Evaluation of Measures Defined for Quality Attributes

Bommireddy Madhav Rao, Bomma Rama Krishna, T. Agnivesh, U. L. N. S. Prasad, Ch. N. N. Jyothi
Dept. of CSE, A. K. R. G College of Engg & Technology, Nallajerl, AP, India

Abstract
There is an increasing research going on evaluation of measures which enhance the software product quality. Unfortunately many organizations perhaps in some situations will be in a situation where they are unable to decide which measure to choose in a particular situation. The objective of our research is carried out on evaluating measures for a set of defined attributes. Firstly, we have compared different quality models and chosen ISO model out of other quality models like Boehm’s model, Mc Call’s model, Furps/Furps++ model, Dromey’s model apart from choosing the model we have also stated the reasons for selecting this particular model. Furthermore, we have defined attributes and sub-attributes of ISO model. In addition to this we have selected a couple of sub-attributes for which we have listed out some metrics and the vital part of our research is to list out some suggestions for an organization to choose by listing out some weakness and strengths of the sub-attributes of ISO model as well as other quality models. Thereby, in our research we also made some suggestions to the organization which measure to choose and which will be a suitable one in particular situations and by choosing better measures results in delivering a quality product.

Keywords
Quality Attributes, ISO Model, Quality Products, Quality Measures

I. Introduction
Our paper focuses on quality as we measure quality from product perspective. In dealing with this paper we have made a detailed study on quality and different quality models. We have also addressed the drawbacks of quality. The most important part of our paper is that we have described a list of measures for ISO model attributes and Sub-attributes. In our paper we have only discussed about two attributes i.e., Functionality and Reliability and the sub-attributes under these quality attributes. We have also stated the reasons for choosing the functionality and reliability attributes. Our main goal of this paper is to give some suggestions to the organizations which would be useful for them to improve the quality of the software product using these measures.

II. Quality
Quality – it is defined as either conformance to product specification or fitness for use of a product. Conformance to product specification means the measurable characteristics of the product developed must satisfy a fixed specification. The quality of a product is said to be good, if it satisfies needs of the customer. It also defines the capacity of the product. How flexible it can be worked up on. That is different people who have defined quality in their own way. Cross by defines quality as “The word “quality” is often used to signify the relative worth of something in such phrases as “good quality”, “bad quality” and “quality of life” - which means different things to each and every person. As follows quality must be defined as “conformance to requirements” if we are to manage it. Consequently, the nonconformity detected is the absence of quality, quality problems become nonconformance problems, and quality becomes definable.” From the above definition Crosby defines that:

• Quality is defined as how well the product conforms to its requirements
• Crosby believes in prevention rather than reflection of errors
• He also believes that it is absolutely possible to reduce the errors to zero.
• Quality cost can be reduced with appropriate training.

A. Problems in Quality
Some of the major problems that are related to quality are, it is impossible to predict quality costs involved in the project. Sometimes quality measures are erroneous and may lead to failure of the product. Lack of updating in the quality models can lead to many ambiguities. Quality remains a too complicated issue because many interaction between software artifacts and the subtle nature of software. The various problems pertaining to quality include are as follows: Different tastes and values – often the evaluation methodology used by each person differ from each other. Individual assessment of quality is not practically possible. From metrics point of possible, it is not ever possible to measure it accurately. Evaluating only code in software engineering doesn’t ensure quality. Because it is not possible to measure performance and other functionality related to code execution accurately.

B. Problems of Quality From Product Perspective
Quality from the product perspective is very important. Quality for a product must describe all the essential features of the product that the product performs as per its specification. Quality from product perspective, we make a study on different product quality attributes and how it evolves. If the product does not perform as per its specification it is a concern. One of the main things we focus when talking for quality from product perspective, is the attributes associated with the quality models. We have different attributes and problems associated with it. For instance, availability is always a problem when working on system dependability because; it is not possible for the systems to be in available state for a dependable system. When an available requirement is high, then it is not possible to get a high reliable system. The cost of measuring maintainability is relatively high. Also a system fails when it performs differently from what is expected from it to perform. It is very hard to measure failure in the catastrophic environment. Due to lack of standardization the security implications applied on quality may vary.

III. Quality Models
Quality models play a pivotal role while measuring the software quality. When considering some years back situation i.e., since 1920, progress in software quality field is named for time consuming and ambiguous besides there are no clear methods stated for assessing the quality of a software product. Fortunately, now-a-days there are a quite number of well known qualities models which are used in organizations to build the quality of a software product. Well, we defined a quality model as “A standard taxonomy of quality attributes”[2]. A quality model clearly defines specific characteristics, quality attributes and measures of the sub-attributes and it also specifies the required characteristics
that software exhibits. Some of the benefits of a quality model are as follows:

- It validates completeness of requirement definitions.
- It identifies software requirements and design objectives.
- It also identifies software testing objectives and also the user acceptance.

Criteria for a complete software product [2]. Many quality models tie different attributes together. Each model helps us to understand several factors while carrying out the evaluation of quality of a software product. Here each model has different set of attributes at all levels which may differ besides they also have different number of hierarchal levels. One difficulty with these quality models is hierarchy is strict and the quality characteristic at each high level are related to exactly to a set of sub-attributes. These high level quality factors are assumed to be independent with each other and are related to the software from user’s point of you rather than considering it from the perspective of the whole system characteristics [3].

A. Background

Software quality is one of the most essential paradigms in the structure of software engineering. Producing high quality software is a preliminary requirement for satisfying the reliability and error free requirements of command and control software [5]. Although software quality has been the core entity in the software engineering environment, it is still a very premature topic. Researchers are unable to decide and study what exactly the word quality pertains to in field of software engineering, because there is no perfect measurement for quality. Due to modernization and globalization, many organizations have invested their large part of capitals on developing the business structure. There is a huge reliance on the software products from the point of automating the business to making strategic roles in the business. To make this happen, the software must be of a high quality, and it should withstand many complexities [4]. It is not easy to develop a high quality software product in the market. It requires various kinds of measures for the attributes. Apart from this, the system complexity measurement affects the quality measurement of a software entity. The only possible way to avoid this is software quality assurance [2]. Although software quality assurance is the solution to the quality attribute of a software product, it still lacks control of providing a high quality product and hence a more powerful technique is introduced which is called as Software Quality Assurance Guidelines. Achieving quality in software is the pivotal goal for every software development organization besides the software quality is the most key aspect in the structure of software engineering. A software product should be designed in such a manner that the requirements should meet the product and by this we can say that the output is to get a software product with good quality. Although software quality is the most important entity, the quality achievement in every organization cannot be the same and successful. Researchers have proposed so many guidelines in order to follow and make a software product with a good quality but they were not up to the mark to define the measures or guidelines to be followed perfectly. In fact a lot of research has been done in this area by providing many facilities for enhancing quality of a software product up to the level and unfortunately it is not a common task. So there should be some proposed effective approaches on the quality issues of a software product. The main focus of this research paper is to enhance the quality management and bring out the effective inkaapproaches on the quality issues. In this paper we present an overview of different quality different quality models. We present an effective approach for evaluating these quality models. We have made a literature study on quality and quality attributes and sub attributes associated to different quality models. We select two important attributes from ISO model and make a study related to that attributes and try to provide appropriate measures for the two selected attributes, that can gives options to an organization to work on the measures for product quality development.

IV. Problems of Quality Models

There are quite a number of problems with the quality models when applied directly and some of the problems are addressed below Firstly, quality is defined as meeting requirements however from a practical point of view to fit into all application domains in one generic model it is very much essential for us to develop a flexible conceptual quality framework. Secondly, the higher level quality factors are measured by a set of certain measures which are used for quantification of other quality attributes. Thereby, the data that is available to support an indirect measure and it also supports many other indirect measures besides it can support the other measures too. Problems here is these models take high-level quality attributes which are independent of each other for granted. Basing on this assumption these high-level quality factors decompose into lower levels. Thirdly, quality models are static and they don’t describe how to measure quality from current values. Therefore, measurement of the system should consider both deterministic and probabilistic approaches in measurement of framework. Lastly, there exists complexity between direct measure and quality factor as well between the indirect measures and quality factors which leads to difficulty in determining overall quality.

V. Our Research Model

In our research paper we have chosen ISO model and we have specified the reasons for choosing this particular model apart from that describe in brief about the attributes and sub-attributes of ISO models.

A. ISO Model

The acronym of ISO is “International Organization For Standardization”. ISO is an international quantity system standard which is applicable to all the organizations within all types of business and it is a process oriented towards quality management. ISO initially addresses an organizations processes and methods and externally manage the quality issues of delivered products and services. This model proposes designing, documenting, implementing, supporting, monitoring, controlling and enhancing the processes.

B. Reasons for Choosing ISO Model

Mostly the overall structure of ISO 9126 -1 is similar to past models that are Mc Call (1997) and Boehm (1978), although there have couple of notable differences. We can observe some changes in functionality characteristics like in many requirements there are specifications for all characteristics but the specified requirements are not pure functional requirements that are specified as non-functional requirements. One interesting aspect about this ISO 9126 model is here compliance is seen as functional characteristics. By using ISO 9126 for derivation of system requirements it gives clarity of definition of purpose and operating capability. When looking at Mc Call’s model it is noted that unfortunately many of the measures defined by Mc Call can be measured only subjectively.
So this leads difficulty in using this framework to set precise and specific quality requirements. Furthermore, some measurable properties like traceability and self-documentation are not really defined meaningfully besides this model is not applicable with respect to the criteria outlined in IEEE standards. Moreover, it emphasis on the product perspective of quality to the detriment of other perspectives too. While discussing about the Boehm’s model the measurable properties and characteristics concentrate as highly technical stakeholders that are typically involved in software life cycle. Characteristics like general utility and As – is utility are too generic and imprecise to be useful for defining verifiable requirements. In the case of Dromey’s quality model, this model focused on the minute details of quality. In this model, the high level characteristics of quality will manifest themselves if the software product from the individual requirements to the programming language variables which exhibit quality carrying properties. Apart from all the quality software models according to our point of view we think that ISO is better because if the following reasons

• It defines an updated quality model.
• It defines a set of external measures.
• It defines a set of internal measures.
• It defines a set of quality in use measures.
• It recognizes all perspectives of quality as important contribution for overall assessment of quality.
• It takes an incremental approach to the software quality that begins with quality in use and something that is easy to grasp for non-technical stake holders and ends with internal quality, something more technically inclined stakeholders will feel more comfortable.

C. Attributes of ISO
ISO is an international standard for evaluating software. It represents the latest and ongoing research by characterizing the software for the purposes of software quality control, software quality assurance and software process enhancement. The ISO model software quality model identifies six main quality attributes besides these attributes have sub attributes in ISO model namely:

1. Functionality
Functionality plays a pivotal role in a software product. The relation between software within all business process and is outside the scope of ISO. It is the only software functionality and essential purpose of the software component.

2. Reliability
Reliability is considered to be an attribute which defines the capability of the system in order to maintain its service provision under defined conditions for defined periods of time.

3. Usability
Usability exists with regard to functionality and gives the ease to use for a given function. This attribute can define as the ability to learn how to use a system and learn ability is also a major sub-attribute of usability.

4. Efficiency
Efficiency is concern with the system resource used when providing the required functionality. It is concerned with the amount of disk space, memory and network which can be a good example of this attribute.

5. Maintainability
Maintainability is impacted by the code of readability or complexity as modularization. This attribute can be defined as the ability to identify and fix the faults within a software component. In other quality models this attribute is termed to be supportability. In this particular attribute testability is one of the sub-attributes of maintainability.

6. Portability
Portability explains how the software can get acclimatize and get adopted within the environment or requirements. The sub-attribute of portability is adaptability.

D. Sub-Attributes of ISO Model
Besides the characteristics we have sub-characteristics for each attribute in ISO model namely:

1. Functionality Sub-attributes:
   (i) Suitability
   (ii) Accurateness
   (iii) Interoperability
   (iv) Compliance

2. Reliability
   Sub-attributes:
   (i) Maturity
   (ii) Fault Tolerance
   (iii) Recoverability

3. Usability
   Sub-attributes:
   (i) Understandability
   (ii) Learn ability
   (iii) Operability

4. Efficiency
   Sub-attributes:
   (i) Time Behavior
   (ii) Resource Behavior

5. Maintainability
   Sub-attributes:
   (i) Analyzability
   (ii) Changeability
   (iii) Stability
   (iv) Testability

Out of these six attributes we have selected the two main attributes that is functionality and reliability in addition to these we have described in detail about the sub-attributes and also the measures used among the above five attributes to be described in a detailed manner in our research paper.

VI. Measures Descriptions of Attributes From Different Quality Models

A. Functionality
It is the ability of the product to perform up to its functional specification for its specified operation.

SUB-Attributes For Functionality in ISO Model
1. Suitability
   It is specified as attribute of a software product that relies on presence of appropriate functions in the software.
   Measures for suitability:
   a. Coverage ratio: It may be defined as the percentage of desired functionality that is actually present in the complete software
product.
• Calculate the percentage of desired functionality of the product.
• Calculate the total functionality of the specified product
• Coverage ratio- desired functionality/total functionality.
b. Functional specification ratio change: It may be defined as The ratio of number of changes the functionality of the software as exhibited (change includes addition, deletion of new features)
c. Improvement request ratio: This may be defined as The number of improvements in the design, the user seeks after the delivery of the product.
d. Claims ratio: The total number of claims made by the user makes per month
2. Accuracy: The attribute that deals with the ability to provide accurate measure related to the functionality of a software product.
Measures for accuracy:
a. General measure
• Prepare a set of possible operations that can be applied on the product/
• Present this operations set to test the product.
• Compare the results from (2) with the actual expected result test outcome/
• Determine the list of operations that are not processed properly.
• Ratio = number of test sets / complete test size.
b. Significant digit ratio
The ratio of the implemented functional digits to the complete set of significant functional digits.
c. Manual conformance ratio: The ratio of functions that are implemented properly and the product that are matched to the functions written as per users requirements.
d. Rounding treatment ratio: The ratio of functions that can perform nearly to its specification to the total number of implemented functions.
3. Interoperability- Ability to perform perfect internal operations without fail.
Measures for Interoperability:
a. Effort interaction ratio
The effort needed to realize interoperability per unit of internal operation performed.
• Select explicitly which interoperatability is desirable for which environment;
• Measure the effort needed to realize this interoperability;
• Measure the interoperability using function points or some related metrics.
• Calculate the effort per interaction, which is (2) / (3) / total number of interactions that can be applied.
b. Matched data format ratio: The ratio of data formats matched to those of the other system in the interoperation.
c. Matched character ratio: The ratio of graphic characters and control characters matched to those of the other systems in interoperation.
d. Matched interface ratio: The ratio of interfaces matched to those of the other systems in interoperation.
e. Observed standard ratio: The ratio of observed standards to the standards made in the system or among the system.
4. Security: The attribute that is defined, to protect the system from unauthorized uses whether accidental or deliberate that effect the program.
Measures for security:
a. Resistance: Protect the system from all kinds of external access that access the system without authorization.
• Find a security expert;
• The security will be responsible for all the security related issues, control and management of software.
b. Hacker resistance: A group of hackers will not be able to access the system, and physical attributes of a system
1. Select a group of renowned hackers;
2. Determine the time put at their disposal;
3. When after each trial period no hacker is able to succeed, then the system is said to be hacker resistant.
c. Ciphered data ratio: The ratio of ciphered data to the data to be ciphered.
d. Access history ratio: The ratio of confidential information that have access histories to all confidential information.
e. Data damage ratio: How many times per month was any data damaged.
5. Traceability- This attribute specifies whether the data is being accessed correctly at required points.

Measures for Traceability
a. Operation control effort
The amount of time that is lost while processing due to operation control activities, manually or automatically.
• Carry out the operation control activities, and determine the processing time needed for sub tasks;
• Determine the time taken for normal control activities
• Operation control effort is the ratio of operation control time and normal processing time.
b. Ease of operation control:
Number of requirements to total number of requirements [11-12].

SUB ATTRIBUTES FOR FUNCTIONALITY IN DROMEYS QUALITY MODEL:
Measuring sub attributes for functionality
Correctness – is a sub attribute for functionality model. Measuring defect density is the appropriate measure for correctness
DD= Total number of defects found during the design inspection/ Number of lines of source code of design statements [11-12]

Sub Attributes For Functionality in Purp’s Quality Model
Measure for functionality in purp’s model
a. Human Factor- This factor relates that human factor is main role in adjudging the functionality of the product.
HF – Average user effort involved/ total number of defects encountered.
b. Capacity – This attribute specifies the maximum capacity of the product to perform up to its specification
So Capacity= ratio of how much the product is capable of performing up to its specification / total capacity expected from the product
c. Joint of characteristics- This attribute specifies the total weight age of all characteristics that can improve the functionality of the product. It also defines how well the combined characteristics’ can help to motivate the product.
JOC = INDIVIDUAL CHARACTERISTICS EXHIBITING FUNCTIONALITY / COMB OF CHARACTERISTICS [11, 12].

6.2. Reliability: - It is termed as the capability of the software product or a program to extend and maintain its level of performance under any specified conditions and to perform its
intended functions [13].

Sub Attributes for Reliability in ISO Model
1. Maturity: This attribute refers to the probability of the software that bears the failure frequency which is caused by faults in the software.

Measures for Maturity:
- **a. Mean Time between Failures**: This measure helps the organizations to calculate the average time that passes between the two failures. It also helps to determine the relevant failures and how long the measurement should be carried out in order to ensure reliable results. Using this attribute we have to measure the interval lengths between the restart of the software product and a failure during this period.
- **b. Mean Time To Failure**: This measure is used to calculate the mean time between the failures i.e., the time between one failure occurrence and the succeeding failure occurrence during a given period of time.
- **c. Product Fault Density**: This measure is useful to calculate the ratio of the number of faults in the released product to the unit volume of the released product. The volume of the product can be measured as the number of pages or KLOC i.e., either by using the user documents or the source code. This measure is very useful to measure the maturity of the software product.
- **d. Test Density**: This measure is used to calculate the ratio of the test volume conducted during the phases of development to the unit volume of a product tested and released. The test volume can be measured by the number of test cases. It is a useful measure which ensures the customers that all the test cases have been carried out successfully and also the product delivered is of a good quality. So, this measure has to be taken care in order to attain maturity.

2. Fault Tolerance: This attribute refers to the ability of the software to tolerate and diagnose faults when the system is put in transition.

Measures for Fault Tolerance:
- **a. Disturbances**: This measure is used to find out how many times the processing has been halted due to the incorrect use within a certain period of time. The main purpose of this measure is that it helps in determining the period of measurement by registering the number of times the processing has been halted and evaluates the case in which this halting was caused by incorrect use.
- **b. Break Down Ratio**: This measure is used to calculate the ratio of the number of observed breakdowns to the number of observed failures through a given period of time. It is a useful measure and it has to be taken care so as to avoid the failure of the project.
- **c. Operational/Input Error Detection Ratio**: This measure is used to calculate the ratio of the number of erroneous operations or inputs detected by a software system to the number of the erroneous operations or inputs conducted during the given period of time.
- **d. Recoverability**: This attribute refers to the ability of the software to recover from its fault/failure at a specified period of time.

Measures for Recoverability:
- **a. Mean Time To Repair**: This measure calculates the average time needed to recover the damaged software product. The main purpose of this measure is that it measures the duration of the periods in which the software product is out-of-order and calculates the average duration of these periods.
- **b. Auto Recovery Ratio**: This measure is used to find out the ratio of the automatically resolved failures to that of the total number of failures that requires automatic recovery. The purpose of this measure is to determine the failures and estimate their possibility of occurring by developing test cases for these failures. This measure also helps in determining which ratios of failures are automatically resolved by the software product.

Sub Attributes for Reliability in McCall’s Model
1. Consistency: This attribute refers to the ability of software to perform in a constant way under any exposure to various environments without any faults

Measures for Consistency:
- **a. Number of Source Lines of Code**: For a project to be consistent enough the number of source lines of code should be less i.e., the less the lines of code (size of the project) then more will be the scope of the project. In other terms, if the size is less then the effort required will also be less. This measure is important and has to be taken care in order to attain consistency and this can be done by minimizing the project size thus making the project successful.
- **b. Number of Test Cases**: Test cases refer to the set of statements or conditions that are used by the tester to determine whether the requirements are satisfied. Requirements should be fully satisfied and for this there should be many test cases. The number of test cases determines the satisfaction of requirements and to test the correct behavior of the application. This measure is very useful and the organizations should take care of this measure by increasing the number of test cases i.e., more the number of test cases more the probability of project success because of the more understanding of the requirements which is very important.

2. Error Tolerance: This attribute explains that the errors have to be tolerated and give some acceptable result.

Measures for Error Tolerance:
- **a. Number of Input/output Related Errors**: This measure is used to calculate the number of input/output errors i.e., if some input’s are not correct which results in the wrong output it should not totally stop the project execution rather it should be able to accommodate changes and correct them accordingly so that the end result is not terminated but rather it should give a satisfactory result. We suggest that this measure should be used which will help the organizations to maintain a dependable, flexible and a fault tolerant system.
- **b. Number of User errors**: This measure helps the organizations to accommodate the user changes (or) errors i.e., if the user gives a wrong data/entry then the system should be able to accept the errors and give an appropriate solution to the user entry rather than just discarding the data and terminating the process. So, no matter of the user errors the system should work and generate some related output.

3. Accuracy: This attribute is been discussed along with its measures in the section 6.1 under the Sub Attributes for Functionality in ISO Model [15].

Sub Attributes for Reliability in Boehm’s Model
1. Accuracy: This attribute is been discussed along with its measures in the section 6.1 under the Sub Attributes for Functionality in ISO Model.
2. Completeness: This attribute is an important characteristic that determines the quality of a conceptual model. Completeness can be measured as the ratio of the performance delivered by the...
Measure of Completeness:

a. Performance: This measure is useful in determining the completeness of a software product. Good performance ratio means the product is ‘complete’ with its specification.

4. Robustness/Integrity: This attribute refers to the “degree to which an executable work product continues to function properly under abnormal conditions or circumstances”. Robustness can be measured by Fault Tolerance and Dependability [14].

Measures for Robustness:

a. Fault Tolerance: This measure is useful in the organizations to withstand the pressures, stress and changes and see that the product is functioning correctly even though there are invalid inputs. This can be done by anticipating the situation and controlling the conditions accordingly.

b. Dependability: This measure depends on the system, process, product, etc to continue its operation although there are some changes in the environmental conditions and presence of partial failures.

5. Consistency: This attribute has been discussed along with its measures in the section 6.2 under the Sub Attributes for Reliability in McCall’s Model [15].

Sub Attributes for Reliability in Furp’s Model

1. Frequency and severity of Failures: This attribute refers to the ability to calculate the frequency of the failures and also the effect caused due to the failure.

Measures for Frequency and severity of failures:

a. Fault Count: This measure is used to calculate the total number of faults occurred during a unit time. It is a useful measure as it helps the organizations to have a track of the total faults which will be helpful to determine the quality of the software product.

2. Recoverability: This attribute has been discussed along with its measures in the section 6.2 under the Sub Attributes for Reliability in ISO Model.

3. Time among Failures (MTTF): This attribute is used to calculate the average time between the failures during the execution of the software product.

Measures for MTTF:

The number of frequent changes the product has encountered during its execution can be measured to ensure the customers about the quality of the product i.e., less the number of failures the better will be the product.

4. Accuracy: This attribute has been discussed along with its measures in the section 6.1 under the Sub Attributes for Functionality in ISO Model [15].

Sub Attributes for Reliability in Dromey’s Model

1. Correctness: This attribute refers to the extent to which a program can satisfy its requirements and accomplishes the user’s mission and objectives. Correctness can be measured by using the Defects/KLOC and Failures/hours of operation.

Measures for Correctness:

a. Defects/KLOC: This measure is used to calculate the number of defects per Lines of code. It is a useful measure to ensure correctness in reliability. The organizations should carefully assess this measure because the less number of defects then the more correct i.e., accurate is the software product.

b. Failures/Hours of operation: This measure is used to calculate the number of failures encountered to the number of hours of operation i.e., the effort. This is a useful measure to the organizations because it helps in counting the failures during the development and evaluation of the software product and the effort required to resolve the failures [15].
<table>
<thead>
<tr>
<th>Interoperability</th>
<th>Effort interaction ratio</th>
<th>This indicator can only be measured after realizing a relative number of interactions. Also, the effort per interaction depends on the interoperability of both software products involved.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Matched data format ratio</td>
<td>Gives the practitioners a scope for exploring data relevant to the specific operations. A higher value is preferred.</td>
<td></td>
</tr>
<tr>
<td>Matched character ratio</td>
<td>This metric may be appropriate in situations where the portability of software is more or less flexible. A higher value is preferred, because it provides a better scope.</td>
<td></td>
</tr>
<tr>
<td>Security</td>
<td>Resistance</td>
<td>This provides companies to work with appropriate security experts, who as a domain functional knowledge and provides an instance for deciding security aspects.</td>
</tr>
<tr>
<td>Hackers resistance</td>
<td>This indicator only deals with appropriate software measurements. Security can be considerably improved by physical and organizational measurements.</td>
<td></td>
</tr>
<tr>
<td>Ciphertext data ratio</td>
<td>This can be used by companies when they are not sure with unauthorized access and anonymous access.</td>
<td></td>
</tr>
<tr>
<td>Data damage ratio</td>
<td>This measure can be used to predict the frequent loss of data due to unauthorized access.</td>
<td></td>
</tr>
<tr>
<td>Access history ratio</td>
<td>This can be used when the information can be accessed by some set of known entities.</td>
<td></td>
</tr>
<tr>
<td>Traceability</td>
<td>Operation control effort</td>
<td>The above measure can be used in situations where the degree of operation on data is relatively high and also one can predict how much effort is required for operation control.</td>
</tr>
<tr>
<td>Ease of operation</td>
<td>Based on the number of requirements met, one can predict the traceability of software.</td>
<td></td>
</tr>
<tr>
<td>Correctness</td>
<td>Defect density</td>
<td>Defect density provides an appropriate measure for tracing the number of requirements specifications met by the product.</td>
</tr>
<tr>
<td>Human factor</td>
<td>Human effort ratio</td>
<td>Amount of human effort that can be involved in deciding the software product.</td>
</tr>
<tr>
<td>Capacity</td>
<td>Capacity ratio</td>
<td>This ratio shows how well the product is capable of performing to its specification.</td>
</tr>
<tr>
<td>Joint of characteristics</td>
<td>Joint of characteristics ratio</td>
<td>Shows how well the combined features of the product can help in maintaining functionality.</td>
</tr>
</tbody>
</table>
Reliability

<table>
<thead>
<tr>
<th>Sub Attribute</th>
<th>Measure</th>
<th>Evaluation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Motivation</td>
<td>Mean Time Between Failures</td>
<td>A higher value is preferred because it shows the performance of the product, more the mean time failure value more the functionality of the product.</td>
</tr>
<tr>
<td>Motivation</td>
<td>Mean Time To Failure</td>
<td>A higher value is preferred because it shows the performance of the product, more the mean time failure value more the functionality of the product.</td>
</tr>
<tr>
<td>Product Fault Density</td>
<td>A lower value of fault is preferred because the less the value the more the quality of the product.</td>
<td></td>
</tr>
<tr>
<td>The Density</td>
<td>A higher value is preferred because a product can be successful only when the number of state conducted during the development stages are more.</td>
<td></td>
</tr>
</tbody>
</table>

VIII. Strengths & Weakness of Metrics of ISO

After making a literature review, we present some of the potential strengths and weakness of measures for functionality and reliability. Some of the main advantages of measuring functionality are that the measures provide an appropriate frame work to evaluate quality. Besides the technique not being effective, it yields results which have relationship with the quality of software. Also helps in providing more effective testing implementation methods. Sometimes good measures provide an idea of how well the product is evolved. The measures related to functionality helps to assess the quality more from the user perspective, compared to developer’s perspective. Some of the weakness of measure for functionality is the inability to quantify or predict the quality of the product developed, because of this it is not possible to measure attributes like reliability accurately. Some of the strengths of measures related to reliability includes, the measures can predict the capacity of reliable operations that can be performed on the product. Sometimes it is very hard to measure reliability because it is the most difficult non behavioral requirement is SRS. Also when measuring reliability, the measures that are developed for hardware are sometimes escaped into software domain. MTTF is used to measure some levels of software, besides it can be sometimes false. MTTF can be sometimes used to predict software reliability, but can sometimes give wrong results. The failure can also be due to user, not software.

IX. Suggestions to the Organizations

We hope that our work serves as a purpose for organization to improve the quality standards. Our evaluation of metrics for different quality attributes from different models, can help organizations for developing right kind of product, with more emphasis on quality. It may help our organization to select right kind of product at right situation for product evaluation. We also make recommendations for organization to select appropriate measures to make the product more evolved and make maximal utility of quality.

X. Conclusion

After a detailed study on various issues related to quality, quality models and measure for different attributes from different models, we would like to conclude that quality is the most important paradigm in the field of software engineering. Without appropriate measure for quality, it’s not possible to produce a good successful product in the market.

References

Mr. Bommireddy Madhav Rao is working as an Associate Professor and Head of the Department of CSE in A.K.R.G. Engineering College, Nallajerla, W.G.Dt, A.P. He has received his M.Sc (Computer Science) degree from Andhra University, and M.Tech (CST) from Andhra University, Visakhapatnam, A.P. India. His research interest includes Data Mining, Network Security, Bio-Informatics.

Mr. Bomma Ramakrishna is working as an Associate Professor, in CSE Department, A.K.R.G College of Engineering and Technology, Nallajerla, W.G.Dt, A.P, India. He has received his B.Tech CSE from JNTUH, AP and M.Tech (SE) from JNTUK, A.P., India. He has published many papers in International Journals and Conferences. He is currently pursuing the Ph.D Degree. His research interests include Software Engineering, Data Mining, Network security, Web security and Bio-Informatics. His research area is Protein Disorder. He is having 7 years of Teaching Experience.

T. Agnivesh, pursuing B.Tech (CSE) from AKRG College of Engineering & Technology, Nallajerla, Affiliated to JNTUK University, Kakinada.

U.L.N.S. Prasad, pursuing B.Tech (CSE) from AKRG College of Engineering & Technology, Nallajerla, Affiliated to JNTUK University, Kakinada.

Ch.N.N. Jyothi, pursuing B.Tech (CSE) from AKRG College of Engineering & Technology, Nallajerla, Affiliated to JNTUK University, Kakinada.