A SURVEY ON MEASURING REUSABILITY, MODULARITY AND EXTENSIBILITY ON COUPLING AND COHESION

[1]G.Maheshwari, [2]Dr.K.Chitra
[1]Research Scholar, [2]Assistant Professor
[1]Department of Computer Science, [2]Department of Computer Science,
[1]Madurai Kamaraj University, [2]Goverment Arts College, Melur ,Madurai, India.

Abstract: This paper presents a research study on how to enhance performance of a system by using extensibility, reusability and modularity. It also describes how they support to increase the system's quality by applying reusability concept and how the changes are adapted by using modularity and extensibility. It also explains elaborately how to measure reusability, modularity and extensibility in coupling and cohesion and how to increase Cohesion and decrease Coupling.

Index Terms – Coupling, Cohesion, reusability, modularity, extensibility.

I. INTRODUCTION

Nowadays the size of data is overflowing due to the increasing number in industries. To use the data sets effectively , we have to identify the data sets which are independent and non-independent, data sets which should be combined together and which should not be combined within a module and maintain them according to it. Coupling and cohesion identify this. A design is said to be efficient if it has High Cohesion and Low Coupling . According to the user requirement the architectural design must be modified periodically. If this is done, it may increase the size of the project, development cost and time will also be increased. So to avoid this we need reusability , which will decrease the size, time and cost of the project. Also, we need modularity and extensibility, to combine the modules which are dependent and which need to interact , to share the data, classes and data structures and adapt to the future changes we need extensibility, to extend the design or to add the number of modules without changing the internal structure of the architecture. But anyhow these changes will sometimes increase the cost and time and also error rates. To overcome this problem we have to measure the reusability, modularity and extensibility. If the reusability increase, the effectiveness also increase.

A. EXTENSIBILITY

It is an ability of extending a system by adding new functionality with minimal effects on its internal structure and data flow. The quality attributes in order to achieve extensibility must be able to identify the problems, identify the design principles suitable to extend the software. The attributes are,

- Modifiability
- Maintainability
- Scalability

After identifying these attributes, the design principles that support modifiability, Maintainability and scalability need to be identified. One could argue that there are more quality attributes. There are so many quality attributes that support extensibility. But these are the primary attributes that influence extensibility.

Modifiability:

A system is said to be modifiable if the minimum number of changes will affect minimum number of elements . to make this easy the codes must be simple and easy to understand. If the coding becomes difficult, one can't support modifiability. After this the basic design principles should be identified.

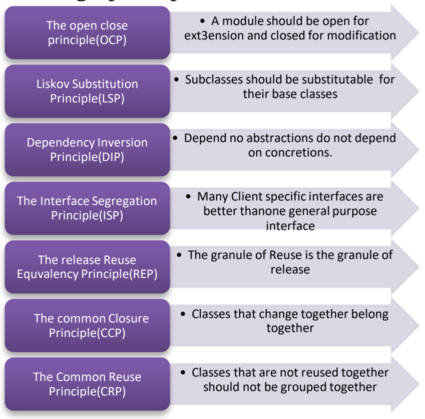
The open close principle(OCP)
• A module should be open for ext3ension and closed for modification
Liskov Substitution Principle(LSP)
• Subclasses should be substitutable for their base classes
Dependency Inversion Principle(DIP)
• Depend no abstractions do not depend on concretions.
The Interface Segregation Principle(ISP)
• Many Client specific interfaces are better thanone general purpose interface
The release Reuse Equvalency Principle(REP)
• The granule of Reuse is the granule of release
The common Closure Principle(CCP)
• Classes that change together belong together
The Common Reuse Principle(CRP)
• Classes that are not reused together should not be grouped together

Figure 1 Design Principles that support Modifiability

Maintainability:
Maintainability is the ability to correct errors or adapt changes if changes are made to satisfy new requirements. The attributes that encourabe maintainability are correctiveness, availability and the factors that are affected by maintainability are corrective maintanence and preventive maintenance.

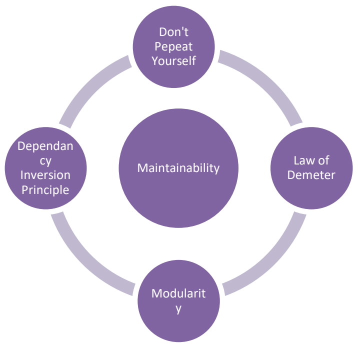

Figure 2 Design principles that support Maintainability

Scalability:
Scalability is the level at which a process can grow and increased demand can be managed. It is more adaptable to the changing needs or demands of its users or clients. The principles of Scalability are Fault Tolerance, Stateless, Parallelization, Asynchronous, Idempotent, Partitioning.
Design principles for Scalability

- Avoid the single point of failure
- Scale horizontally, not vertically
- Push work as far away from the core as possible
- API first
- Cache everything , always
- Provide as fresh as needed data
- Design for maintenance and Automation
- Asynchronous rather than synchronous

B. REUSABILITY
The reusability improves the quality and production of software. It is essential for quality assurance. Some software quality parameters are quantifiable, but it is difficult to measure and quantify a non functional parameter like reusability.
Factors for reusability estimation

- Coupling
- Cohesion
- Interface Complexity
- Security
- Response time
- Statelessness

Design for reuse : The reusability is measured by identifying the size of the component, required effort, productivity and parameters. Reusable components have more parameters than the single time used components. But the productivity is lower for the reusable components than the single time -used components. During the development time, more time is spent on writing code than designing components. Based on the frequency of use, the design principles are ranked. A feedback report is also prepared and the reasons why the reuse design is used and not used are identified. The results were correlated and it shows that the reuse design principles were independent of each other.
Design with reuse : Design principles: like clarity, well-defined interface, generality, understandability, and separate concepts from content increased the ease of reuse but variability and commonality decreased the ease of reuse. Testing the components before integrating them into a system have no relationship with the reusability of components. A content analysis of the feedback is presented identifying the challenges of components that were not easy to reuse. Features that make a component's reusability easy were also identified. The identified factors within the model are: a set of reuse design principles like well-defined interface, clarity and understandability, commonality and variability analysis, and generality and component testing,size of component.

There are many software development technologies to support reusability. They are,

1. **Module Oriented Approach (MOA)**

It is an approach to develop the software, which depends on perception of procedure call, which is also known as subroutine, method and function. It involves complexity and requires significant amount of reusability. Some restrictions and characteristics of Module Oriented Approach are,

- It provides reusability of code.
- It is robustly modular than structure.
- Unable to data binding with operations.

To overcome these limitations, Object Oriented Approach (OOA) is introduced.

2. Object Oriented Approach (OOA)

OOA splits problem into objects that abstract behavior and data into a single unit. The concept of inheritance, modularity, polymorphism and encapsulation are introduced. Some restrictions and characteristics of Object Oriented Approach are:

- It rely on data relatively than procedure.
- Allows message passing and functions through interaction of objects.
- It provides the concept of classes, encapsulation and hides implementation details.
- It is an easy way to add new data and functions, whenever required.
- Ability to provide polymorphism and inheritances.

With large size projects, it has been observed that for OOA it is difficult to separate readability, security and modifiability. To overcome the limitations of object oriented approach (OOA), Aspect Oriented (AO) approach is introduced.

3. Component Based Software Development (CBSD)

This is widely accepted in terms of cost effectiveness Software development is a process of integrating composed diverse software components. The main problem is the reclamation and selection of appropriate software components from a huge variety of reusable components. The major advantages are :

- Provides in-time and high quality solutions for new software development.
- It also offers high productivity, flexibility and quality through reusability, replace-ability, efficient maintainability, and scale ability.
- It also reduces development time for new system.

4. Aspect Oriented Software Development (AOSD)

It combines Module Oriented Approach (MOA) and Object Oriented Approach (OOA). Some characteristics are,

- It is most popular languages that used in AOP.
- Supports characteristics such as point cuts, advice, joint points and introduction .
- Defines new constructor, maintain in favour of modular accomplishment for crosscutting concerns.
- Provides consistency checking, protocol management, synchronization and others.

5. **Cohesion**

Cohesion defines the level of dependability among elements within a module. The greater the cohesion, better the program design. Highly cohesive modules are considered to be highly reusable. It is easy to maintain high cohesive modules of a software system

6. Coupling

Coupling defines the level of dependability between the elements of different modules of software. The level at which there is interaction and interference existing among the modules is described in this . Lower the coupling, higher the reusability. It exhibits a supplier client server relationship within the design elements . Meausurements in coupling are,

i. Intra-modular Coupling density (ICD) : It defines the connection between coupling and cohesion among the elements of modules in a module based software system

ii. Import Coupling: Measures the count of services taken from other modules by a dependent module. \Import coupling which is denoted by Iv, represents the value of the extent to which packages is dependent on other design elements.

iii. Export Coupling: Measures the services which offered to another dependent module by a module. For the component to be reusable export should be more than import.

C. MODULARITY

It is defined as Linking so many individual modules to form an absolute system. Modularization is based on functions which are related, implementation considerations, data links, or other criteria. In modularization changes made in one module should not affect other modules. If software is divided indefinitely, the effort required to develop it will become small. Figure 3 shows that the effort(cost) to develop an individual software module does not decrease , as the total number of modules increases. D display technic applications.

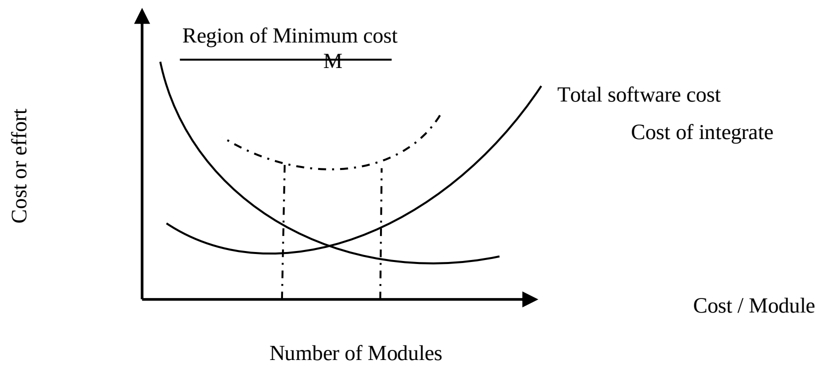

Figure 3 Modularity and Software Cost

II. Previous Works

In architectural design software quality plays a main role. In measuring quality, we need to consider the non-functional and functional parameters. For example, For calculating the Design Structure Quality Index(DSQI) which is used in evaluating object oriented software packages, the steps are,

1. **For evaluating DSQI, the values given below should be identified.**

S1 = total number of modules defined in the program architecture
S2=number of modules whose correct function depends on source of data input or that produce data to be used elsewhere
S3=the number of modules whose function depends on pre processing
S4=number of databases
S5=total number of unique database items
S6=number of database segments
S7=number of modules with a single entry and exit

2.Once these values are determined , the intermediate values like,

If discrete methods are used for developing architectural design then $D_1= 1$, else $D_1 = 0$
Module independence D2=1 - (S2 / S1)
Modules not depend on prior processing D3=1 – (S3 / S1)
Database size D4=1-(S5 / S4)
Database compartmentalization D5= (S6 / S4)
Module entry / exit characteristic D6 = 1 – (S7 / S1)

3. Once all the intermediate values are calculated, OSQI is calculated by the following equation.

$DSQI = \sum W_i D_i$

Where, $\sum W_i = 1$ (W_i is the weighting of the importance of intermediate values).

- **Cohesion metrics:** Cohesiveness of a module can be indicated by the definitions of the following concepts and measures.
- **Data slice:** it is a walking backward through a module, which looks for values of data that affect the state of the module as the walk starts
- **Data tokens:** set of variables used in a module
- **Glue tokens:** set of data tokens, which lies on one or more data slice
- **Superglue tokens:** Tokens, which are present in every data slice in the module
- **Stickiness:** Stickiness of the glue token, which depends on the number of data slices that it binds.
- **Coupling Metrics:** The degree to which a module is connected to other modules, global data and the outside environment. A metric for module coupling has been proposed, which includes data and control flow coupling, global coupling, and environmental coupling.

Measures defined for data and control flow coupling are listed below.
d_i = total number of input data parameters
c_i = total number of input control parameters
d_o= total number of output data parameters
c_o= total number of output control parameters
$1§ Measures defined for global coupling are listed below.
g_d= number of global variables utilized as data
g_c = number of global variables utilized as control
$1§ Measures defined for environmental coupling are listed below.

w = number of modules called
r = number of modules calling the modules under consideration
By using the above mentioned measures, module-coupling indicator (m_c) is calculated by using the following equation.

$m_c = K/M$

Where

K = proportionality constant
$M = d_i + (a*c_i) + d_o + (b*c_o) + g_d + (c*g_c) + w + r$.

Note that K, a, b, and c are empirically derived. The values of m_c and overall module coupling are inversely proportional to each other. In other words, as the value of m_c increases, the overall module coupling decreases.

To calculate DSQI,

C

S1	24	total number of modules defined in the program architecture
S2	4	number of modules whose correct function depends on source of data input or that produce data to be used
S3	4	the number of modules whose function depends on pre processing
S4	22	number of databases
S5	5	total number of unique database items
S6	80	number of database segments
S7	0	number of modules with a single entry and exit

D!	1	Program structure
D2	8333	Module independence
D3	8333	Modules not dependent on prior processing
D4	77273	Data base size
D5	-2.63	Database compartmentalization
D6	1.000	Module entrance and exit characteriustics

With this intermediate values DSQI is computed and the relative weights also Wi is also calculated. With this the level of coupling and cohesion also determined.

III. Conclusion

The quality of a software can be measured based on functional and non-functional attributes. There are so amny quality measures and metrics for that. But they are system and user dependent. The ultimate aim is to develop a quality software with reduced cost and time. For that reusability, modularity and extensibility plays main role. In this paper some existing metrics to measure these attributes in coupling and cohesion are given. In future metrics which are independent of system and user, with minimal cost and maximum performance need to be developed.

REFERENCES

[1] Amjad Hudaib, Ammar Huneiti, Islam Othman (2016) Software Reusability Classification and Predication Using Self-OrganizingvMap (SOM) Scientific Research Publishing, http://creativecommons.org/licenses/by/4.0/

[2] P. K. Singh, O. P. Sangwan, A. Pratap, A. P. Singh, "A Quantitative Evaluation of reusability for Aspect Oriented Software using Multi-criteria Decision Making Approach" published in world Applied Sciences Journal, Volume 30,Issue 12, Pages 1966-76, 2014.

[3] Maggo, S. and Gupta, C. (2014) A Machine Learning Based Efficient Software Reusability Prediction Model for Java Based Object Oriented Software. *International Journal of Information Technology and Computer Science* (*IJITCS*), http://dx.doi.org/10.5815/ijitcs.2014.02.01

[4]Goel, B.M. and Bhatia, P.K. (2013) Analysis of Reusability of Object-Oriented Systems Using Object-Oriented Metrics.*ACM SIGSOFT Software Engineering Notes*, **38**, 1-5. http://dx.doi.org/10.1145/2492248.2492264

[5] Singh, G. (2013) Metrics for Measuring the Quality of Object-Oriented Software. *ACM SIGSOFT Software Engineering Notes*, **38**, 1-5. http://dx.doi.org/10.1145/2507288.2507311

[6] Rotaru, O.P. and Dobre, M. (2005) Reusability Metrics for Software Components. *Proceedings of the* 3rd *ACS*/*IEEE International Conference on Computer Systems and Applications*, Cairo, 2005, 24-29. http://dx.doi.org/10.1109/aiccsa.2005.1387023

JETIR