product quality", IEEE Transactions on Software Engineering, no. 2, pp. 146-163, (1995).

7. Grady, R. B., Practical software metrics for project management and process improvement, Prentice Hall, (1992).

8. ISO-IEC. International Standard 9126. Information technology soft-ware product evaluation-quality characteristics and guideline for their use, Geneva(1991).

9. J. McCall, P. Richards, and G. Walters. Factors in Software Quality. RADC TR-77- 369 1977. US Rome Air Development Center Reports NTIS AD/A-049 014,015,055, (1977).

10. Kent Beck, Extreme Programming Explained: Embrace Change. Addison-Wesley Professional, Oct, (1999).

11. L. Buglione and Abran A. Geometrical and statistical foundations of a three dimensional model of software performance. Advances in engineering software, 30:913 919, (1999).

12. L Buglione and Alain A. A Quality Factor for Software, Qualita99 (3rd International Congress on Quality and Reliability) , Paris, France, ISBN 2-900-781- 43-4,335-344 (March 25-26, 1999)

13. Lionel C. Briand, John W. Daly, and J for coupling measurement in object-oriented systems, IEEE Transactions on Software Engineering, 25(1):91 121, (January/February 1999).

14. Martin Neil and Norman Fenton. Predicting software quality using Bayesian belief networks. NASA/Goddard Space Flight Centre, (December 1996).

15. Maryoly Ortega, Mara A. Perez, and Teresita Rojas. A systemic quality model for evaluating software products. Laboratorio de Investigacin en Sistemas de Informacin.

16. Roger S. Pressman, "Software Engineering: A Practitioner's Approach" McGraw-Hill , (2005).

17. Ronan Fitzpatrick. Software quality definitions and strategic issues Staffo rdshire University, (1996).

18. S. Morasca and L. C. Briand. Towards a Theoretical Framework for Measuring Software Attributes. In Proceedings of the 4th International Software Metrics Conference, pp119-126, Albuquerque, New Mexico, (November 1997).

19. Stephen H. Kan. Metrics and Models in Software Quality Engineering , Second Edition (2005) .

20. Wolfgang B. Strigel, Geo Flamank, and Gareth Jones. What are software metrics? Software Productivity Center Inc., (1992).

21. Webster Dictionary, http://www.m-w.com/

22. Web Services Activity http://www.w3.org/2002/ws/

ملخص الرسالة

إن بناء وتطوير جوده البرمجيات هو ما تركز عليه كل الشركات.هذه الدراسة تركز على تقديم دراسة لجوده البرمجيات كأداة فعالة في إدارة و تنظيم العمليات اليومية في العديد من المنظمات المحلية في ليبيا .

إن المنظومات كمنتج قد تكون أكثر ملائمة لبيئة عمل من البيئات الأخرى. و اهتمامنا هنا أن نتأكد من تطابق المنظومات المحلية مع القياسات العالمية وان نتأكد أن هذه المنظومات تلبى احتياجاتنا.

آملين أن تعكس هذه الدراسة صوره حقيقية وان تكون مرجع لتطوير البرمجيات المحلية، و قد اتبعنا طريقه الاستقراء في دراسة وجمع المعلومات معتمدين على خواص نموذج ماكول McCall's modelفي تحديد جوده البرمجيات لنتأكد من مدى كفاءة البرمجيات المحلية .

لقد استهدفنا المنظمات الكبرى كحقل لبحثنا بما أنها يمكنها أن تعكس صوره صادقه بعملياتها اليومية الكبيرة وهذا كاف ليعكس جوده البرمجيات في العمل على المدى الطويل وأيضا لانّ هذه المنظمات تحتوى على كل الأطراف القادرة على الحكم على جوده المنظومة كمنتج (مستخدم ، مطور ، مدير) .

ولكي نكمل هذه الدراسة بنتائج موثقه، فقد اخترنا برنامج SPSS لتحليل المعطيات المتحصل عليها. كذلك استعملنا معامل كرونباخ للتأكد من موثوقية نتائج بحثنا.

وقد حصلنا على نتائج متوسطه بعد تطبيق دراستنا على المنظمات المحلية. و عموما فان جودة البرمجيات المحلية فوق المتوسط ولكنها ضعيفة في بعض الخصائص وهي قابلية إعادة الاستخدام reusability وقابلية نقل النظام من بيئة لأخرى portability وقابلية اتصال النظام مع النظم الأخرى interoperability .

نحن يمكن أن نتحصل على نتائج أفضل إذا قمنا بتحسين برامجنا المحلية في هذه الخواص .

الجماهيرية العربية الليبية الشعبية الاشتراكية العظمى

جامعة التحدي

سرت

إن الدراسة ليست غاية في حد ذاتها وإنما الغاية هي خلق الإنسان النموذجي الجديد

التاريخ : ....

الموافق : 2009/2/1

الرقم الإشاري : ك.ع/2009/1/56/928

كلية العلوم

قسم الحاسوب

عنوان البحث

ضمان الجودة للبرمجيات المطلقة باستخدام نموذج الماك كول

مقترح من الطالب

أحمد صالم محمد عبد الله المجبري

** لجنة المناقشة :

1 – د. محمد أحمد اخليف ....

(مشرفاً)

2 – د. ادريس سامي الفقيه ....

(ممتحناً داخلياً)

3 – د. عمر مصطفى الصلابي ....

(ممتحناً خارجياً)

يعتمد

د. أحمد فرج مبروك

أمين اللجنة الشعبية لكلية العلوم

الجماهيرية العربية الليبية الشعبية الاشتراكية العظمى
جامعة التحدي
أمين اللجنة الشعبية لكلية العلوم

WWW.altahadi.edu.ly

Info@altahadi.edu.ly 674 ✉ 054/60361 - 62151 📠 054/60363-65704 ✆

جامعة التحدي
كلية العلوم
قسم الحاسوب

ضمان جودة البرمجيات المحلية
باستخدام م نموذج ماكول McCall's model

مقدمة من الطالب

احمد صالح محمد عبدالله

إشراف : د.محمد احمد اخليف

• رسالة مقدمة لقسم الحاسوب

كمتطلب جزئي للحصول على درجة الماجستير في علوم الحاسوب

العام الجامعي 2009/2008