Software Quality Analyser

Vibhash Yadav
Associate Professor
Krishna Institute of Technology, Kanpur
Vibhashds10@yahoo.com

Abstract- This paper describes an improved analysis view of software attribute dependency relations for the assessment of high-level design quality attributes in object oriented designs. It describes relationships and dependencies of quality attributes such as reusability, flexibility, understandability, functionality, extendibility, effectiveness. This paper includes the study of a hierarchical model for object-oriented design quality assessment and the empirical study of object-oriented metrics. The relationships, or links, from design properties to quality attributes are weighted in accordance with their influence and importance.

Index Terms- Quality Attributes, Design Metrics, Coupling, Cohesion, Reusability, Flexibility.

1. Introduction

In a world of highly competitive products, the importance of delivering quality is no longer an advantage but a necessary factor for a company to be successful. While there is uniform agreement that we need quality software, the question arises how, when, and where you measure and assure quality are far from settled issues.

Traditional software product metrics that evaluate product characteristics such as size, complexity, performance and quality must be changed to rely on some fundamentally different notions such as encapsulation, inheritance, and polymorphism which are inherent in object-orientation. This has led to the definition of many new metrics.

Many of the metrics and quality models currently available for object-oriented software analyses can be applied only after a product is complete. And as per the objective of our project there is a need for metrics and models that can be applied in the early stages of development to ensure that the analysis and design have favorable internal properties that will lead to the development of a quality end product.

This should significantly help in reducing rework during and after implementation, as well as designing effective test plans and better project and resource planning.

Fortunately, the object-oriented approach naturally lends itself to an early assessment and evaluation. The approach provides the information needed to assess the quality of a design's, classes, structure, and relationships before they are committed to an implementation.

7. Range Of Constants

Based upon the design property to quality attribute relationship, the relative significance of individual design properties that influence a quality attribute is weighted proportionally so that the computed values of all quality attributes have the same range. A range of 0 to ±1 was selected for the computed values of the quality attributes.

A weighted value of +0.5 to +1 was used for the positive influences and a value of -0.5 to -1 was used for the negative influences. Therefore, the range of above attributes will be

TABLE 2
Quality Attributes-Range of Constant

Quality Attributes	Range
Reusability	+0.5 to +1
Flexibility	+0.5 to +1
Understandability	+0.5 to +1
Functionality	+0.5 to +1
Extendibility	-0.5 to -1
Effectiveness	-0.5 to -1

6. Attribute Dependency Relations

After the review of the ISO 9126 attributes-"functionality", "reliability", "efficiency", "usability", "maintainability" and "portability", these attributes were modified by a new set of attributes-"reusability", "flexibility", "understandability", "functionality", "extendibility" and "effectiveness" based on their behavior and effect on software quality as a whole.

6.1 Reusability

Reusability reflects the presence of object-oriented design characteristics that allow a design to be reapplied to a new problem without significant effort.

Now for software to have a better quality there should be high cohesion and low coupling.
Furthermore, cohesion can be stated as the degree to which methods within a class are related to one another and work together to provide well bounded behavior. Therefore, self-contained classes can easily be plugged in for reuse in another system since they do not depend on other classes to function.
Since self contained classes can easily be reused in other systems so,

Cohesion ∝ Reusability

Thus it proves the relation that

Software Quality ∝ Reusability

6.2 FLEXIBILTY

Flexibility can be stated as the characteristic that allows the incorporation of changes in a design. It is the ability of a design to be adapted to provide functionality related capabilities.
This is so because inheritance has the potential to adversely influence flexibility and understandability.
When an object is inherited from one class to another, it also means that the two classes are inter-dependent on eachother. That means they are coupled together. This gives rise to the relation,

This proves that,

Software Quality ∝ Flexibility

6.3 UNDERSTANDABILITY

Understandability is the property of the design that enables it to be easily learned and comprehended. This is directly related to the complexity of the design structure.
Understandability is related to coupling. Coupling is a measure of the strength of association established by a connection from one entity to another. Strong coupling complicates a system, thus making it less efficient. Hence in order to gain a better software quality coupling between objects(CBO) values must be less, so that classes are easy to understand, reuse and maintain.

This means,

Software Quality ∝ Understandibility

6.4 FUNCTIONALITY

Functionality is the responsibility assigned to the classes of a design, which are made available by the classes through their public interfaces.
The quality attributes-design property relationship table given under clearly states that cohesion is directly related to functionality.

TABLE 1
Quality Attributes—Design Property Relationships

	Reusability	Flexibility	Understandability	Functionality	Extendibility	Effectiveness
Design Size	↑			↑		
Hierarchies				↑		
Abstraction					↑	↑
Encapsulation		↑	↑			↑
Coupling						
Cohesion	↑		↑	↑		
Composition		↑				↑
Inheritance					↑	↑
Polymorphism		↑		↑	↑	↑
Messaging	↑			↑		
Complexity						

This means,

Software Quality ∝ Functionality

6.5 EXTENDIBILTY

Extendibility refers to the presence and usage of properties in an existing design that allows for the incorporation of new requirements in the design.
Therefore,

Software Quality ∝ _____1_____
Extendibility

6.6 EFFECTIVENESS

Effectiveness refers to a design's ability to achieve the desired functionality and behavior using object-oriented design concepts and techniques.
Therefore,

Software Quality ∝ _____1_____
Effectiveness

The above discussion can be briefed as,

- Software Quality depends directly on Reusability, Flexibility, Understandability and Functionality
- Extendibility and Effectiveness casts an indirect effect on the quality of software.

7. NEW METRICS DERIVED FOR SOFTWARE QUALITY

S.No	Metric	Name of Metric

1.	DC	Design size in Class
2.	NH	Number of Hierarchies
3.	ACA	Average count of ancestors
4.	DAM	Data access metrics
5.	CC	Class Coupling
6.	CAM	Cohesion among Methods in class
7.	MA	Measures of aggregation
8.	CPM	Count of polymorphic methods
9.	CIS	Class interface size
10.	TNM	Total number of methods
11.	MOA	Measure of Abstraction

Table 1

S.No	Metrics	Description
1.	DC	It counts the total number of classes
2.	NH	It counts the total number of class hierarchies
3.	ACA	It counts the total number of classes from which a class inherits the information
4.	DAM	It is the ratio of total number of protected/private attributes to the total number of attributes defined in the class
5.	CC	It counts the value through which different classes are directly related to
6.	CAM	It computes the value through which methods of class are related
7.	MA	It counts the number of data declarations whose types are user defined
8.	CPM	It counts the method of polymorphic behavior
9.	CIS	It counts the number of public methods in a class
10.	TNM	It counts the total number of methods in a class
11.	MOA	It is the ratio of number of the methods inherited by a class to the total number of methods accessible by member methods of this class

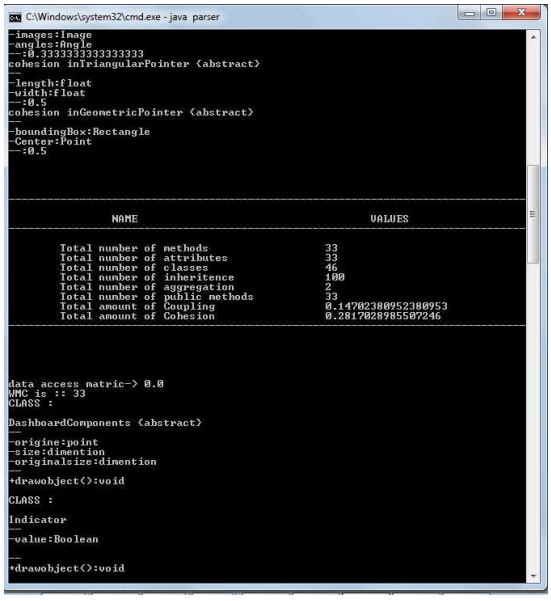
Table 2

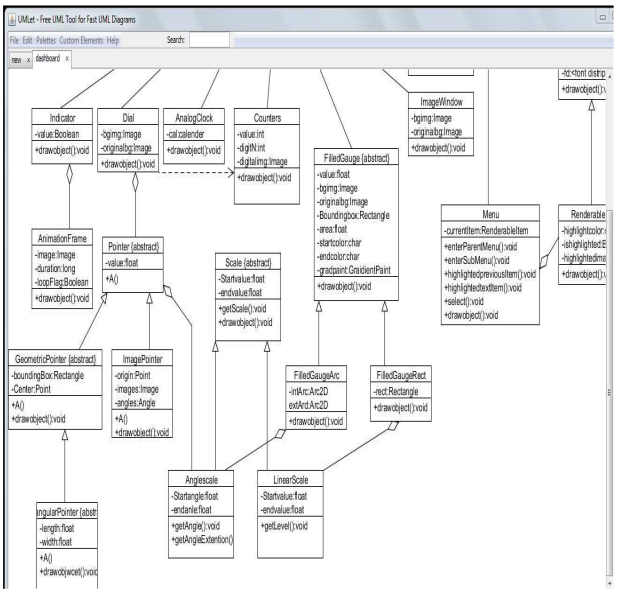
8. Description Of Standard Properties Used For Metrics

S.No	Design property	Description
1.	Design Size	Number of classes used in a design
2.	Hierarchies	Count of number of non inherited classes that have children in a design
3.	Abstraction	Measure of Generalization aspect of a design
4.	Encapsulation	Defined as the enclosing of data and behavior in single construct
5.	Coupling	Interdependency of objects on one another
6.	Cohesion	Relatedness of methods and attributes in a class
7.	Composition	It measures "part of","consist of","has" or "part whole" relationships
8.	Polymorphism	Ability to substitute objects whose interfaces matches for one another at run time

Table 3

9. Snapshots To Brief Up The Outputs Obtained





10. Results Obtained So Far Are As follows -

• As first approach we have used Open Source JAVA tool named UMLet for extraction of design metric information

• For rapidly drawing the UML diagrams with a pop-up-free and light weight user interface, we have used UMLet version 9.03 under GNU(Generic Public License.System design in the form of 09 UML diagrams (Class, Activity, Collaboration, and Sequence diagrams etc.) is given as input to the UMLet tool.

• UMLet creates .Uxf document along with the UML diagrams and from Uxf we get XML document. This XML document is used to extract the information for calculating quality metrics/measures.

• The quality related data obtained through UMLet, is saved in XML format which is the universal format for structured documents and data on the web.

• Values of the quality metrics/measures have been obtained from the sample UML diagrams of selected projects.

• With the help of the parser we have obtained the values of -

- Number of classes
- Number of methods
- Number of inheritance
- Number of attributes
- Number of aggregation
- Number of coupling
- Number of cohesion
- Number of relations
- Number of interacton frames
- Number of classes

11. Need For The Attribute Comparator

The Varity and complexity of software increased from day to day, the software quality assurance must be used to make a balance between quality and productivity. The
practice of applying software metrics to a software process and to a software product is a complex task that requires study and discipline and which brings knowledge of the status of the process and / or product of software in regards to the goals to achieve.
It is important for each software development project to define its specific meaning of
software quality during the planning phase. Such a definition contributes to the basis for setting objectives and practical measures of quality progress and determination of readiness for release to customers. Quality improvements affect operations performance in various ways, such as increasing revenue, reducing costs and improving productivity. Quality has been regarded as one of the major drivers of competitive strategy in every industry. The American National Standards Institute (ANSI) and American Society for Quality (ANQ) define quality as: *"The totality of features and characteristics of a product or service that impact its ability to satisfy given needs."*
This project deals with the measurement of the quality of object-oriented designs. The implemented system will allow the user to design any system using UML and then provide quality readings from different perspectives of the system. The system will provide the user metric readings that provide different views of the quality characteristics of the system in question. The user will use the tool to identify potential problem areas and fix them before the project goes into implementation stage.
The earlier that a fault is detected and removed, the easier it

is to fix. Object oriented metrics focus on the combination of functions and data as an integrated object. Object oriented paradigm substantially improves productivity due to the effect of reuse. Requirement specifications, designs and test plans are all artifacts that could potentially be fully or partially be reused in different projects.

12. Research Problem

The research problem is that the existing systems have quality assessment only at the completion time of that project; we don't have any system which give us this opportunity to measure the quality at the beginning stage or at intermediate stages. So the work is that to develop such metrics or tools, which make these things feasible.

13. Objective Of Research Work

1) Analysis of existing measures and metrics for design strength and quality of object oriented software.
2) Identification of limitations and defects in the existing measures.
3) Development of new measures for design strength and quality.
4) Development of new methods for assessment of design strength and quality of object oriented software.
5) Evaluation and verification of the new measures and methods.

14. Hierarchical Model

The ISO 9126 attributes-"functionality", "reliability", "efficiency", "usability", "maintainability" and "portability" were selected as the initial set of quality attributes in the QMOOD model. The validation of the QMOOD quality model was carried out at two levels: validation of the individual quality attributes effectiveness and validation of the overall software quality estimation.
Since the quality attributes in QMOOD are important development characteristics of most software systems, they should indicate desirable trends that are normally expected of reusable, flexible, extendible and effective frameworks applications. The hierarchical model, QMOOD, for assessment of high level design quality attributes in object-oriented designs has been developed and validated using structural and functional information from object-oriented designs of several releases of two large and popular commercial framework systems. The model's ability to estimate overall design quality from design information has also been demonstrated using several functionally equivalent projects where the overall quality estimate computed by the model had statistically significant correlation with the assessment of overall project quality characteristics determined by independent evaluators.
Models of this type can be effectively used in monitoring the quality of software product. While the issues of quality assessment still remain somewhat elusive, the QMOOD model also offers a tool that will allow various models and approaches to be quickly evaluated on real-world projects, helping accelerate the convergence of differing quality views.

15. Empirical Study Of Object Oriented Metrics

A key element of any engineering process is measurement. We use measurements to assess the quality of the engineered product or the process used to build it. We attempt to derive a set of indirect measures that lead to metrics that provide an indication of the quality of some representation of software. Realizing the importance of software metrics, numbers of metrics have been defined for software. These metrics try to capture different aspects of software product and its process. We investigate 22 metrics proposed by various researchers. The metrics are first defined and then explained using practical applications. They are applied on standard projects on the basis of which descriptive statistics, principal component analysis and correlation analysis is presented. Finally review of the empirical study concerning chosen metrics and subset of these measures that provide sufficient information is given and metrics providing overlapping information are excluded from the set.
There are only 12.5% of the total classes that have high coupling metrics values. There are 4.1% classes with deep hierarchy. Since earlier empirical studies suggests that classes with more coupling and deep hierarchy are fault prone, the identified classes (12.5%) must be thoroughly checked during testing. Theoretical analysis of these metrics suggests that out of 14 class level metrics, 6 metrics (NOA, NOM, MPC, DAC, LCOM and LCC).
The research work is focused at the object oriented metrics for improving the quality of the system/software. So in software, we need to identify the necessary metrics that provide useful information, otherwise the managers will be

lost into so many numbers and the purpose of metrics would be lost. As the number of metrics available in literature is large, it becomes a tedious process to understand the computation of these metrics and draw inferences from their values.

16. Summary Of Research Work

The research work is focused at the object oriented metrics for improving the quality of the system/software. We have examined various existing metrics and tried to find out the limitations/disadvantages in those. Limitations of existing metrics are as follows.

1. Lack of measure of reuse (MOOD)
2. Polymorphism (MOOD)
3. External coupling (MOOD)
4. It does not provide a complete quality model suitable for use in search-based refactoring (MOOD)
5. Lack of effective metrics definitions (QMOOD)
6. Metrics are defined in natural language, and are in some cases ambiguous (QMOOD)
7. In order to implement the metrics for replicable studies it is necessary to define them more precisely (QMOOD)
8. No dependence on high level characteristics (MOOSE)
9. No explicit quality characteristics (MOOSE)
10. Metrics suite focuses on object oriented design measures except encapsulation (CK)
11. The classes with high WMC values are having high LCOM and RFC values (CK)
12. High values for WMC, LCOM and CBO increases complexity of the system (CK)

So in software, we need to identify the necessary metrics that provide useful information, otherwise the managers will be lost into so many numbers and the purpose of metrics would be lost. As the number of metrics available in literature is large, it becomes a tedious process to understand the computation of these metrics and draw inferences from their values.

Realizing the importance of software metrics, number of metrics has been defined for software. These metrics try to capture different aspects of software product and its process. Some of the metrics also try to capture the same aspect of software e.g., there are number of metrics to measure the coupling between different classes. Software developers need to explicitly state the relation between the different metrics measuring the same aspect of software.

17. Identifying Design Quality Attributes

After the review of the ISO 9126 attributes-"functionality", "reliability", "efficiency", "usability", "maintainability" and "portability", these attributes were modified by a new set of attributes-"reusability", "flexibility", "understandability", "functionality", "extendibility" and "effectiveness" based on their behavior and effect on software quality as a whole.

TABLE 1

Quality Attributes	Definitions
Reusability	Reflects the presence of object oriented design characteristics that allow a design to be reapplied to a new problem without significant effort.
Flexibility	Characteristics that allow the incorporation of changes in a design. The ability of a design to be adapted to provide functionally related capabilities.
Understandability	The properties of the design that enable it to be easily learned and comprehended. This directly relates to the complexity of the design structure.
Functionality	The responsibilities assigned to the classes of a design, which are made available by the classes through their public interfaces.
Extendibility	Refers to the presence and usage of properties in an existing design that allow for the incorporation of new requirements in the design.
Effectiveness	This refers to a design's ability to achieve the desired functionality and behavior using object oriented design concepts and techniques.

18. Identifying Object Oriented Design Metrics

A survey of existing design metrics revealed that there are several metrics that can be modified and used in the assessment of some design properties, such as abstraction, messaging, and inheritance. However, there are several other design properties, such as encapsulation and composition, for which no object oriented design metrics exist. Also, while metrics to assess complexity, cohesion, and coupling have already been defined, these metrics require a nearly complete implementation of classes before they can be calculated and, therefore, cannot be used in QMOOD. This led to the definition of five new metrics, the data access metric(DAM), the direct class coupling metric(DCC), the cohesion among methods of class metric(CAM), the measure of aggregation metric(MOA), and the measure of functional abstraction metric(MFA) that could be calculated from design information only. The complete table is given below.

TABLE 2

Metric	Name (Design Property)
DSC	Design Size in Classes
NOH	Number of Hierarchies
ANA	Average Number Of Ancestors (Abstraction)
DAM	Data Access Metric (Encapsulation)
DCC	Direct Class Coupling (Coupling)
CAM	Cohesion Among Methods of Class (Cohesion)
MOA	Measure of Aggregation (Composition)
MFA	Measure of Functional Abstraction (Inheritance)
NOP	Number of Polymorphic Methods (Polymorphism)
CIS	Class Interface Size (Messaging)
NOM	Number of Methods (Complexity)

19. Problem Formulation

- We have identified an optimal set of structural and reuse Metrics for assessment of design Quality.
- Two approaches have been used for the assessment of Quality.
- The first approach takes UML diagrams of a Object Oriented Software as input and calculates Design Quality from that.
- In the second approach, code base has been taken for assessment of the product quality.
- Design quality and Product quality have been assessed for three category of software.
- The categories include software with well established and known quality i.e. Good, Bad, and Medium.
- Under Good category two Projects named Taming Java Threads which is an open source threading library, and JDOM that provides a way to represent XML document have been taken.
- Medium category includes Bonforum: An open source chat application.
- In Bad category we have taken a few students projects and Eview Applet.

20. Work Completed Earlier

- A survey paper "Object Oriented Metrics: Features and Defects in S/W Design & Quality" has been published in International Journal of Computing and Applications, Vol. 6, No. 2, July-December 2011, pp. 139-141.
- From the survey of different metrics set proposed by Software Engineers/Researchers, we have dentified an optimal set of structural and reuse Metrics for the assessment of the design Quality. This optimal set of Metrics under various categories is as follows-

1. **Size metrics**
   - Number of Attributes per class (NOA)
   - Number of Methods per class (NOM)
   - Weighted Method per class (WMC)

2. **Coupling Metrics**
   - Coupling between Objects (CBO)
   - Data Abstraction Coupling (DAC)

3. **Cohesion Metrics**
   - Lack of cohesion in Methods (LCOM)
   - Loose class Cohesion (LCC)
   - Information flow based Cohesion (ICH)

4. **Inheritance Metrics**
   - Depth of Inheritance tree (DIT)
   - Number of Children (NOC)

5. **Information Hiding Metrics**
   - Method hiding Factor (MHF)
   - Attribute hiding Factor (AHF)

6. **Polymorphism Metrics**
   - Polymorphism Factor (PF)

7. **Reuse Metrics**
   - Reuse Ratio (U)

As first approach we have used Open Source JAVA tool named UMLet for extraction of design metric information.

For rapidly drawing the UML diagrams with a pop-up-free and light weight user interface, we have used UMLet version 9.03 under GNU (Generic Public License).It supports editing of all UML diagrams in .uxf format and thus provides a solid base to which quality measurement capabilities can be added. System design in the form of 09 UML diagrams (Class, Activity, Collaboration, and Sequence diagrams etc.) is given as input to the UMLet tool.

UMLet creates .Uxf document along with the UML diagrams and from Uxf we get XML document. This XML document is used to extract the information for calculating quality metrics/measures.

The quality related data obtained through UMLet, is saved in XML format which is the universal format for structured documents and data on the web. For processing of the XML document obtained from the UML diagrams we have used functionalities of Document Object Model (DOM) and SAX parsers with writing appropriate codes.

Values of the quality metrics/measures have been obtained from the sample UML diagrams of selected projects.

http://www.youtube.com/watch?v=SoYlFZngbT8

21. Three Years (2010-2013) Progress (PERT CHART)

1. Literature Survey
2. Analysis of existing measures & metrics
3. Identification of limitations and defects
4. Conceptualization of idea & problem formulation
5. Development of new measures
6. Development of new models for design strength and quality assessment
7. Evaluation & verification of models & measures
8. Writing of thesis