METRICS FORMATION ON THE BASIS OF COUPLING AND COHESION FOR SOFTWARE REUSABILITY

Er. Deepshikha Chhabra
Assistant Professor
Chandigarh University
deepshikha.cse@cumail.in

Er. Priya Batta
Assistant Professor
Chandigarh University
priya.cse@cumail.in

ABSTRACT

The ability to reuse depends in an essential way on the ability to identify commonalities among existing parts and to develop bigger things from smaller parts. The cohesion means how much one part of module is depend on another inside module where as coupling means how much one module depend on another module. In our approach we are determining the reusable components of a software system and enhancing the accuracy of the methods for determining them.We have also discussed a proposed approach how we can manage to determine the reusable components on the basis of reusability metrics.

Keywords: Coupling, Cohesion, Metrics.

1. INTRODUCTION

Software reusability means using the assets which already exist in one or other form in the field of software engineering and computer science during the process of software development. Assets may include test suites, documentation, code which can be considered as products as well as by products of the processes that occur during software development [1]. The ability to reuse depends in an essential way on the ability to identify commonalities among existing parts and to develop bigger things from smaller parts. Reusability is often considered as a required feature of platform software [2]. Reusability involves aspects to software development which are not needed to be considered where reusability is not required. Module based software development is one of the best way to develop the big project. In Module based development the project can easily divide in multiple developers so that development of project gets faster and bugs can find at initial stage. One of the major advantages of module development is reusability of module means the module which can later be used in another project if the sum of requirements of the project would be remains same. Now a day's the process of developing the software is going through change that is making the use of the modules which are already running in various source software which are open and close and this emerging change is extreme. Fitting those best components after they had been extracted into ongoing component based development software brings big challenges to run that software without errors and produce desired outcomes [3]. Because of Reusability module many developer make the module and test it robustness and then sell online by name of plug-in. In our research we are going to develop an Algorithm which will find the reusability in project based on cohesion and coupling. The cohesion means how much one part of module is depend on another inside module where as coupling means how much one module depend on another module. In our

approach we are determining the reusable components of a software system and enhancing the accuracy of the methods for determining them. We are using genetic algorithm and fuzzy c mean algorithm to find the cohesion and coupling between the components of a software system on basis of which reusability of the components is determined. Already existing software projects are collected and technique was applied on them to determine the reusable components of those projects and f-measure value is computed for comparison with previous techniques.

1.1 COHESION:

Cohesion refers to the degree of strength with which different elements in a module are related to each other is the degree to which the elements of a module belong together. So the strength of the belongingness between various elements within a module is given by cohesion. For example, functionalities are strongly related in those systems that are very cohesive. In case object-oriented programming, the class is said to be highly cohesive if those methods which are serving that class share some common functionalities[16]. In a system which is highly cohesive reusability and readability of the code is increased along with the fact that complexity continues to remain difficult to manage is remaining manageable.

Cohesion is said to be increased in case:

- The functions in the class and the function that are accessed by using the methods of that class have many commonalities.
- Unrelated sets of data and coarsely grained data are avoided and methods carry out a lesser number of activities that are related.

Advantages of strong cohesion are:

- Reduces complexity of the module as they are simpler and have few operations.
- Increase in the reusability of reusability is increased because developers of the new applications will find the required components easily within some number of operations of operations that are cohesive and are present in the module.
- Increases the maintainability of the system, as changes that are logical n the area influence lesser number of modules and changes that are done in one module does require less numerous adjustments in every other module.

While a module, on a fundamental level, can have impeccable union by just being comprised of a nuclear or single component and having only one function,. Thus a module with a single has an element which either is too narrow or too complicated to accomplish task and hence is tightly coupled to other modules. cohesion is thus considered to be balanced with both unit complexity and coupling.

1.2 COUPLING:

Coupling is said to be the manner and degree of interdependence between different modules of a software. It measures of closely two routines or modules are connected and is also referred as strength of the relationships between different modules[5].

Disadvantages of coupling:

Systems that are tightly coupled exhibits the following characteristics that are often seen as disadvantages:

1. A change in one module forces the changes in other modules also.
2. due to the increased dependency between the module assembling of modules may involve more effort

3. it might be harder to reuse a particular module because the dependent modules are required be included.

Functional design is an approach to decrease the coupling that seeks to limit the responsibilities of modules along with its functionality. Coupling between two classes say, **A** and **B** will increase if:

- A has an attribute which refers to B.
- an object B is called on service by A.
- A contains a method which references B (via return type or parameter).
- A implements class B.

A relationship that contains one module that interacts with other modules through a stable and simple interface and is not required to be concerned to other module's internal implementation is referred as low coupling. Systems like CORBA or COM which allow different objects to interact with each other without requiring to know anything about the implementation of other objects. These systems also allow the objects to communicate with objects that are written in some other languages.

2. LITERATURE SCRUTINIZING

Mohammad Zulkernine et.al [2] had related the complexity, cohesion and coupling to the security failures of the software when they are in their operational stage. Though it is hard is to find out the vulnerabilities until they manifest themselves in the form of security failures at operational phases of software, since security concerns for the most part remain not tended to or not known sufficiently early amid the product improvement life cycle. Numerous studies have uncovered the intricacy, union and coupling as the vital markers of the of the software architecture and which is a standout amongst the most vital and early design choices that impacts the nature of the product at long last . Probably all theories measurements had been effectively utilized to demonstrate the issues in the product as a rule but no well defined guidelines were available about using these metric for the purpose of determining the vulnerabilities. They proposed how CCC metric can be used to indicate vulnerabilities and thus made this metric to aid in the conception of the architecture which is even more secure which further promised a secure, design, code and eventually a better software.

Jeff Offutt[12] et.al presented a static analysis tool to determine the coupling in order to reduce the extent of coupling among the java classes and make the software highly cohesive . This static analysis tool was made to detect the coupling based on the source code. They discussed coupling in the software on the basis of object-oriented relationships between the classes with an attention on the sorts of couplings that are not distinguished till the execution is finished and proposed an instrument for static investigation which measured the couplings among classes in Java bundles. The coupling estimation depended on source code was utilized to gauge the coupling among the classes in a bundle of java, which was quantitative and more exact than the measures which were connected already yet alongside that there was the inconvenience of not being accessible before usage and henceforth it didn't function admirably for some predictive endeavors.

Jehad Al Dallal et.al[3] discussed several metrics that had been proposed in order to measure how much the class members related .and use of Connectivity-based class cohesion metrics for measuring the degree with which class members are connected to each other. They proposed new class cohesion metric with discriminative power higher than cohesion metrics which had already existed. The probability of cohesion metric

incorrectly determining the classes with different connectivity patterns to be cohesively equal was determines using discriminative –power study. the comparison of connectivity-based metrics, including PCC, to other non-connectivity-based cohesion metrics in terms of explaining a fault which is present from a statistical standpoint better was done through the fault investigation study . The results the theoretically demonstrated that PCCC fulfills the key union properties. The outcomes additionally showed that PCCC measures those cohesion viewpoints that were not dictated by different measurements, and it is obviously better than other availability based measurements yet in contrast with some other non-connectivity based cohesion metric it is poor as far as its capacity for anticipating faulty classes.

Sunju Oh et.al[14] worked on the strategy to quantify the coupling and attachment for ontology modules utilizing one union metric and two coupling measurements, Along with this the new coupling measurements additionally contributed in checking the consistency between the ontology modules. Different surely understood check structures and exact tests for supplementing the past examinations were utilized to accept the proposed measurements. The outcomes offered the ontology engineers important criteria to assess cosmology and helped ontology users to choose qualifying cosmology modules.

Kailash Patidar et.al[15] presented measurements of the object after that finding the coupling and cohesion between different objects, the association between numbers of classes is then measured, check the different sort of dependencies like direct dependencies, indirect dependencies, number of out and in metrics in object oriented programming, I/O dependencies. Before their work many coupling and cohesion measures had been introduced in numerous surveys in order to identify and measure the complexity in design of object oriented systems. Many of metrics had been built and proposed to measure of object-oriented software properties such as size, cohesion coupling, inheritance,. The coupling is considered an important aspect in the evaluation of maintainability and reusability of components or services.

Vangipuram Radhakrishna et al.[8] : In their research, they instigated a generalized approach to cluster a given set of documents or components using a new similarity measure function known as XOR function in order to find degree of similarity among the software components . They constructed a matrix known as similarity matrix of n-1 by n order for any given set of components . They designed and defined the algorithm for clustering of components that takes input as similarity matrix and set of clusters formed dynamically in comparison to other clustering algorithm which predefines the cluster count and forms the output.

Yannis Smaragdakis et.al[10] They presented a method to map the participating suites of classes specifically into exemplified executions of c++ . Their technique was a change over the methodology utilized by Van Hilst and Not kin for executing the joint e plans based on collaboration and a heading a stage towards more reusable parts of item situated frameworks. In their work paper proposed a technique so as to guide the configuration elements into embodied article situated usage. In that way they interpreted the he reusability points of interest of abnormal state plan specifically to the execution. Their usage required just the elements of standard programming dialect like legacy and parameterization. Their work depended on a refined utilization of legacy parameterization and settled classes keeping in mind the end goal to accomplish epitomes of "substantial scale" object oriented segments. They

actualized a product venture which was medium-sized and was created in Java utilizing their segment approach: The compiler-pre compiler generator proposed by ali epitomized dialect expansions in type of collaboration components. The components were of generous size and they framed the fundamentals of the language expansion system given by ali. In all, their technique yielded a significant reusability and modularity advantages in an elegant manner.

4. DETERMINATION OF REUSABLE COMPONENTS PROPOSED APPROACH

We will select an existing software project work is done determining the relationships and dependencies among the elements of the modules within a package and across the packages and values of the import coupling and export coupling are found out based on which the component of the existing software projects are considered to be reusable and not reusable. so here we have developed a system using the proposed methodology to which the software projects based on java are passed and it shows all type of dependencies between all the elements of different components along with their export and import coupling value and gives the names of the components that are reusable. It also provides the measure of the accuracy of the algorithm in terms of f-measure and time taken by the algorithm to find out the cohesion and coupling for any given project.Different factors of comparison will be taken to compare our results with the previous techniques which includes f-measure, PCM and PLC software.

5.CONCLUSION

Software reusability means using the assets which already exist in one or other form in the field of software engineering . We have surveyed a number of articles for our reference to pursue in software reusability. Here we have scrutinized the detailed study by various authors for developing software reusability metrics on the basis of cohesion and coupling. We have also discussed a proposed approach how we can manage to determine the reusable components on the basis of reusability metrics.

References:

[1] Wen-Hwa Liao et al., "GRID: A fully Location Aware Routing Protocols for MANET", Telecommunications Systems, pp. 1548-1557 (2001).

[2] Mohammad Zulkernine, et al. "Using complexity, coupling, and cohesion metrics as early indicators of vulnerabilities" Journal of Systems Architecture Volume 57, Issue 3,(2011), 294–313.

[3] Jehad Al Dallal et al. "An object-oriented high-level design-based class cohesion metric" Information and Software Technology, Volume 52, Issue 12, (2010),1346–1361.

[4] Bhullar, Rohit K., Lokesh Pawar, and Vijay Kumar. "A novel prime numbers based hashing technique for minimizing collisions." *Next Generation Computing Technologies (NGCT), 2016 2nd International Conference on*. IEEE, 2016.

[5] S. Arora, et al. "Novel stress calculation in parallel processor systems using buddy approach with enhanced short term CPU scheduling." *Communication and Computing Systems*. CRC Press, 2016. 663-672.

[6] Verma, Shiv Kumar, Manpreet Kaur, and Rohit Kumar. "Hybrid Image Fusion Algorithm Using Laplacian Pyramid and PCA Method." *Proceedings of the Second International Conference on Information and Communication Technology for Competitive Strategies*. ACM, 2016.

[7] Rohit Kumar "Pragmatic Implementation of Power Optimization in Wireless Sensor Networks " *MATRIX Academic International Online Journal of Engineering and Technology* 1.2 (2016): 1-6.

[8] Vangipuram Radhakrishna et al. "Document Clustering Using Hybrid XOR Similarity Function for Efficient Software Component Reuse" Procedia Computer Science Volume 17, (2013),121-128.

[9] Rohit Kumar " Advanced Tools and Techniques for Re-configurable Processor Architectures" *MATRIX Academic International Online Journal of Engineering and Technology* 1.2 (2016): 1-6.

[10] Yannis Smaragdakis et al. "Object-Oriented Frameworks and Product Lines" Software Product Lines The Springer International Series in Engineering and Computer Science Volume 576 ,(2000) 227-247.

[11] Bhullar, Rohit K., et al. "Intelligent stress calculation and scheduling in segmented processor systems using buddy approach." *Journal of Intelligent & Fuzzy Systems* 32.4 (2017): 3129-3142.

[12] Jeff Offutt et al. "Quantitatively measuring object-oriented couplings" Software Quality Journal ,Volume 16, Issue 4, (2008),489–512.

[13] L. Pawar, R. Kumar, S. Arora, A. K. Manocha, "Optimized Route Selection On The Basis Of Discontinuity And Energy Consumption In Delay Tolerant Networks", Springer: Advances in Intelligent Systems and Computing, ISBN: 978-981-10-3769-6, 2016.

[14] Sunju Oh et al. " Cohesion and coupling metrics for ontology modules" Information Technology and Management (2011), 12:81.

[15] Kailash Patidar et al. "Coupling and Cohesion Measures in Object Oriented Programming" International Journal of Advanced Research in Computer Science and Software Engineering Volume 3, Issue 3, (2013).
[16] Bhullar, Rohit K., et al. "Intelligent stress calculation and scheduling in segmented processor systems using buddy approach." *Journal of Intelligent & Fuzzy Systems* 32.4 (2017): 3129-3142.