# Dynamic Coupling Metrics for Object Oriented Software

**Ravi Arora[1] & Mukul Kumar[2]**
[1]Assistant Professor, Department of Computer Science & Engineering
HMR Institute of Technology & Management
[2]Department of Computer Science & HMR Institute of Technology & Management

Received: May 12, 2018 Accepted: June 14, 2018

**ABSTRACT** *This Paper Introduce the concept of Coupling in between software modules from this we can check the accuracy of the software and find the bounding between one software module and another form this we can ensure the compatibility in between one to another module. This paper introduces the concept of Coupling and its type static and dynamic coupling. This is challenges task because it deals with the software testing, test cases module from the test suit. Selection of test case is depending on the software coupling under those test cases which may reduce time complexity and fault coverage rate. Mainly we focus on impact on Dynamic coupling and impact of open-source available code and notice the behavior of future prediction and also describe the term comes under dynamic coupling.*

***Keywords-*** *Object-Oriented Systems; Coupling; Test Case Selection , Model-based Testing, Dynamic coupling, OOP*

## 1. INTRODUCTION

The complexity of any software system majorly depends on mainly algorithmic complexity [1] and Structural complexity [2]. The structural complexity of the software that shows the huge design and branches of program codes. Algorithmic complexity depends on the expends hardware and scale of an actual problem and their resources.In order to decrease algorithmic complexity, there are several but effective approaches of optimization algorithms to solve the corresponding problem.

what is coupling: Coupling is the measure of degree of interdependence between modules, it mean when in a system develop two different modules are how much dependent and intent with each other, wherever we design a system then different models have their different functions performed, so coupling is a process of checking the dependency of the on function to another function.

**There are two ways of coupling:**

High Coupling and low Coupling

High Coupling It describe the way where we can see two modules are strongly related or not.

High Coupling It describe the way where we can see two modules are strongly related or not it mean both are highly dependent with each other. Low coupling it mean both of the modules have low dependency with each other so working with low coupling is more flexible because if we get some error in low coupling the easily find and fixed but in high coupling when we receive any error then then its disturbed the whole system because both are fully/ strongly depended with each other so its very difficult task for debug or isolated the problem in high coupling, that's is the reason from which we can prefer the low coupling whenever we design any system how models become couple or independent:

when module share data or exchange any kind of data between each other or the makes function calls to each other then we can identify modules are couple or dependent to each other, IN this we use the global data that leads to increase the coupling.

So how to control the coupling:

we should control the amount of information exchange between the modules , it mean sending only desired information, so wherever we use this kind of information then we can control the information Insure that pass only data not a control information(control information control the flow in between modules step by step )

Few Coupling are:

- Data Coupling
- Stamp Coupling
- Control Coupling
- External Coupling

- Data coupling: its the one of the desirable coupling, This coupling method is occurs when passing necessary information in between modules (there is no control information and data is passed in parameters).

- Stamp Coupling: In this kind of coupling method where we send the desired structure data which need to be exchange, the complete data structure passes from one module to another module but the major drawback is some time we don't require to send the complete data structure only part of the data structure is needed by another module only part of data structure part is needed.
- Control Coupling: In this type of coupling where communication in between modules occurs by passing control Infor or a module control the flow of another modules. As we know sending control information is no desirable so this is not valid for design purpose.
  eg flag, data that is set by one module and used by another module.
- External Coupling: In this Type of coupling in which where dependency of a module on another module which is share externally imposedcommunication protocol, data format, or device interfaces.

Especially, Object-oriented languages provide the way in which providing features that let to build such models that can easily extendible, flexible, maintainable; understandable that leads to reusable systems with better coordination with their environment. There are many important concepts introduced from object-oriented languages support is an Inheritance, Abstraction, encapsulation, polymorphism, message passing, information hiding and user-defined classes. These features (growth rapidly). In this paper, we mainly focuses on structural of software complexity, with the help of this engineers can control their complexity and add more feasibly approaches than used as algorithmic complexity.
On the basis of research prospective overall criteria of development model is divided into following : structure attributes, such as software code, research thatcorrelate in between software restructuring/quality and structure attributes shown in research [4-6]. The mainproblem with these popular researches as software the complexity takes more attention rather than small granularity on their local structural characteristics. As a result, much valuable information is lost. That impedes with understanding the whole software structure and main focus on reusing successful development of software that leads theexperiences on the system level. This situation occur due to our engineers more focus on the coding structure where a higher level of education and research approach is missing. Recently there rise in research that tends to focus and maintain the complexity network discipline, Which emphasizes understanding system and behavior as a whole instead of focusing on local Structure.
**The rest of the paper is divided into some section as follows:**
Section 2 showing comprehensive reviews in terms of dynamic coupling metrics.
Section 3 categorizes the present role of dynamic coupling metrics on the basis of their key feature.
Section 4 discusses the key observations.

### 2.1 Hassoun et al. Metrics
Hassoun et al. Metrics [5] proposed the Dynamic Coupling Metric (DCM) form DCM coupling, it takes part of program execution into account. "DCM can be able to use as the measurement of complete system runtime and their level/bounding build in between particular objects for coupling and it also provides the runtime environment complexity of the system.

### 2.2 Yacoub et al. Metrics
Yacoub et al.Metrics [2] proposed two ways of working first check the level of bounding is used Export Object Coupling (EOC) and Import Object Coupling. "This metrics shows the Measurements t early development phases from executable models design."

### 2.3 Zaidman and Demeyer Metrics
Zaidman and Demeyer Metrics [6] proposed (CQFS) Class Request for Service, in this metric help to analyze event traces large-scale industrial applications. CQFS registers every message that the instantiations of a certain class sends during the execution of a program. This metric can also abstract duplicate instantiations from the same class but a downside is that different objects that react differently due to polymorphic behavior also get abstracted.

### 2.4 Singh and Singh Metrics
Singh and Singh Metrics [7] Introduce of four class-level in dynamic coupling, the quality of object oriented software systems. They mainly focus on finding the key feature of coupled classes, most of the activity is done on run time help of both import and export coupling measures.

## 3. CATEGORIZATION BASED ON DYNAMIC COUPLING METRIC
There are many different aspects of working progress in dynamic coupling metrics characteristics on the basis of they classified.

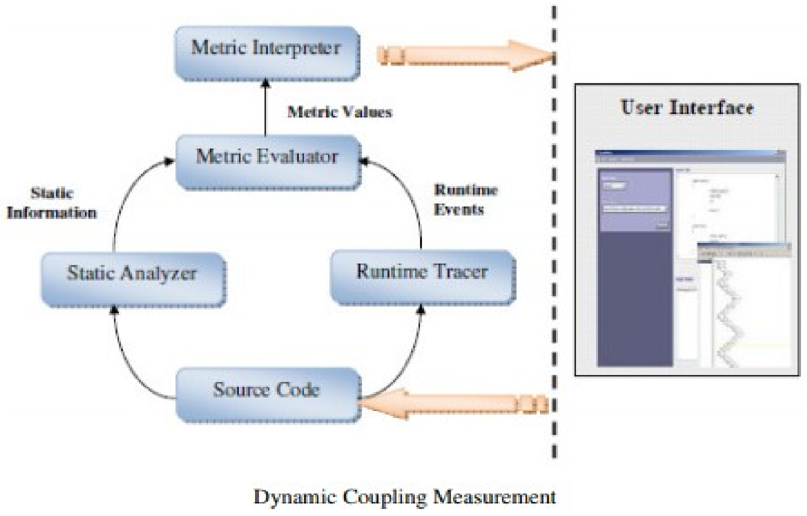


**Fig:1 Dynamic Coupling Measuring[3]**

We have divided into following major characteristics, that's shows the important perspective of various research purposes such as identification of all the class level coupling are:

- Dynamic analysis techniques
- Static metrics Relationship
- Real world applicability
- Metric Mapping Level
- Metric validation

## 3.1 Dynamic Analysis Techniques

Dynamic analysis techniques are the method that employed with analyzing the overall capture data while executing the program. Dynamic analysis is a process through which we can identify the actual working of the program that shows the information level of couplings made by the component in a program. The dynamic analysis is a process that invokes the number of different object-oriented features like polymorphism (more than one way of achieving same things while runtime/ executing), dynamic binding, inheritance, data abstraction.

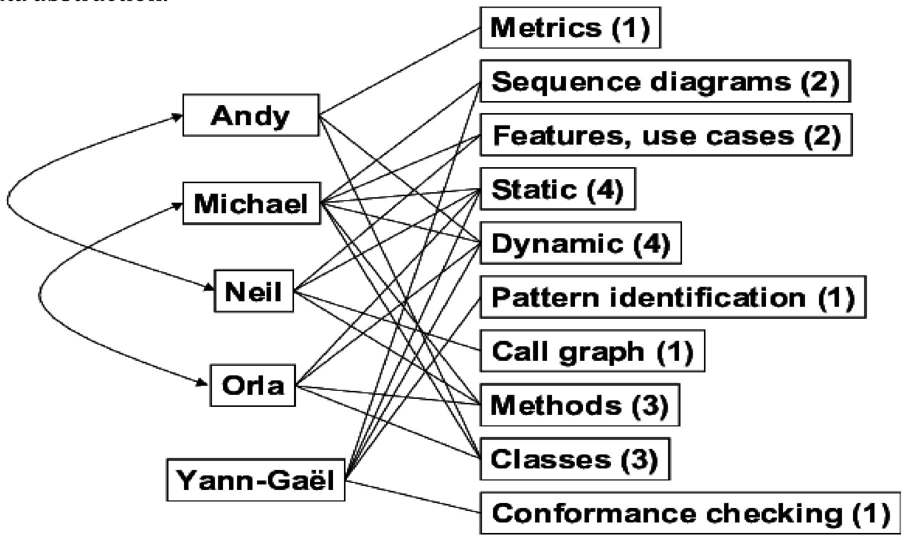


**Fig 2: Feature characterization of 1dynamic analysis techniques (https://www.researchgate.net/figure/Feature-characterisation-of-the-5-presented-dynamic-analysis-techniques_fig1_250371026)**

- **Dynamic Analysis using AOP:**

This approach shows the latest trend in performing dynamic analysis task. AspectJ [8] is an extension option of introducing oriented programming developed to use for Java, it introduces the new concepts like advice, join points, inter-type declarations point cuts and aspects. Hassoun et al. [9] used AspectJ for developing interceptive code as an independent programming unit with target programming with waving tool provided by the language environment in order to collect runtime data.

- **Executable Design Models:**

Dynamic concepts collect the information while executable design models. These models can be used as the fact coordinating with runtime through simulations that's leads to used real-time applications. IOC and EOC [2] metrics use ROOM (Real-time Object-Oriented modeling) models to leads to identify the aspects of the dynamic object-oriented programs. From this, we can pre-predict the further outcome or we can say final result in beginning stage or designing stage the major advantage of this technique. However, as said data is collected dynamically while execution on scenarios, that makes possible to check this scenario will work or not and if work then what will be the output on correlation is a mathematical relationship in which two or more events or variables are not causally related to each other.

## 3.2 Static Metrics Relationship

As per Study, we notice that the limited number of dynamiccoupling metrics have done and most of them are derived from static metrics.

Example: Mitchell and Power's Dynamic Coupling Between Object (DCBO) [3] metric is derived from Chidamber and Kemerer's CBO [10] metric. The Afferent (Ca) and Efferent (Ce) Coupling metrics proposed by Martin [11] have become bases for many dynamic coupling metrics. Yacoub's IOC and EOC metrics [2], Mitchell and Power's RI, RE, RDI, RDE metrics [4], Singh's DCa metric [9] and the metric suite from Arisholm et al. [5] seem to be dynamic variations of Martin's static metrics.[12]

## 3.3 Real World Applicability

The reason behind investigating the dynamic coupling
metrics was their inherent characteristic in terms actual coupling behavior that bring the future predicting with making it necessary to investigate the proposed metrics that applicable to real world applications. During the study face, most of the cases are found proposed method investigated only on the limited part of real-world applications. Java applications are most favorable for almost all the software metric investigations because of their major advantage of open source from this, there is the number of available open-source code available that assist the runtime metric data collection.

## 3.4 Metric Validation

A newly proposed way of software metric that introduces the forward challenging task of proving its validity and usefulness information.

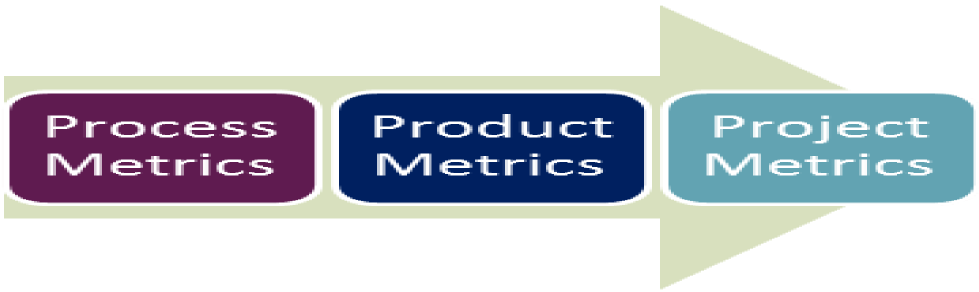
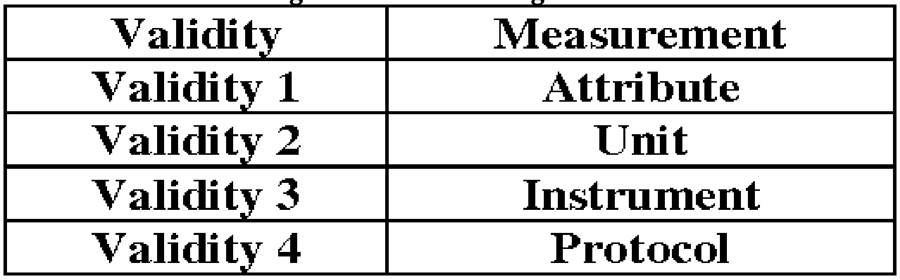


**Fig 3: Software Testing Metric**



| Validity | Measurement |
|---|---|
| Validity 1 | Attribute |
| Validity 2 | Unit |
| Validity 3 | Instrument |
| Validity 4 | Protocol |

**Fig 4: Metric Validation**

Software metric makes the validation on some criteria that meet to ensure its appropriateness for intended use [13]. The validated based on the following conditions [14]:
i. “The product metric must be associated with some important external attribute (feasibility and reliability).”
ii. “The product metric is actually measuring coupling metric is really measuring coupling.”
iii. “The product metric must be an improvement version over pre-existing product. it means it describes the better future production on the basis of collected matric that deals with reduction of faults".
Software metrics validation can be seen as theoretical in nature
[15]. The theoretical validation verifies that a metric meets the set of properties defined for the measurement entity. The validated theoretically is presented by using properties proposed by Weyuker [16], Fenton et al. [17] and, Briand et al. [18].[11] V. Gupta, J.K. Chhabra, Measurement of Dynamic Metrics using Dynamic Analysis of Programs, in *Proceedings of WSEAS Conference on Applied Computing Conference (ACC 2008) (*pp. 81-86), Istanbul, Turkey, 2008.

## 4. Key Observations

This section describes the study of dynamic coupling with the complex network, as per study shows there are the number of online resources available for conducting the task of dynamic software metric but there are some of the work that doesn't require the source code to execute the program and getting desirable [19]. As the most prevalent Object-Oriented software, Java prefers as true object-oriented programing language support that can run on various kinds of calculative devices(platform independent) and is an efficient, secure and reliable language. High cohesion and low coupling between objects will decrease the complexity of the system.
The relationship in between these metrics with external quality attributes was also studied and we observed that in most of situation dynamic coupling metrics proposed share a relationship with maintainability and change proneness and also maintain the level of object-oriented development metrics.

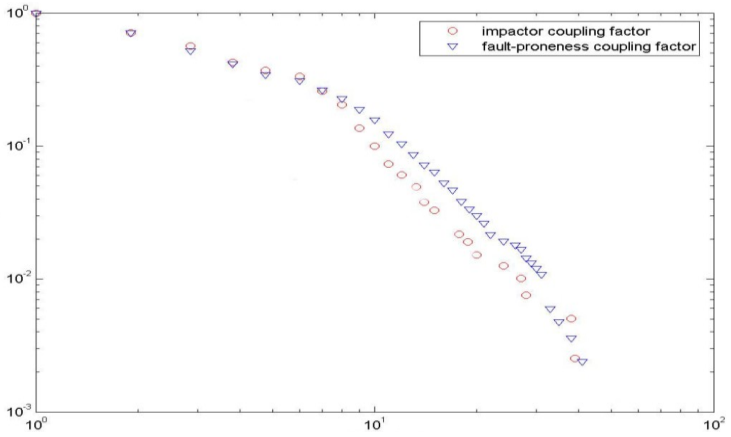


**Fig. 5:Log-log plot of the cumulative distribution of reciprocal of impact coupling factor and fault-proneness coupling factor of JEdit in its dynamic coupling network[19]**

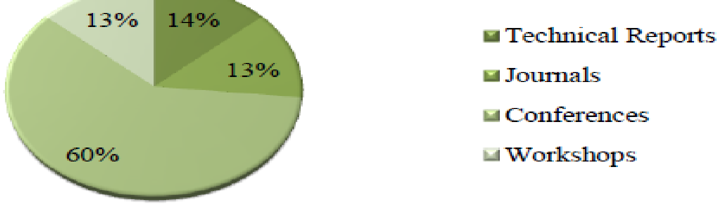


**Fig 6: Distribution of Research Publications**

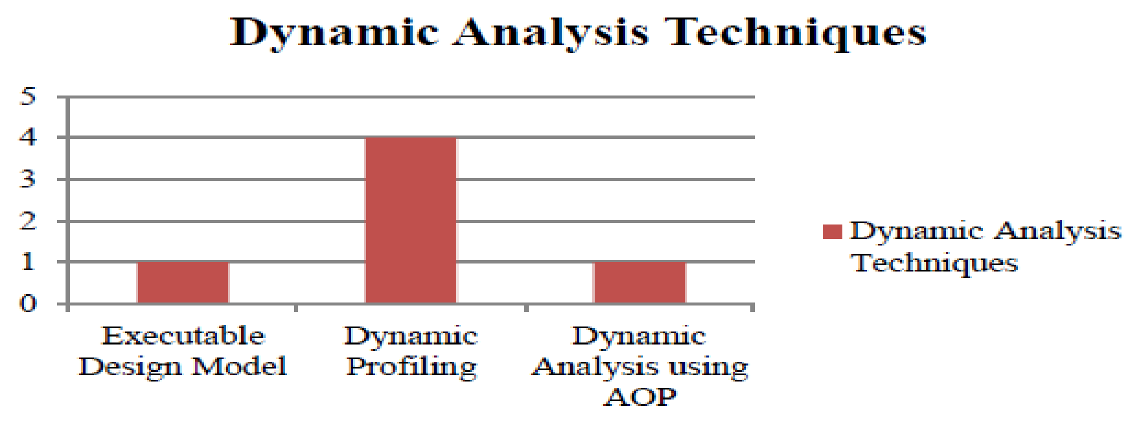


**Fig 7: Dynamic Analysis Techniques**

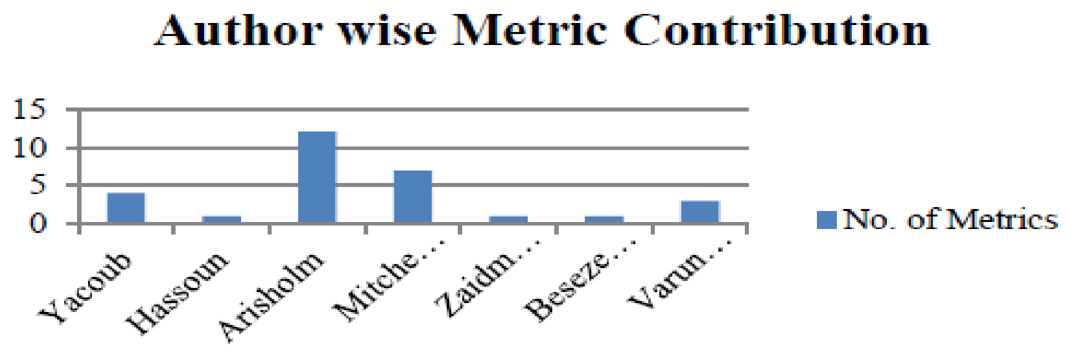


**Fig 8: Author wise Metric Contribution**

## 5. CONCLUSIONS AND FUTURE RECOMMENDATIONS

This paper provides the survey of dynamic coupling metrics on the basis of object-oriented systems. An attempt to characterize the software metrics according to their characteristics like Dynamic analysis techniques, Static metrics Relationship, Real world applicability, Metric Mapping Level, Metric validation. On the basis study, we can be observed that only a limited amount of work has been done in this field. We discuss the advantage and disadvantage of coupling and also introduce the type of coupling with other objects. We also make study about the Object-oriented property bounded with the software development process.

## REFERENCES

1. R. Adamov and L. Richter, "A Proposal for Measuring the Structural Complexity of Programs," J. Systems and Software, vol.12, pp. 55-70, 1990.
2. J. Hartmanis, "On Computational Complexity and the Nature of Computer Science," Comm. ACM, vol.37, pp. 37-43, 1994.
3. https://www.duo.uio.no/bitstream/handle/10852/9090/thesis2.pdf?sequence=1.
4. A. Mitchell, J.F. Power, An empirical investigation into the dimensions of run-time coupling in java programs, in Third Conference on the Principles and Practice of Programming in Java (pp. 9-14), Las Vegas, Nevada, USA, 2004.
5. Y. Hassoun, R. Johnson, S. Counsell, A Dynamic Runtime Coupling Metric for Meta Level Architectures, in Proceedings of Eighth Euromicro Working Conference on Software Maintenance and Reengineering (CSMR '04), pp. 339, 2004.
6. A. Zaidman, S. Demeyer, Analyzing large event traces with the help of coupling metrics, in Proceedings of the Fourth International Workshop on OO Reengineering, University Antwerpen, 2004.
7. E. Arisholm, L.C. Briand, A. Foyen, Dynamic coupling measures for object oriented software, in IEEE Transactions on Software Engineering, 30(8):491-506, 2004.

8. A. Beszedes, T. Gergely, S. Farago et al., Dynamic Function Coupling Metric and Its Use in Software Evolution, in Proceedings of the Eleventh European Conference on Software Maintenance and Reengineering (CSMR) (pp. 103-112), IEEE Computer Society, Washington DC, 2007.
9. P. Singh, H. Singh, Class-level Dynamic Coupling Metrics for Static and Dynamic Analysis of Object-Oriented Systems, International Journal of Information and Telecommunication Technology, 1(1): 16-28, 2010.
10. V. Gupta, Validation of Dynamic Coupling Metrics for Object- Oriented Software, ACM SIGSOFT Software Engineering Notes, 36(5), 2011 .
11. R. Martin, OO Design Quality Metrics - An Analysis of Dependencies, in Workshop Pragmatic and Theoretical Directions in Object-Oriented Software Metrics, OOPSLA'94, 1994.
12. Dynamic Coupling Metrics for Object Oriented Software Systems- A Survey (http://doi.acm.org/10.1145/2579281.2579296)
13. A. Meneely, B. Smith, L. Williams, Validating software metrics: A spectrum of philosophies. ACM Transactions on Software Engineering and Methodology, 21(4), 2012.
14. K. El. Emam, A methodology for validating software product metrics, Technical Report NRC/ERB-1076, National Research Council of Canada, 2000.
15. B. Kitchenham, S.L. Pfleeger, N.E. Fenton, Towards a Framework for Software Measurement Validation, in IEEE Transactions on Software Engineering, 21(12): 929-944, 1995.
16. E.J. Weyuker, Evaluating software complexity measures, in IEEE Transactions on Software Engineering, 14(9):1357-1365, 1988.
17. N.E. Fenton, S.L. Pfleeger, Software Metrics: A Rigorous & Practical Approach, Second Edition, International Thomson Computer Press, London, 1997.
18. L.C. Briand, J.W. Daly, J.K. Wust JK, A unified framework for coupling measurement in object-oriented systems, in IEEE Transactions on Software Engineering, 25(1):91-121, 1999.
19. A Dynamic Coupling Network of Object-Oriented Software(978-0-7695-5011-4/13 $26.00 © 2013 IEEE DOI 10.1109/IHMSC.2013.246)