International Journal of Innovative Technology and Exploring Engineering (IJITEE)
ISSN: 2278-3075 (Online), Volume-10 Issue-2, December 2020

Maintainability Metrics for Object-Oriented Software System Modifiability

Hanumantha Rao Battu, R.Satya Prasad, D.N.V.Syma Kumar

Abstract: Modifiability of the Object-Oriented system with inheritance states that how much amount of information would be modified in the individual class level and total system level. Measuring the modifiability in the appropriate manner leads to improve the maintainability and also quality of the specified software system. Existed metrics for modifiability gives the more complexity values and typical to work with them. Here main concentration is on to consider the proper modifiability of the Object-Oriented system and reduce the complexity in judging the modifiability. The identified modifiability metrics of the class and system level are validated with the well known Weyker's properties to strengthen the proposed metrics and which may utilized in their research work by researchers.

Keywords: Maintainability metrics, Inheritance hierarchy, System Modifiability, Class Modifiability, System Maintenance, Weyker's properties

I. INTRODUCTION

An Object-Oriented or traditional software system measure their functional or non-functional requirements with metrics and estimated models. A metric gives the accurate measurement for the functional requirements and appropriate measurement for the non functional requirements like quality and its parameters. In the issue of the improving software quality several studies were existed [1], [2]. The usage of existed Object-Oriented metrics in the several programming languages and programming fields [3] in suitable manner. .The ISO/IEC-9126-1 states that the Modifiability factor of Maintainability can be defined as "process of corrections, improvements or adoptions of the software to changes in environment and in requirements and functional specifications" [4]. Previously so many authors measure the modifiability [5], [6], [7], [8], [9], [10], [11], [12] to maintain the Object-Oriented software system with their metrics and models in proper manner. Inheritance is the specialized property of the Object-Oriented system to extend and modify the behaviors from one existed class to new derived classes. By considering the inheritance hierarchy of the software system several metrics [13], [14], [15], [16], [17], [18], [24], [28] were developed to measure the maintainability parameter.

Revised Manuscript Received on December 30, 2020.
***** Correspondence Author

Mr. Hanumantha Rao Battu*, Research Scholar, Department of Computer Science and Engineering, Acharya Nagarjuna University, Guntur ,Andhra Pradesh, India. E-mail: hanuma9999@yahoo.com

Dr.R.Satya Prasad, Professor, Department of Computer Science and Engineering, Acharya Nagarjuna University, Guntur ,Andhra Pradesh, India.E-mail: prof.rsp@gmail.com

Dr. D.N.V.Syma Kumar , Associate Professor, Department of Computer Science and Engineering, St. Ann's College of Engineering & Technology, chirala, Andhra Pradesh, India. E-mail: prof.dnvsk@gmail.com

Several observations were done [27], [28], [29], [30], [31], [32] on the Inheritance based developed Object-Oriented techniques helps by reducing the redundancy and improving efficiency of the software system. Every proposed metric need to prove its theoretical and mathematical background by satisfying the well known properties like weyker's for any software metrics. Weyker's[19] developed total nine number of properties which would help in judging the validity of the metrics. The developed object-Oriented metrics were validated against the weyker's properties by the more number of researchers [14], [20], [21], [22], [23],[28],[33], [35]. In the developed weyker's properties some of the properties applicable to inside data of Object-Oriented classes and traditional programs purpose. Hence those metrics which may developed based on the class based inheritance (not the inside data and functions) were may not applicable to less number of weyker's properties [33].

The Organization of this paper is in the following manner. Literature survey covers the developed Object-Oriented inheritance based maintainability metrics like DIT, NAC, NOC, NDC, AM and AID. The proposed metrics discussed in the Section-3.Section-4 deals about the validity ofproposed metrics against the weyker's properties. Comparison done between the existed and proposedmetrics was done in Section-5. The Conclusion and Future scope of this paper was presented at Section-6 & Section-7 respectively.

II. LITERATURE SURVEY

This Literature survey gives detail the report on the existed maintainability metrics which were based on the Object-Oriented software class inheritance property. Depth of Inheritance Tree (DIT) metric was developed by chidember-kerner [24], [25],[28] states that maximum depth from the root node to the current node. The problem of this metric is the occurrence of ambiguity in several situations in the measurement of maintainability. W.Li[16] proposes the new metric called Number of Ancestor Classes(NAC) to give the solution to the DIT metric ambiguity problem. NAC metric measures the number of classes effect the class design with the inheritance property in the Object-Oriented inheritance hierarchy. Chidember-kerner [24], [25], [28] proposes another metric named Number of Childs (NOC) which gives the number of classes that are directly(immediate classes) inherited from the individual class. W.Li [16] proposes new metric called Number Descendent Classes (NDC) to consider total number of sub classes affected with the inheritance. Henderson-Sellers [15] developed Average Depth of Inheritance (AID) metric for applying the average complexity values in the DIT metric.

Sheldon-jerath [18] proposed modifiability metric called Average Modifiability [AM] by considering the successors and understandability of the individual classes.

III. NEW INHERITANCE METRICS

The proposed metrics in this paper are named as Individual Class Modifiability (ICM) and System Modifiability (SM). In the process of identifying Modifiability metrics preferred the two different inheritance hierarchy based class diagrams which were placed at the end of this paper. In the two class diagrams first one is represented with Directed Acyclic Graph (DAG) with no loops [26] with the multiple inheritance property. Second class diagram with the normal tree hierarchies (Inheritance) having the simple relations. The Modifiability factor of the class diagram calculated based on two rules. The Rule-1 says that the variables of the class causes to the same class modifiability. Rule-2 says that the subclasses which are inherited by the specified class are to be modified. Based on this inheritance behavior of classes total inherited classed can effect in best case, half of the inherited classed to be modified in average case and no on class need to be modified by preferring the worst case scenario. Generally average case should be followed for the most of the situations. Hence average case class modifiability was considered in this paper also for measuring the metrics. The Individual Class Modifiability (ICM) metric considers the specified class taken into account and the average number of inherited classes of the given class. The Total System Modifiability (SM) metric is the sum of all the individual classes modifiability values divisible by the total number of classes of the system.

Individual Class Modifiability is

ICM = (Number of Sub Classes/2)+1

Total System Modifiability is

SM = $\sum_{i=1}^{n} ICM_{i}$ / n

ICM_i = Individual Class Modifiability of the Class i.

n= Number of classes.

Applying the above metrics on figure1

ICM(H)=3.5	ICM(G)=4
ICM(F)=3	ICM(E)=1.5
ICM(D)=1	ICM(C) =2

ICM(B) = ICM(A) = 1

SM of figure1 = 2.125

Applying the above metrics on figure2

ICM (P) =4

ICM (Q) =ACM(R) =2

ICM(S) =ICM (T) =ICM (U) =ICM (V)=1

SM of figure2 = 1.714

IV. PROPOSED METRICS VALIDATION

The metric validation is the best way to strengthen the proposed metrics to use in future better way. The Weyker's [19] developed nine properties give the mathematical and functional proofs in the process of metrics validation.

Property-1:- Non-Coarseness

The class X and class Y are having the proposed metric M then the Non-Coarseness found that if M (X) ≠ M (Y) was satisfied.

In Figure-1, ICM of class H is different from ICM of class G. ICM (H) =3.5 and ICM (G) =4 then ICM(H) ≠ICM (G). In Figure-2 ICM of class P is different from ICM of class Q. Values of ICM (P) =4 and ICM (Q) =2 then ACM (P) ≠ACM (Q). Hence ICM metric satisfies the weyker's property-1.TheFigure-1 and Figure-2 gives two different ISM values for the both class diagrams. SM value of Figure1 is 2.125 is different from the figure2 SM value is 1.714. It means that SM metric satisfies the weyker's property-1.

Hence the proposed metrics ICM and SM are satisfies the weyker's first property Non-Coarseness successfully.

Property-2:- Granularity

Granularity property sates that the metric value must be applied on the finite number of programs. The proposed metric value is the non-negative complexity value. The object-oriented metric class level metrics are automatically satisfied this property [4] because class levels are having by the every object-oriented inheritance hierarch. Here proposed metrics also has the inheritance hierarchy with class levels. So, the proposed ICM and SM are also satisfies the weyker's second property.

Property-3:- Non-Uniqueness

The same metric value given by the two different classes X and Y. It means that M(X)=M(Y). The Figure-1 gives the ICM value of class D is same as ICM values of class B. So, ICM(D)=1 and ICM(B)=1. It means that ICM(D)=ICM(B). In Figure-2 ICM value of the class Q is same as the class R. Here ICM(Q)=2 and ICM(R)=2.So ICM(Q)=ICM(R). Hence proposed ICM metric satisfies this Non-Uniqueness property. At the system level by considering the various class diagrams the SM metric value of one class diagram is equal to another class diagram .Hence SM metric also satisfies the weyker's third property Non-Uniqueness successfully.

Property-4 :- Design Implementation

If two designers design the class diagrams with same number of classes it has to show the two different metric values. The designed class diagrams must be utilized the proposed metric.

The proposed metrics ICM and SM were satisfied this Design Implementation property because any two designers draw the number of class diagrams with the same number of classes of the same program but they would follow the different inheritance hierarchy and class levels. Hence the designs of the two systems would be different. The proposed metrics ICM and SM also suitable for the different designs for different designers. Hence ICM and SM metrics were satisfies the weyker's fourth property successfully.

Property-5:- Monotonicity

This property states that metric value of the grouping of two different metric valued classes is greater than or equal to the individual classes. Suppose X and Y classes are having the two different metric values then the grouping of the both denoted as X+Y metric value is greater than or equal to the individual X and Y classes metric values.

It means that M(X+Y) ≥ M(X) and M (X+Y) ≥ M(Y).
Here the proposed metrics need fulfill the three possible sub situations for satisfying the weyker's monotonicity property.
1. If class X and class Y are siblings.
The Figure-1(b) shows that ICM(X)=1 and ICM(Y)=1.The grouping of the both siblings as one class as X+Y then the finalized value of the ICM(X+Y)=1, which is equal to the both the metrics of X and Y. The Figure-2(b) shows that ICM(Q)=2 and ICM(R)=2. The grouping of both siblings as one class as Q+R, then the finalized result of the ICM(Q+R)=3 which is greater than both the metrics of classes Q and R individually.
Hence case-1 was successfully satisfied by the ICM metric.
If class X is the child of another class Y.
The Figure-1(c) gives the ICM(C)=2 and ICM(B)=1.The Combination of the C and B classes gives metric value as ICM(C+B)= 1.5. The resultant ICM(C+B) metric value is equal to class E and less than the values of class C. The Figure-2(c) gives the values of the metric ICM(Q)=2 and ICM(S)=1. The combination of the Q and S gives the metric value as ICM (Q+S)=1, which is less than the metric value of Q and greater than Q.
So case-2 of Monotonicity property is not satisfied by the proposed metric ICM, because the combination of the two individual classes (one is child to another) break the structure and reduce the Object-Oriented Class Inheritance hierarchy. Oncethe structure of the class diagram was changed then the possibility of reducing the metric value is very much high. So ACM is not satisfied the monotonicity property (case-2). The existed inheritance hierarchy based metrics like DIT,NAC,NDC,AID and AM also not satisfying the weyker's fifth property[23] because they were also focused on the class only not the inside matter of the class.
If class X and class Y are neither siblings nor children of each other.
The Figure-1(d) shows that the ICM(C)=2 and ICM(E)=1.5. The combination of C and E classes gives the metric value ICM(C+E) =2, which is same to class C metric value and greater value than class E metric value. The Figure-2(d) shows that ICM (Q) =2 and ICM (V) =1. The combination of both the classes metric value is ICM (Q+V) =2, which is greater than the class V metric value and equal to the class Q metric value.
Hence case-3 of Monotonicity property is satisfied by ICM metric successfully.
The proposed ICM metric satisfies the weyker's monotonicity property (two situations) but failed in the second situation. Hence the proposed ICM and SM metrics are not satisfies this property similar to the existed inheritance based metrics like DIT, NAC, NDC, AID and AM.
Property-6:- Non-Equivalence of Interaction
If class X and class Y shows the same metric values then individual combination of these individual classes with another class Z the finalized metric values of X+Z not equal to the metric values of Y+Z.
In the Figure-1 the metric values of ICM(B)=ICM(D)=1. The combinations of two these classes with another class C as displayed in Figure-1(e) and Figure-1(f) .The finalized metric values of ICM(B+C)=1.5 and ICM(D+C)=2 but both are not equal. So ICM(B+C)≠ICM(D+C).Consider the Figure-2 with the metric values of ICM (Q) = ICM (R) =2.
The individual combination of these two classes with class T as displayed as in Figure-2 (e) and Figure-2(f).The finalized metric values of ICM(Q+T)=1.5 and ICM(R+T)=2, but the both values are not equal. So ICM (Q+T) ≠ ICM(R+T). Hence the proposed ICM metric satisfies the weyker's sixth property. By considering the system level the proposed modifiability metric SM also satisfies the weyker's sixth property.
Property-7:- Significance of Permutation
The proposed metric values not to be changed even permutations on the program statements are performed. This metric is not applicable to class diagram based metrics (not considering the inside data and functions).This property only applicable for traditional programming where the inside matter of the program taken major consideration for the selection of the metrics. This property is not suitable for most of the object-oriented metrics because these are considered the class as single unit not the inside data of the class. The existed metrics like DIT, NAC, NOC, NDC, AID and AM also not satisfies this property [23]. Hence the proposed metrics ICM and ASM also not applicable to this property.
Property-8:- No Change of Remaining
If the class name is renamed then the metric values of the given class need not to be changed.
This property can be applicable to all the object-oriented class based metrics because rename the class not shown any effect on that class metric value. The proposed ICM and SM metrics are also based on the object-oriented class inheritance hierarchy. Hence the ICM and SM metrics are automatically satisfies the weyker's eighth property successfully.
Property-9:- Interaction complexity
If two classes X and Y combination denoted by X+Y then the metric value of X+Y is greater than the summation of the individual classes X and Y.
It means M(X+Y)>M(X) +M(Y)
Any developed metric based on the object-oriented class diagram is not applicable for this property [23].
The combination of two classes into single one takes lowest or equal metric values than the addition of two class's individual metric values.
So, this property can not feasible for classbased metrics. Here the proposed metrics are based on the Object-Oriented design.
Hence the ICM and SM metrics are also not satisfies this Interaction complexity property-V.

V. COMPARISION WITH EXISTED METRICS AND RESULTS

Here the proposed ICM and SM metrics comparison was done with the existed metrics namely DIT, NOC, NAC, NDC, AID and AM.
For the DIT, NOC, NAC, NDC and AID metrics validation results are taken and against weyker's properties.

Based on the Weyker's properties applicability comparison has done between the proposed and existed metrics then results are place in below table format.

Table-I: Inheritance based Metrics validation with Weyker's properties

Property/ Metric	DIT	NOC	NAC	NDC	AID	AM	ICM	SM
1	√	√	√	√	√	√	√	√
2	√	√	√	√	√	√	√	√
3	√	√	√	√	√	√	√	√
4	√	√	√	√	√	√	√	√
5	×	√	×	×	×	×	×	×
6	√	√	√	√	√	√	√	√
7	×	×	×	×	×	×	×	×
8	√	√	√	√	√	√	√	√
9	×	×	×	×	×	×	×	×

.

√ - weyker's property satisfied by the metric.

× - weyker's property not satisfied by the metric.

In the results comparison main task is to reduce the modifiability complexity value of the system. For this reason comparison done between the proposed SM and existed AM metrics. In this comparison SM had given the much reduced results than the AM results. The finalized results of the Average system modifiability of the proposed SM and existed AM are placed in below table.

Table-II: System level Modifiability metric values

Figure	AM	SM
1	4.37	2.125
2	3.1	1.714

VI. CONCLUSION

In this paper various existed inheritance metrics are discussed their validity with the weyker's properties. Here main focus is on the reduction of modifiability complexity value of the system level (SM) and measure the individual class modifiability (ICM) in simple manner. The proposed ICM and Sm metrics are validated with weyker's properties was done and compared the previous existed inheritance based metrics validity also. The results of the system modifiability metric(SM) had given the lowest values compared with the existed AM values.

FUTURE WORK

Similar to many authors the proposed metrics in this paper also focused only the class as a unit not the inside details of the class. In future extend this work to the inside details of the class also. Here our concentration is on the modifiability factor of the maintainability. The other factors factor's metrics also need to be consider for improving the maintainability and the quality of the Object-Oriented software system.

REFERENCES

1. Amjan Shaik,C. R. K. Reddy, Bala Manda, Prakashini. C, Deepthi. KMetrics for Object Oriented Design Software Systems: A Survey Journal of Emerging Trends in Engineering and Applied Sciences (JETEAS) 1 (2): 190-198.
2. M.S.Ranwat,A.Mittal,S.K.Dubey Survey on impact of software metrics on software quality (IJACSA)International journal of Advanced Computer Science and Applications, Vol.3,No.1,2012.
3. Darcy, D.P.—Kemerer, C. F.: OO Metrics in Practice. IEEE
4. Softw. 22,6 November 2005, pp. 17–19. DOI:
5. http://dx.doi.org/10.1109/MS.2005.
6. ISO/IEC 9126-1 (2001) Institute of Electrical and Electronics Engineers, Part1: Quality Model.
7. Dromey, G.R. (1995) A model for software product quality, IEEE Transaction on Software Engineering.
8. Harrison W., Magel K., Kluczny R. and DeDock (1982), "Applying software complexity metrics to program maintenance" , IEEE Computer, vol. 15, pp. 65-79.
9. Kiewkanya M., Jindasawat N. and Muenchaisri P. (2004) "A Methodology for Constructing Maintainability Model of Object-Oriented Design," Proc. 4th International Conference on Quality Software, pp. 206 – 213, IEEE Computer Society, 2004.
10. Mattson M.K., Deursen A.V., Reiger R., Canfora G., Ihme T., Engel T., Chiorean D., Lehman M.M. and Wernke J. (2006) A Model of Maintainability Suggestion for Future Research, In Proceedings of Software Engineering Research & Practice, pp. 436-441.
11. Kiewkanya M. and Muenchaisri P. (2011) Constructing Modifiability Metrics by Considering the Different Relationships. Chiang Mai J. Sci.; 38 (Special Issue): pp. 82-98.
12. D.N.V. Syma Kumar and R. Satya Prasad "Maintainability Model with Object-Oriented Software Metrics" International Journal of Engineering & Technology IJET-IJENS Vol:15 No:04 , August 2015.
13. D.N.V. Syma Kumar and R. Satya Prasad "Object-Oriented Software Metrics for Maintanability" International journal of Computer science & Information Security IJCSIS Vol.13 , No.10, October 2015.
14. N.Nwe and Thu "Measuring Modifiability in model development using object-oriented metrics" Advances in science, Technology and Engineering systems ASTESJ vol.3, No.1,244-251(2018) .
15. Poels G. and Dedene G. (2001) "Evaluating the Effect of Inheritance on the Modifiability of Object-Oriented Business Domain Models", *5th European Conference on Software Maintenance and Reengineering (CSMR 2001)*, Lisbon, Portugal, pp. 20-29.
16. Abreu, F.B.—Carapuca, R.: Candidate Metrics for Object-Oriented Software within a Taxonomy Framework. Journal of System Software, Vol. 26, 1994,pp.87–96
17. Henderson-Sellers, B.: Object Oriented Metrics: Measures of Complexity. Pren-tice Hall PTR: Englewood Cliffs, NJ, 1996; pp. 130–132.
18. Li, W.: Another Metric Suite for Object-Oriented Programming. Journal of Systems and Software, Vol. 44, 1998, pp. 155–162.
19. Lorenz, M.—Kidd, J.: Object-Oriented Software Metrics. Prentice Hall 1994,ISBN: 013179292X.
20. Sheldon, F.T.—Jerath, K.—Chung, H.: Metrics for Maintainability of Class In-heritance Hierarchies. Journal of Software Maintenance 14, 3 May 2002, pp. 147–160.
21. Weyuker, E. J.: Evaluating Software Complexity Measures. IEEE Transactions on Software Engineering, Vol. 14, 1988, No. 9, pp. 1357–1365.
22. Cherniavsky, J.—Smith, C.: OnWeyukers Axioms for Software Complexity Mea-sures. IEEE Transaction on Software Engineering, Vol. 17, 1991, No. 6, pp. 636–638.
23. Gursaran, G.R.: On the Applicability of Weyuker Property Nine to Object-Oriented Structural Inheritance Complexity Metrics. IEEE Transaction on Software Engineering, Vol. 27, 2001, No. 4, pp. 361–364.
24. Sharma, N.—Joshi, P.—Joshi, R.K.: Applicability of Weyuker's Property 9 to Object-Oriented Metrics. IEEE Transaction on Software Engineering, Vol. 32, 2006, No. 3, pp. 209–211.
25. Deepti Mishra: New Inheritance complexity metricsfor object –oriented software systems:An evaluation with weyker's properties Computing and Informatics, Vol. 30, 2011, 267–293.
26. Chidamber, S.R.—Kemerer, C.F.: To wards A Metrics Suite for Object Oriented Design,OOPSLA'91,pp. 197-211,1991.
27. Chidamber, S.R.—Kemerer, C.F.: A Metrics Suite for Object Oriented Design, M.I.T.Solan School of Management 1993.
28. wang CC, shih TK ,paiWC An automatic approach to object –oriented software testing and metrics for c++ inheritance hierarchies, proceedings International Conference on Automated Software Engineering (ASE'97), IEEE Computer Society press 1997;934-938

29. Basili VR,Biand LC Melo WL A validation of object-oriented metrics as quality indicators, Technical Report,University of Maryland, Department of computer science,1995; 242-249.
30. Chidamber, S.R.—Kemerer, C.F.: A Metrics Suite for Object Oriented Design.IEEE Transactions on Software Engineering, Vol. 20, 1994, No. 6, pp. 476–493.
31. Ghassan alkadi, Application of a revised DIT metric to Redesign an OO Design, Journal of Object technology , Vol. 2,Issue 3,pp 897-910,2005.
32. Basili, V.R.:Viewing Maintenance As Reuse Oriented Software Development. IEEE Software, Vol. 7, 1990, No. 1, pp. 19–25.
33. Cartwright, M.—Shepperd, M.: An Empirical Analysis of Object Oriented Soft-ware in Industry. In: Bournemouth Metrics Workshop, April, Bournemouth, UK1996.
34. Li, W.—Henry, S.: Object-Oriented Metrics That Predict Maintainability. Journal of Systems and Software, Vol. 23, 1994, No. 2, pp. 111–122.
35. Sanjay Misra and Ibrahim Akman :Applicability of weyuker's Properties on OO Metrics: Some Misunderstandings ,ComSIS Vol. 5, No. 1, June 2008.
36. K. Rajnish, A. K. Choudhary, A. M. Agrawal, "Inheritance Metrics for Object-Oriented Design", *IJCSIT*, Vol. 2 No.6 , December2010,pp.13-26.
37. K. Rajnish and V. Bhattacherjee, "Class Inheritance Metrics-An Analytical and Empirical Approach", *INFOCOMP-Journal of Computer Science*, Federal University of Lavras, Brazil, Vol. 7 No.3, pp. 25-34, 2008. Brazil, Vol. 7 No.3, pp. 25-34, 2008.

AUTHORS PROFILE

Mr. Hanumantha Rao Battu, had M.Tech Degree in Computer Science & Engineering from AAIDU University, Allahabad, U.P.. He is currently pursuing his Ph.D. in the Department of Computer Science & Engineering from Acharya Nagarjuna University, Guntur, Andhra Pradesh. His Research Interest includes Software Engineering, Computer Networks, Operating Systems and ArtificialIntelligence.

Dr. R. Satya Prasad received Ph.D. Degree in computer science in the faculty of Engineering in 2007 from Acharya Nagarjuna University, Andhra Pradesh. He received gold medal from Acharya Nagarjuna University for his outstanding performance in master's degree. He is currently working as Professor & HOD in the department of Computer Science & Engineering., Acharya Nagarjuna University. His current research is focused on Software engineering. He has published several papers in National & International Journals.

Dr.D. N.V.Syma Kumar, Presently working as an Associate Professor in the department of Computer Science & Engineering at St.Anns College of Engineering &Technology, Chirala. He had received his Ph.d. degree from Krishna University in the stream of Computer Science & Engineering in 2017.He had published more than 10 international publications in Scopus and various reputed journals. His research interests are Software Engineering, Data Mining, Artificial Intelligence and Machine learning.

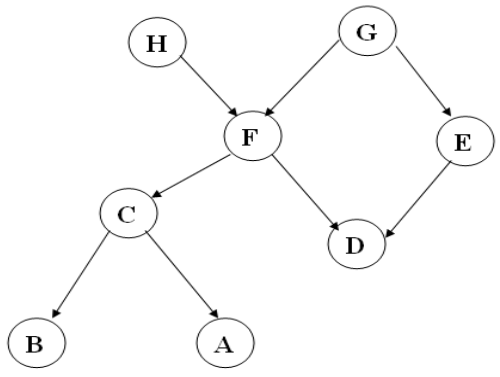

Fig 1(a) : Class Inheritance Hierarchy

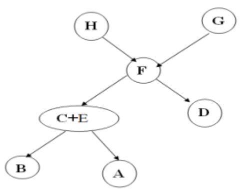

Fig 1(d) : Combined C+E Class Inheritance Hierarchy(case-3)property-5.

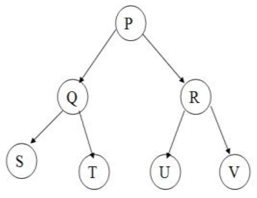

Fig 2(a) : Class Inheritance Hierarchy

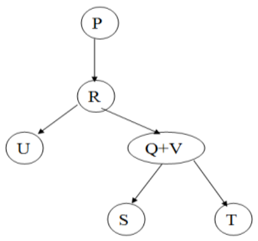

Fig 2(d) : Combined Q+V Class Inheritance Hierarchy (case-2) property-5.

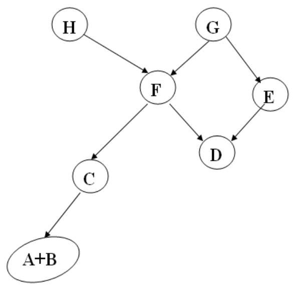

Fig 1(b) : Combined A+B Class Inheritance Hierarchy(case-1)property-5.

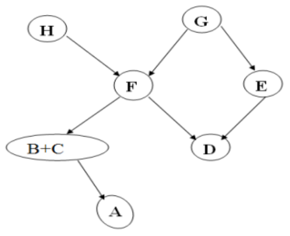

Fig 1(e) :Combined B+C Class Inheritance Hierarchy property-6.

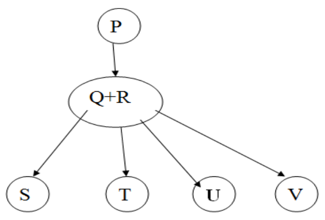

Fig .2(b) : Combined Q+R Class Inheritance Hierarchy (case-1)property-5.

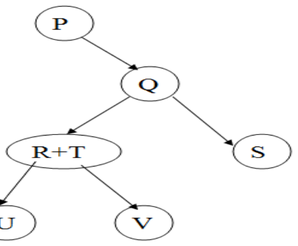

Fig 2(F) : Combined R+T Class Inheritance Hierarchy property-6

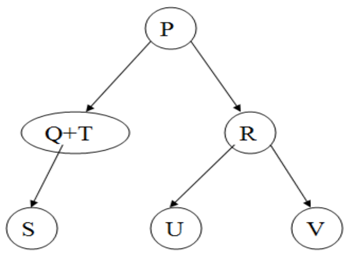

Fig 2(E) : Combined Q+T Class Inheritance Hierarchy property-6.

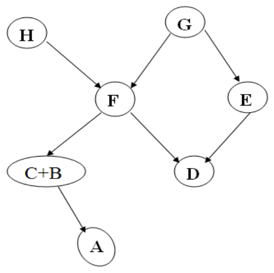

Fig 1(c) : Combined C+B Class Inheritance Hierarchy(case-2)property-5.

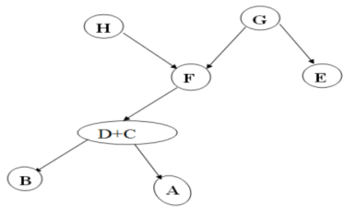

Fig 1(f) : Combined D+C Class Inheritance Hierarchy property-6.

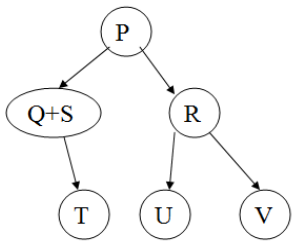

Fig 2(c) : Combined Q+S Class Inheritance Hierarchy (case-2)property-5.