International Journal for Innovative Engineering and Management Research
A Peer Revieved Open Access International Journal
www.ijiemr.org

COPY RIGHT

ELSEVIER SSRN

IJIEMR Transactions, online available on 23rd Dec 2019. Link :http://www.ijiemr.org/downloads.php?vol=Volume-08&issue=ISSUE-12

Title: **RECOGNITION OF SOFTWARE DESIGN PATTERNS USING MACHINE LEARNING TECHNIQUES**

Volume 08, Issue 12, Pages: 188–193.

Paper Authors

V. SUJAY, DR. M. BABU REDDY

Krishna University,AP , India

USE THIS BARCODE TO ACCESS YOUR ONLINE PAPER

To Secure Your Paper As Per UGC Guidelines We Are Providing A Electronic Bar Code

International Journal for Innovative Engineering and Management Research
A Peer Revieved Open Access International Journal
www.ijiemr.org

RECOGNITION OF SOFTWARE DESIGN PATTERNS USING MACHINE LEARNING TECHNIQUES

[1]V. SUJAY, [2]DR. M. BABU REDDY
[1]Research Scholar, Department of CSE, Krishna University,AP , India
[2]Assistant Professor, Department of CSE, Krishna University,AP,India

Abstract:
Software architecture centric development and design patterns are the reusable code that offers effective solutions to frequent design issues. In the reverse engineering process, though, it is often required to recognize and identify design patterns from source code. In recent study, Recognition of software design patterns based on ML Techniques is identified. Firstly, the development of a preparation data set based on software measurements. For object identification processes ML Techniques such like LRNN and Decision Tree are extended.
Keywords: Design Pattern, SDP, Machine Learning, Reusability

1. Introduction

The pattern of Software architecture design is a set of methods used to transfer information across a specific domai n.A significant number of software trends fo r the design of a specific system are identifie d in the research.A variety of methods and te chniques are used to identify such patterns.T o improve the development of the relevant c ode, software patterns are useful. Software p attern design is often obtained by using patte rn model concepts that identify the solution t o the recurring problem of design.

Gamma et al. generally referred to as GoF uses the process framework for their st ructure designs. The patterns catalogue is structured into 3 ways:

1. Creational,
2. Structural
3. Behavioural Patterns.

These three software design patterns also to help to develop Software Maintenance by creation unambiguous account of class and object relations.

In the field of forward growth, the standard of arrangement structures is seen, where assortments have been used as a solution for their reiterated arrangement issues. A technique orchestrated procedure relies upon the characterizations of its people which serve variety employments during programming improvement. Barely any models have similar structure to instances of state and system.

Presently, the amount of patterns is growing, which exists in the function of a huge numb er of patterns. Moreover, the process of ident ification may be not based on important patt erns.

2. Literature Review

A huge number of methods for pattern recognition available and Different methods are based on continuous analysis. When two or more patterns have the same architecture.

Tsantalis et al. formulated a design p attern recognition technique that requires co nsistency rating methods at the vertices of a diagram that shows a design pattern structur e.

Dong et al.introduced the technique of design correlation to determine design pat

International Journal for Innovative Engineering and Management Research
A Peer Revieved Open Access International Journal
www.ijiemr.org

terns of software projects by calculating thei r standard cross similarity.

Pradhan et al.have applied two techniques, for example, diagram isomorphism and standardized cross relationship for the identification of programming configuration designs.

Di Martino and Espositon,proposed a software machine learning technique. Web ontology language (OWL) was used to ident ify variations.

3. Software Architecture Centric Development and Design Patterns Techniques

3.1 Software Architecture Centric Development [SACD]:

Architecture-centric development technique in software platform is accurate and cost-effective. In terms of the quality factor and time to market products, usually used techniques in software development are very expensive and unreliable In terms of quality and delivery of the products.

We can reuse parts from each design phase to optimize production time and to provide better quality in order to decrease development time. For the creation of new software previously designed, verified and functional parts to be reused.

In software development, the software industry is growing and adopting new approaches to enhance software quality, speed and efficiency. To SDLC, software development processes are used to obtain, evaluate, model, execute, check, and maintain parameters. The impact of the improvement in specifications reduces the chance of uncertainty.

Requirement collection process is a very time-consuming phase and a lot of effort is expected when obtaining standards as criteria are the backbone of any software if the standards are not properly collected this will lead the system to failure and if the expectations are incorrectly collected; there will be no use of software Consequent phases involved:

1. Requirement Elicitation
2. Analysis
3. Verification

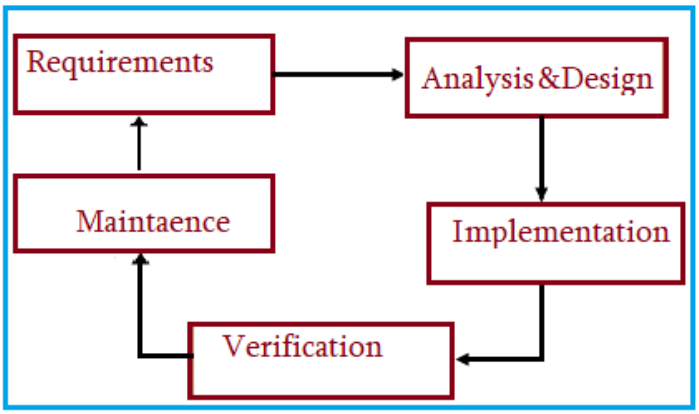


Fig3.1.Conventional Method to Soft ware Architecture Centric Development

3.2 .Software Design Pattern

Abstract Factory Pattern:

AFP is mainly used to offer an interface for build families build families of similar (or dependent) artefacts without identifying their relevant subclasses. Abstract factory and adapter patterns are perceived to find patterns from raw data.

The pattern includes of four participants including Abstract Factory, Concrete Factory, Abstract Product and Concrete Product.

International Journal for Innovative Engineering and Management Research
A Peer Revieved Open Access International Journal
www.ijiemr.org

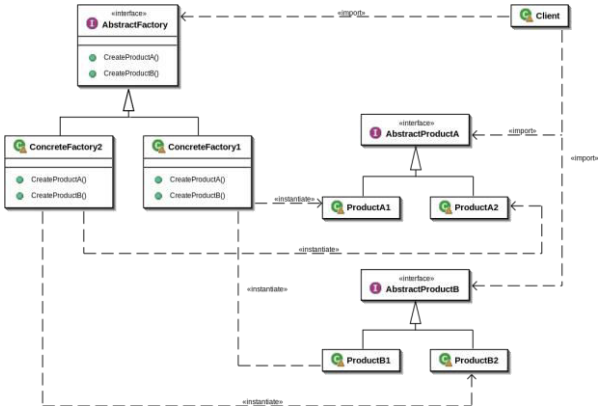


Fig.3.2.1.a.Abstract Factory Pattern

Advantages:

- Application software protects from r elevant (acquisition) classes.
- It enhances a trade of group objects
- Stimulates accuracy around objects.

Adapter Pattern:

This approach is easy to recognize as there are several adapters in the real world. Consider, for example, a USB adapter for Ethernet.We needs this if we have an Ethernet port on one side and a USB interface on the other. Since they are mutually incompatible. We're using an adapter to convert one to another.

Advantages:

- Allows being reusable and versatile.
- User class doesn't have to use a speci fic device and can use gene expressio n to swap

  Between various adapter modules.

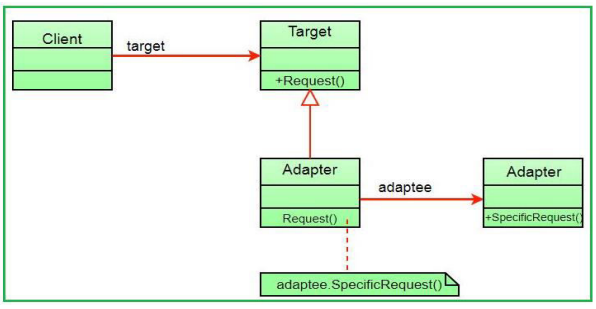


Fig.3.2.1.a.Adapter Pattern

4. Proposed work

The proposed is categorized into two sections, like pattern- based data set planning and software design pattern recognition.

1. Preparation of Training Dataset

Raw data is prepared in this process to train the Optimization Algorithms used in this study. Training data set design contains four other subprograms such as Identifying Software Patterns, Specifying Instructor Patterns, Creating OO Criteria-Based Attribute Vectors, and Performing Pattern-Based Data Pre-Processing.

Definition of Design Patterns

A design pattern continually calls, en ables and outlines a specific architecture in o bject

oriented structures that solves a persistent de sign issue.

Selection of Patterns Participants

The design pattern is a set of one or more sections, also known as the participant's pattern. In a technique-based system, these member classes perform a specific role. Such features require a specific verification during design pattern recognition.

Preparation of OO Metrics-Based Feature Vectors

It can be prepared by using various software pattern detection tools such as similarity scoring algorithm.

Pre-processing of Metrics-Based Dataset

Pre-processing of metrics-based dataset is carried out before learning process

2. Recognition of Software Design Pattern

Binary descriptive statistics are trained in this phase to understand design patterns. It determines whether or not

International Journal for Innovative Engineering and Management Research
A Peer Revieved Open Access International Journal
www.ijiemr.org

selection of pattern members in software are instance of the real patterns of the application. The design pattern recognition process requires three subprograms, like statistics set learning, software pattern recognition, and outcomes cross-validation.

I. Learning of Metrics-Based Feature Vectors:

II.Design Pattern Recognition Process

III.Validation for Result conformance

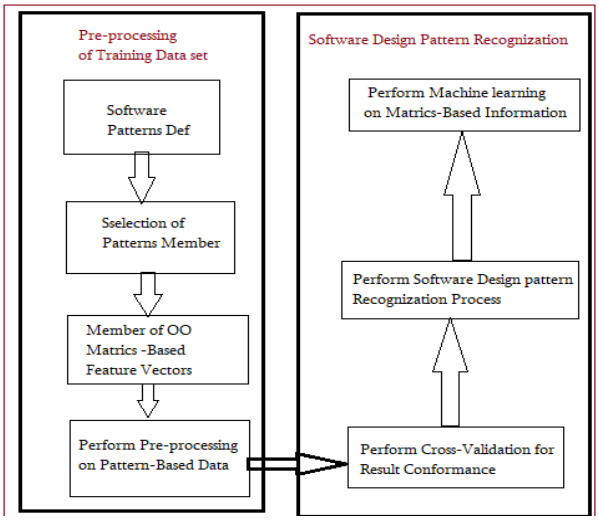


Fig.4.1Software Design Pattern Recognition Model

5. Results

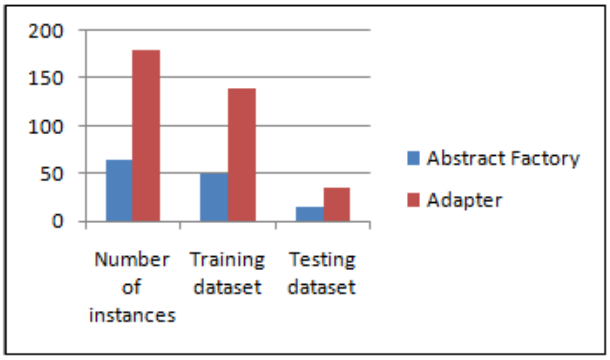
The open source Code is used to identify software design patterns in a number of experiments. Many such pattern examples are classified as training data set as well as training data as shown in Table. An analysis of the proposed research is also performed using a pattern repository, during the test process the archive includes nine open source software.

TABLE

Dataset before Learning Process

Software patterns	Number of instances	Training dataset	Testing dataset
Abstract Factory	65	50	15
Adapter	180	140	35

Fig.5.1.Graphical Representation of Dataset before Learning Process



6. Conclusion

In this paper, we are using learning-based techniques like LRNN and Decision Tree, the proposed concept Machine Approaches. To find out various Object-Oriented Metrics Are Recognized For The Preparing Of Metric-Based Datasets. The design pattern recognition process using the design patterns of Abstract Factory and Adapter. In future work enhance the advanced Machine Learning Techniques.

References

[1] E. Gamma, R. Helm, R. Johnson, and J. Vlissides, Design patterns: Elements of Reusable Object-Oriented Software. Addison-Wesley, 1995.

[2] A. K. Dwivedi and S. K. Rath, “Incorporating security features in service-oriented architecture using security patterns,” ACM SIGSOFT Software Engineering Notes, vol. 40, no. 1, pp. 1–6, 2015.

[3] M. Fowler, Patterns of enterprise application architecture. Addison-Wesley, Boston, USA, 2002.

[4] H. Zhu and I. Bayley, "On the compos ability of design patterns, "Software Engineering, IEEE Transactions on, vol. 41, no. 11, pp. 1138–1152, 2015.

[5] N. Tsantalis, A. Chatzigeorgiou, G. Stephanides, and S. T. Halkidis, "Design pattern detection using similarity scoring," Software Engineering,IEEE Transactions on, vol. 32, no. 11, pp. 896–909, 2006.

[6] J. Dietrich and C. Elgar, "Towards a web of patterns," Web Semantics: Science, Services and Agents on the World Wide Web, vol. 5, no. 2, pp.108–116, 2007.

[7] A. K. Dwivedi, A. Tirkey, and S. K. Rath, "An ontology based approach for formal modeling of structural design patterns," in Contemporary Computing (IC3), 2016 Ninth International Conference on. IEEE, 2016, pp. 208–213.

[8] M. Zanoni, F. A. Fontana, and F. Stella, "On applying machine learning techniques for design pattern detection," Journal of Systems and Software, vol. 103, pp. 102–117, 2015.

[9] A. K. Dwivedi and S. K. Rath, "Formalization of web security patterns,"INFOCOMP Journal of Computer Science, vol. 14, no. 1, pp. 14–25, 2015.

[10] J. Dong, Y. Zhao, and T. Peng, "A review of design pattern mining techniques, "International Journal of Software Engineering and Knowledge Engineering, vol. 19, no. 06, pp. 823–855, 2009.

ABOUT THE AUTHORS

V. Sujay received his M.Tech. Degree from SRM University and is currently pursuing Doctor of Philosophy at Krishna University, India with Computer Science and Engineering specialization. He has been actively involved in teaching for the past 7 years and now he is working as Guest Faculty of Computer Science and Engineering at Sri Krishnadevaraya University, Anantapur, Andhra Pradesh, India. His research interests, include Software Engineering, Database Management System, C Programming, etc.,

Dr. M. Babu Reddy received his Master's Degree and Doctor of Philosophy from Acharya Nagarjuna University, India with Computer Science and Engineering specialization. He has been actively involved in teaching and research for the past 18 years

and now he is working as Assistant Professor of Computer Science at Krishna University, Machilipatnam, Andhra Pradesh, India. His research interests, include Machine Learning, Software Engineering, Algorithm Complexity Analysis, and Data Mining.