Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 125 (2018) 754–761

Procedia Computer Science

www.elsevier.com/locate/procedia

6th International Conference on Smart Computing and Communications, ICSCC 2017, 7-8 December 2017, Kurukshetra, India

Dynamic Reconfiguration of robot software component in real time distributed system using clustering techniques

Amit Kumar Srivastava[a],Shishir Kumar[b,**],

[a,b]*Jaypee University of Engineering & Technology,Guna 473226, India*

[a]*amitsri1983@gmail.com,*[b]*dr.shishir@yahoo.com*

Abstract

The requirement of the distributed system in optimum resource is creating the new scope of business for the software industry in the area of component based technology with data mining techniques. In optimization of software, the development cost, redesign complexity, resource planning has been growing area of search based software engineering. The reuse of artifacts has been growing over the last decade and the recent works have focus on dynamic reconfiguration of logical component design. However, there are still gaps for identification and reconfigurations of software service component. In this paper, the proposed algorithm gives effective approach for dynamically reconfigurable of the design in real time application with new software service component using clustering techniques. This algorithm is able to handle the large system data set and also applicable for the multiple hierarchy redesign problem. The proposed algorithm compared against similarity score algorithm using a case study of robot path planner class diagram.

Keywords: Distributed System, Software Component, Clustering, Similarity Score Algorithm

1. Introduction

To identify a relevant software component from the software component set is a challenging task and it has been mentioned as NP complete problem [1]. According to Birkmeier and Overhage three categories of components are business oriented components, architectural or logical component and the technical components [2].Usually the reuse of software service component and component based development is not in practice of robot application[3].It seems to be an demanding area in the robotic software service field[4].During the development life cycle of robot software, a software designer is responsible to decompose a system using relevant software components. However, because of lack

* *Corresponding author
E-mail address: dr.shishir@yahoo.com (Dr.Shishir Kumar).

of perfection it is an extremely difficult and error prone task to identify and decompose logical component without any supporting tool[2].To overcome this difficulty, several methods have been suggested to identify taxonomic position of vulnerability or software defect density or logical component[5, 6, 7, 8, 9]. In current scenario developers are trying to find the suitable software component for the existing system in a specified design patterns.

According to the Christopher Alexander et.al, a unique pattern can be used in the design of object oriented software component[10]It also helps to developer to choose alternative that make a system reusable and avoid that compromise reusability.

This paper concentrates on dynamic reconfiguration of robot software service using reuse software service component using clustering techniques. A component is representing the prerequisite information on software services can often be obtained directly from the software requirements phase [11].The class diagram of robot software service and Gamma et.al.[12].The class diagram represent the static structure, the attributes operations and the relationship between the classes. Class diagram of robot software service is partitioned into the subsystem having at least one generalization or specialization relation in between the classes.This subsystem class diagram is the first input for the proposed algorithm and for second input we take the catalog of design pattern [12].For a large system, Tsantalis et.al. handle the issues of multiple inheritance in system but not applicable for the robot software service component.

The proposed algorithm also dealt with the problem of reuse entity for defined requirement on the basis of similarity coefficient using clustering techniques[13].The algorithm uses centroid which has been decided by the similarity between the subsystem and design patterns and also valid for the outlier. The component identification problem is based on feature based similarity of the search space but features are not clear, e.g. analysis classes. It is not easy task to define features of analysis classes in a multidimensional space. The proposed algorithm also has self organized capability to organize the number of components upto the matched design pattern features. It also dealt the problem of feature based multidimensional space having some noise in data. For analysis of proposed algorithm the theoretical comparison is shown using a case study of Robot path planner class diagram.

The rest section of this paper is arranged as, Section 2 presents problem formulation. Related work is shown in Section 3. Mathematical background of algorithm is presented in section 4. Section 5: Describe the steps of proposed algorithm. Result analysis and conclusion has been shown in section 6 and 7.

2. Problem Formulation

In Rational Unified Process (RUP) several problem models is defined for the representation of requirement of a common problem [14]According to the recommendation of RUP methodology, the analysis class diagram consist the description of the all three classes: boundary (interface), control and entity class. In Fig 1. User describes the objective in form of a story about the requirement. First step of the proposed method is the prepare a class diagram according to the objective by Visual Paradigm Tool 12.1. and then identify the subsystem of class hierarchies. After collection of class diagram of system and types design pattern, prepare a cluster that defines the second problem. In third step, the reconfigure the logic view of the system is performed on the basis of clustering of design pattern and subsystem. The logical view structure of the system has the self organized capability to update with the component structure if any optimum possible partitioning threshold is achieved. The goal of this paper is as follows:

- To partition the multiple hierarchies (Generalization relationship between the classes) of the software service using the cluster of design pattern
- To identify the cluster of similar type of subsystem and design pattern of robot software services logical components, used as a reuse entity for defined requirements on the basis of Pearson & Heron-II similarity coefficient.
- Systems having dynamic reconfigurable capability to rearrange the logical view of the robot software service according to the design pattern cluster.

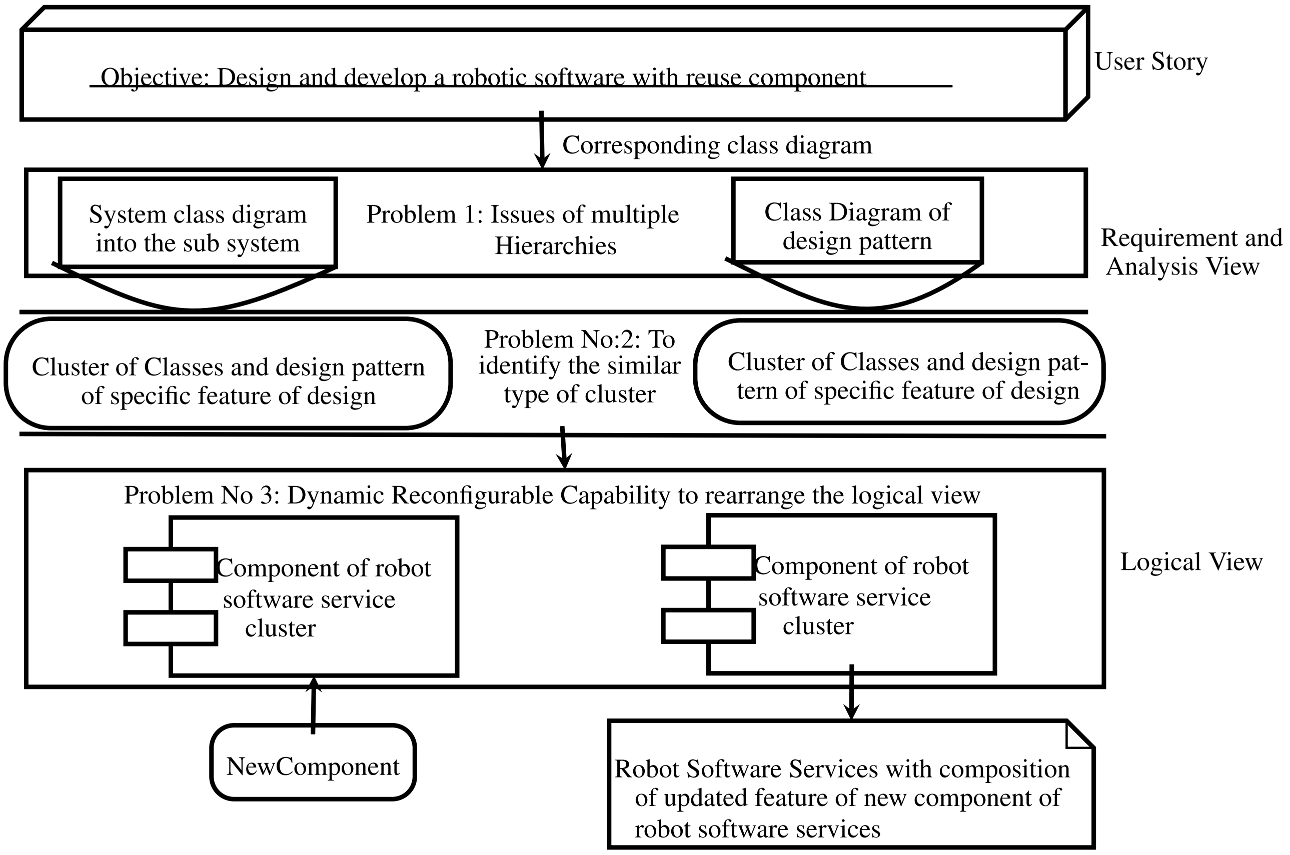
Objective: Design and develop a robotic software with reuse component
User Story
Corresponding class diagram
System class digram into the sub system
Problem 1: Issues of multiple Hierarchies
Class Diagram of design pattern
Requirement and Analysis View
Cluster of Classes and design pattern of specific feature of design
Problem No:2: To identify the similar type of cluster
Cluster of Classes and design pattern of specific feature of design
Problem No 3: Dynamic Reconfigurable Capability to rearrange the logical view
Component of robot software service cluster
Component of robot software service cluster
Logical View
NewComponent
Robot Software Services with composition of updated feature of new component of robot software services

Figure.1 Problem Formulation

3. Related Work

D.Brugali et.al.[3]tutorial part-I shows the scope of component based software engineering in several aspects of robotic application.They show that many software engineering concepts are useful for the development of robotic software services [15, 16].The efficiency and quality of robotic software component is expressed by computational performance or reliability.

According to D.Brugali 2009, the functional point of view, components are divided into the three categories: Application, Horizontal and Vertical. The horizontal components provide system services such as interface to the hardware devices. The vertical components consist specific functional unit such as kinematics, motion planning, deliberative control and address the prerequisite requirement of target applications domains such as service, space or humanoid robotics.

J.Poulin in 1996 observed that the vertical components have more weight. In historical data the reuse of vertical component, up to 65% of any application while horizontal components contribute not more than 25% and application components accounts for 15%.[17].

I.A.Nesnas suggests two level hierarchical robot capability layer in CLARAty project [18]: Decision Layer and Functional Layer. The synchronization of robot functionality and reusable components is a challenging design task. But in current situation implementation of reusable components plays a vital role to reduce the complexity of a real time application. In reference to the identification of implemented design patterns could be useful for the comprehension of an existing design and provides the ground for further improvements [19].

According to Gamma et.al. design pattern is description of communicating objects and classes that are customized to solve a general design problem in a particular context. Gamma et.al. suggested a catalog of design pattern by two criteria: Purpose and Scope. The purpose category defines what pattern does. Pattern can have either Creational, structural or behavioral purpose. The second criteria: Scope, specifies whether the pattern applies primarily to classes or to objects. Nikolaos Tsantalis et.al. suggest design pattern detection approach by using Similarity score algorithm[20].They suggested a testbed system as well as a design pattern to be identified in terms of graphs. For the identification/detection of pattern it uses a graph similarity algorithm [21]which takes the test systems and the pattern graph and estimates the similarity score between their vertices.As we aware about the demand of classical clustering techniques is one of the most relevant application domains of data mining and machine learning. Using clustering techniques several effective method[22, 23, 24]available to identify the logical components of business application domain. In existing literature there are several feature based classical clustering approaches are available to identify logical components. In one step up advancement S.M.H.Hasheminejad et al. suggest an evolutionary computation approach SCI-GA[25] for identification of logical software components with the aim of mapping the identification of component problem as a optimization problem. Clustering techniques have been helpful for extract the information by grouping the data of common features.

4. Mathematical background of similarity Algorithm

Blondel et al.[21] formulated a similarity score iterative algorithm for calculating the similarity between vertices of two different graphs by using the concept of authority and hub. The similarity matrix C is defined as suggested a generalization of the $n_A \times n_B$ matrix whose real entry C_{ij} expresses how similar vertex j (in G_A) is to vertex i (in G_B) and is called the similarity score between the two vertices , where G_A and G_B are two directed graphs with, n_A and n_B vertices respectively. The algorithm uses equation(1) for calculating the similarity matrix C.

$$C_{K+1} = \frac{A_{G_B} C_K A_{G_A}^T + A_{G_B}^T C_K A_{G_A}}{\| A_{G_A} C_K A_{G_B}^T + A_{G_B}^T C_K A_{G_A} \|} \tag{1}$$

For example consider the robat class diagram and design pattern given in figure 2 (a)& (b) for explanation of similarity score algorithm. Using steps defined by Blondel et al. we prepare a association and generalization graph and it is shown in figure 3. The graph of corresponding adjacency matrix of the Generalization graphs(GenA$_{G_{CD}}$,GenA$_{G_{DP}}$) and Association graphs(AssocA$_{G_{CD}}$,AssocA$_{G_{DP}}$) are as follows:

$$\text{GenA}_{G_{CD}} = \begin{array}{c|cccc} & RP & NL & GC & MB \\ \hline RP & 0 & 0 & 0 & 0 \\ NL & 0 & 0 & 0 & 0 \\ GC & 0 & 1 & 0 & 0 \\ MB & 0 & 1 & 0 & 0 \end{array}, \text{GenA}_{G_{DP}} = \begin{array}{c|ccc} & A & B & C \\ \hline A & 0 & 0 & 0 \\ B & 0 & 0 & 0 \\ C & 0 & 1 & 0 \end{array}$$

$$\text{AssocA}_{G_{CD}} = \begin{array}{c|cccc} & RP & NL & GC & MB \\ \hline RP & 0 & 0 & 0 & 0 \\ NL & 1 & 0 & 0 & 0 \\ GC & 0 & 0 & 0 & 0 \\ MB & 0 & 0 & 0 & 0 \end{array}, \text{AssocA}_{G_{DP}} = \begin{array}{c|ccc} & A & B & C \\ \hline A & 0 & 0 & 0 \\ B & 1 & 0 & 0 \\ C & 0 & 0 & 0 \end{array}$$

where RP=Robat Path,NL=NavigLeg,GC=Geometric Curve,MB=Motion Behavior

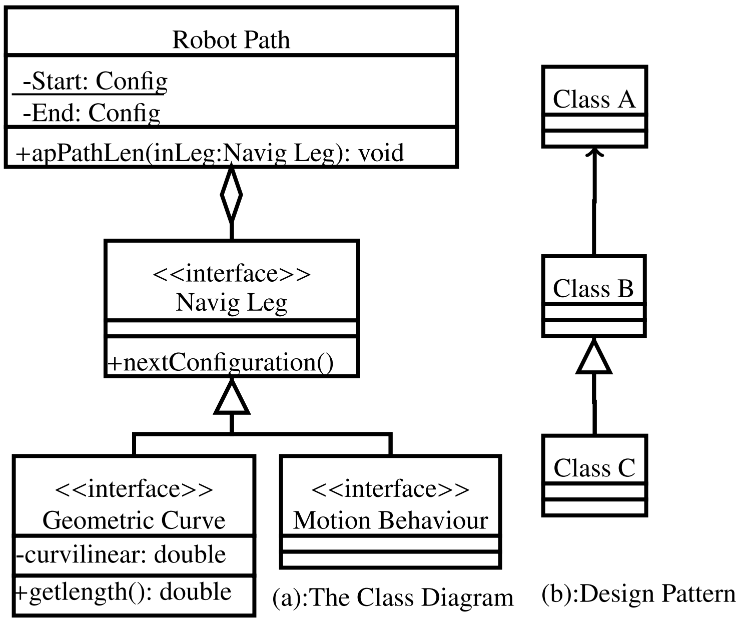

Figure 2: A simple case study of path planner robot

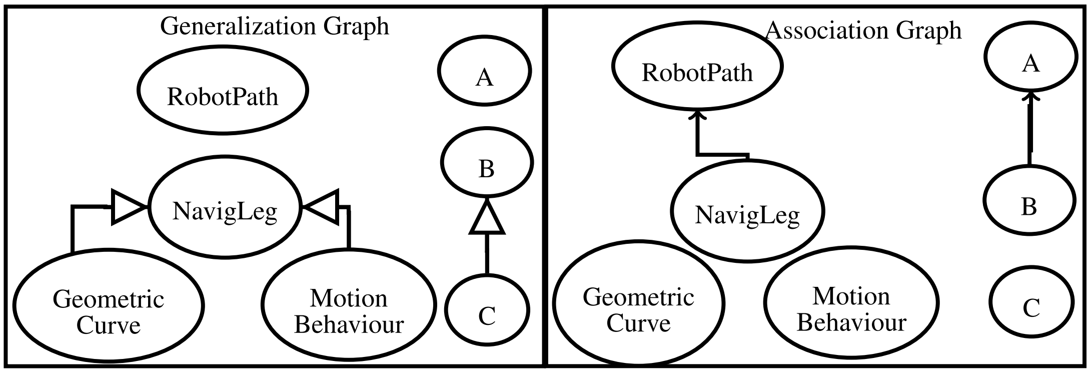

Figure 3. Generalization and Association graph of the Figure 2(a & B)

After that summing of both the generalization and association graph the normalized score are computed and it is given below.

$$\text{Sum}_{DP,CD} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1.33 & 0 \\ 0 & 0.33 & 0.33 \\ 0 & 0.33 & 0.33 \end{bmatrix}, \text{NormScores}_{CD,DP} = \begin{array}{r} \\ \textit{RobotPath} \\ \textit{NavigLeg} \\ \textit{GeometricCurve} \\ \textit{MotionBehaviour} \end{array} \begin{array}{c} \begin{array}{ccc} A & B & C \end{array} \\ \begin{bmatrix} 0.5 & 0.5 & 0.5 \\ 0 & 0.67 & 0 \\ 0 & 0.17 & 0.17 \\ 0 & 0.17 & 0.17 \end{bmatrix} \end{array}$$

Here,the normalized score matrix shows the strong similarity between NavigLeg and B Class.The normalized score of class pair(RobotPath,A) has reasonable score.The normalized score about the similarity between the implemented robot system class diagram and standard design pattern helps in reducing the time complexity as well as development cost.However,the similarity score algorithm couldn't handle the situation of class hierarchies of a large system.In the proposed approach Tasntalis et al. methodology is used for class hierarchies of a large system and Pearson & Heron-II et.al.method is used to find the similarity coefficient for binary data.The Pearson & Heron-II similarity coefficient is given in equation(2).

$$sim_{Pearson\&Heron-II}(A,B) = \cos\left(\frac{\pi\sqrt{bc}}{\sqrt{ad}+\sqrt{bc}}\right) \tag{2}$$

where, a is the number of variables present in both the object S_A& S_B having value 1. b is the number of variables present in object (S_A) and absent from object (S_B). c is the number of variables present in object (S_B) and absent from object (S_A) and d represents the number of variables absent from both the objects.Using the proposed algorithm we prepare corresponding generalization and association graph and it is shown in figure 4

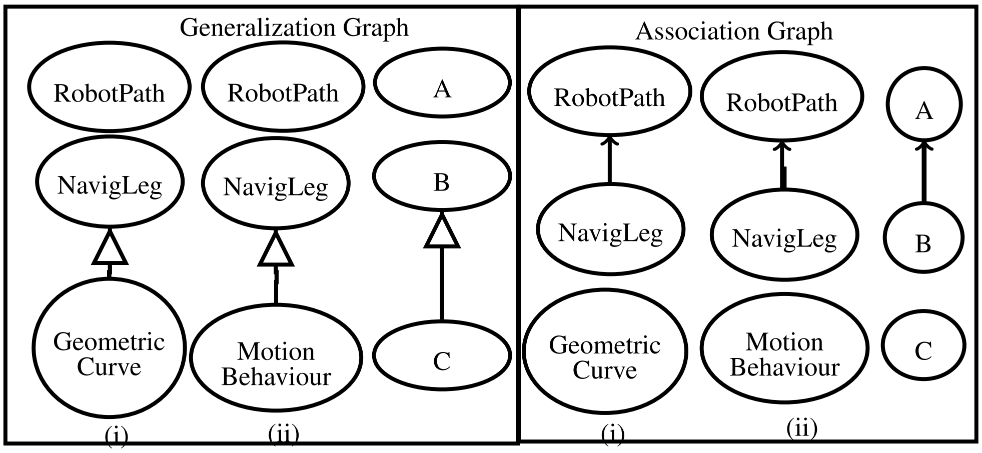

Figure 4. Corresponding graphs for the Figure 2(a & b)

For the computation of the parameter of similarity coefficient (a,b,c&d), we divide the system in subsystem.In figure 4, graph of the system is in consider as a same number of vertex as the design pattern graph.On the basis of adjacency matrix the value of a,b,c&d for the pair of genralization $GenA_{G_{CD}}$,$GenA_{G_{DP}}$ and Association $AssocA_{G_{CD}}$,$AssocA_{G_{DP}}$ are given in table 1.

Table 1. The parameter value for $G_{CD(i)}$(RobotPath)to G_{DP}(A) and $G_{CD(i)}$(RobotPath) to G_{DP}(B)

Pair	a	b	c	d
GenA $G_{CD(i)}$(RobotPath)-GenA G_{DP}(A)	0	0	0	3
GenA $G_{CD(i)}$(RobotPath)-GenA G_{DP}(B)	0	0	0	3

Similarly, we can compute the object value for other pair combination.Now the similarity coefficient for generalization graph is given below

$$\mathrm{Sim}_{CD(i),DP} = \begin{array}{r} \\ RobotPath \\ NavigLeg \\ GeometricCurve \end{array} \begin{array}{c} \begin{array}{ccc} A & B & C \end{array} \\ \begin{bmatrix} 1 & 1 & 0.66 \\ 1 & 1 & 0.66 \\ 0.66 & 0.66 & 1 \end{bmatrix} \end{array}$$

Here,the diagonal value in the similarity matrix represents the system class is exactly similar with the design pattern.By using the Pearson & Heron-II similarity coefficient can be identify the type of design pattern in subsystem structure.

5. Proposed Methodology

The steps of proposed methodology is given as follows:

1. Convert the software system into the subsystem of class diagram on the basis following property.
   (a) Atleast one generalization relation exist in the subsystem.
   (b) Maximum number of classes in subsystem is equivalent to the number of classes in design pattern.
2. Initialization of a adjacency matrix for all type of graphs is arranged in n×n adjacency matrix.Where n is the number of classes.
3. Prepare a cluster according to the catalog of design pattern.
   (a) Initialize:Centroid C(for each category of design pattern)=Sim(A,B). where, A is the adjacency matrix of system class diagram & B represents adjacency matrix of design pattern.
4. Calculate the similarity matrix using equation (2) for all types of graph pair(Class diagram and design pattern)
5. Identification of modified or new component for software system.
   (a) Initialize the adjacency matrix for new component of all relation graphs.
   (b) Repeat step (2)
   (c) Update the cluster with new subsystem.
6. Termination condition
   (a) If the number of iteration is greater than the number of clusters.
   (b) Otherwise Repeat step (2).

6. Result Evaluation

Similarity Score algorithm is useful for the identification of logical component for small software system where less no of adjacency matrix are available. However, the selection of iteration is (issue of convergence) challenging task.This similarity score algorithm for large system is not a better choice due to issues of multiple hierarchies in a large system which plays vital role in system class diagram.For identification of similar type of components the similarity score also depends upon the relation of classes of component.Blondel et al. similarity score algorithm has a gap to handle issues of hierarchies as shown in section 4. In case of normalized matrix we observe that the the component of path planner robot Navig and design pattern B class has similarity coefficient value as 0.67 that shows strong similarity between these two class. However Robatpath and A class has reasonable score 0.5. But by using proposed algorithm the similarity coefficient between Robat path and A class is exactly 1 and NavigLeg and B class also having 1 that shows improved value of similarity score if we compare each relation class diagram with design pattern diagram. In proposed algorithm clustering techniques are used for the multiple hierarchies which is divided into the combination of subsystem and clustered with the corresponding design pattern without loosing logical structure.In result the similarity coefficient matrix gives better precise component on the basis relational features of classes. The use of clustering approach also helps to handle the high dimensional data set as well as missing data.

7. Conclusion

The identification of the reuse software component of robot system is a challenging task if we are not using expert judgment and automated tool.The proposed approach helps to identify the similar type of components using catalog of design patterns and clustering techniques.This approach have the ability to handle the large software system having multiple inheritance classes.The proposed algorithm have ability to self organized and adjust the new arrive component according the similarity of design pattern.The approach also handle the issue of convergence using the total number of catalog of design pattern.

References

[1] Z-g Cai, X-h Yang, X-y Wang, and A Kavs: A Fuzzy-based Approach for Business Component identification. Journal of Zhejiang University-SCIENCE C (Computers & Electronics). 12(9):707-720, (2011). doi:10.1631/jzus.C1000337.

[2] D Birkmeier and S Overhage. On Component Identification Approaches Classification, State of the Art, and Comparison. In Proceedings of CBSE 2009, LNCS 5582, pages 1-18, 2009. doi: 10.1007/978-3-642-02414-6$_1$.

[3] Brugali, Davide, and Patrizia Scandurra. ”Component-based robotic engineering (part i)[tutorial].” IEEE Robotics & Automation Magazine 16.4 (2009): 84-96.

[4] I.Cmkovic,Component-Based Approach for embedded Systems,New York:IEEE Press,1994.

[5] Srivastava,k.Amit,and Kumar,Shishir.”An Effective Computational Technique for Taxonomic Position of Security Vulnerability in Software Development”Journal of Computational Science,https://doi.org/10.1016/j.jocs.2017.08.003.

[6] Verma, D. and Kumar, S., Exponential Relationship Based Approach For Predictions Of Defect Density Using Optimal Module Sizes, Proceedings Of National Academy Of Sciences Section A: Physical Sciences, vol. 82, issue 2, pp. 201-208, 2016.

[7] Verma, D. and Kumar, S., Prediction of Defect Density for Open Source Software Using Repository Metrics, Journal of Web Engineering, Rinton Press, vol. 16, issue 3 & 4, pp. 293-310, 2017.

[8] Verma, D. and Kumar, S., Empirical Validation of Defect Density Prediction Using Static Code Metrics, Accepted for publication in Advances in Information Sciences and Service Sciences, 2017.

[9] Verma Dinesh, and Shishir Kumar. ”An improved approach for reduction of defect density using optimal module sizes.” Advances in Software Engineering,4, 2014.

[10] Alexander, Christopher, Sara Ishikawa, Murray Silverstein, Max Jacobson, Ingrid Fiksdahl-King, and Shlomo Angel. ”A pattern language.” (1977).

[11] J Kim, S Park, and V Sugumaran: DRAMA: A framework for domain requirements analysis and modeling architectures in software product lines. The Journal of Systems and Software. 81(1):37-55, 2008. doi:10.1016/j.jss.2007.04.011.

[12] Gamma, Erich. Design patterns: elements of reusable object-oriented software. Pearson Education India, 1995.

[13] Kumar,Ajay and Kumar,Shishir. Experimental Analysis of Sequential Clustering Algorithm on Big Data, The IIOAB Journal SPECIAL ISSUE -RECENT ADVANCES IN BIG DATA ANALYSIS (ABDA)(2016), Vol.7, pp.8-18.

[14] P Kruchten. The Rational Unified Process An Introduction. 2nd ed. Addison Wesley, 2000.

[15] Brugali, Davide, ed. Software engineering for experimental robotics. Vol. 30. Springer, 2007.

[16] Brugali, Davide, and Erwin Prassler. ”Software engineering for robotics [From the Guest Editors].” IEEE Robotics & Automation Magazine 16.1 (2009): 9-15.

[17] Poulin, Jeffrey S. Measuring software reuse. Addison-Wesley, 1997.

[18] Nesnas, Issa. ”The claraty project: coping with hardware and software heterogeneity.” Software Engineering for Experimental Robotics (2007): 31-70.

[19] Vokc, Marek. ”An efficient tool for recovering Design Patterns from C++ Code.” Journal of Object Technology 5.1 (2006): 139-157.

[20] Tsantalis, Nikolaos, et al. ”Design pattern detection using similarity scoring.” IEEE transactions on software engineering 32.11 (2006).

[21] Blondel, Vincent D., et al. ”A measure of similarity between graph vertices: Applications to synonym extraction and web searching.” SIAM review 46.4 (2004): 647-666.

[22] Lee, Jong Kook, et al. ”Component identification method with coupling and cohesion.” Software Engineering Conference, 2001. APSEC 2001. Eighth Asia-Pacific. IEEE, 2001.

[23] Kim, Soo Dong, and Soo Ho Chang. ”A systematic method to identify software components.” Software Engineering Conference, 2004. 11th Asia-Pacific. IEEE, 2004.

[24] Shahmohammadi, Gholamreza, Saeed Jalili, and Seyed Mohammad Hossein Hasheminejad. ”Identification of System Software Components Using Clustering Approach.” Journal of Object Technology 9.6 (2010): 77-98.

[25] Hasheminejad, Seyed Mohammad Hossein, and Saeed Jalili. ”SCI-GA: Software Component Identification using Genetic Algorithm.” Journal of Object Technology 12.2 (2013): 3-1.