oDyRM: Optimized Dynamic Reusability Model for Enhanced Software Consistency

R. Selvarani
Professor, Dept. of CSE
ACED Alliance University
Bengaluru, India

P. Mangayarkarasi
Research Scholar
Visvesvaraya Technological University
Belgaum, India

***Abstract*—Accomplishment of optimization technique on Object Oriented design component is a very challenging task. The prior model DyRM has introduced a technique to perform modeling of design reusability under three real-time constraints. The proposed study extends the DyRM model by incorporating optimization using multilayered perception techniques in neural network. The system takes the similar input as is done by the prior DyRM, which is subjected to Levenberg-Marquardt optimization algorithm using multi-layer perceptron of configuration 4-24-2 to generate the optimal output of consistency factor. The paper discusses the underlying technique elaborately and presents the outcome that shows a good curve fits between experimental and predicted data. The model is therefore termed as optimized DyRM (oDyRM) to evaluate the consistency factor associated with the proposed model.**

Keywords*—Cost; Back propagation Algorithm; Design Reusability; Object Oriented Design; Optimization; Project management

I. Introduction

The study carried out in software reusability and software consistency, models and metrics suggests that it potentially benefits the clients from economic and performance perspectives. In the software industry the term 'reuse' is associated with cost efficiency for improving software development processes [1]. Various models for validating the preciseness of the design process depends on accuracy of results accomplished from a model that is directly proportional to the input data [2], [3], [4]. Accuracy is closely associated with reality. However, results may not be always accurate and hence sensitivity analysis is carried out. In order to gauge the level of accuracy and the factors affecting it, the study considers mathematical modelling for the purpose of optimization of design reusability. Mathematical modelling proved to be useful for validation and verification of the Software Reusability metrics. The other benefits of the software metrics are i) Development Benefits, ii) Maintenance, iii) Quantification of benefits and cost validation iv) and Use of economic models for validation. Economic models of reuse can help in making decisions concerning reuse and its applicability to address problems of uncertainty. The proposed study is an extension of the prior study where the enhancement is being carried out using optimization principle. The proposed system uses neural network to perform optimization and retrieve the consistency of the proposed software reusability model. Section 2 discusses about the related work followed by discussion of problem identification in Section -3. Section -4 discusses about proposed model followed by research methodology in Section-5. Implementation of proposed model is discussed in Section-6 followed by result discussion in Section-7. Finally, Section-8 makes concluding remarks.

II. Related Work

Many significant studies in the area of software engineering focus on reusability aspects as well as software consistency for study. The study introduced by Nair and Selvarani [5] presented a framework with the capability to compute the reusability index. The authors considered three of the Chidamber and Kemerer metrics viz. DIT (Depth of Inheritance Tree), RFC (Response for a Class) and WMC (Weighted Methods per Class). Same authors also carried out a complete analysis of the relationships that exist between internal quality attributes in terms of the complete suite of Chidamber and Kemerer metrics and the reusability index of software systems [6]. They presented a new regression technique for mapping the association between reusability and design metrics. Das et al. [7] studied about the mitigation techniques for the errors in the software modelling. They carried out the simulation study based on continuous time factor on case study of flight control software. Gargoor and Saleem [8] adopted swarm optimization technique along with neural network and exhibit better predictive capabilities to analyze software consistency issues. Strong et al. [9] adopted statistical methods to enhance the software consistency factor.

Emphasis on software consistency laid by Wason et al. [10] state the significance of automata-based approach. Anjum et al. [11] proposed a soft computing based technique using Poisson process to evaluate the software consistency. Similar direction of study also carried out by Yakonoyna et al. [12]. Sabbineni and Kurra [13] implemented a dynamic technique for the purpose of consistency allocation of software components. Sheakh [14] presented an enhanced algorithm to improve the performance of software consistency. However, the extent of the author's contribution is found to be poorly discussed with less evidence to prove its efficiencies. Kumar also carried out by Antony [17]. Fetaji et al. [18] and Singh et al. [19] also carried out empirical investigation towards improving reusability as well as software consistency on the object-oriented design components.

III. Problem Description

The identified problems after reviewing the existing research contribution towards software reusability are i)

Existing study emphasizes on code reusability and not design reusability, ii) Existing techniques of software reusability doesn't consider essential attributes of project management e.g. human resources, skill gap, requirement volatility, training, cost of new development, etc., iii) Less extent of optimization of the design reusability from the OO component design is found, iv) The studies using CK metric discussed by various authors are found with theoretical assumptions. Majority of the studies considering CK metrics manipulates the same formulations with minor concern to introduce practical scenarios, v) Few consideration or attempt to model real-time constraints are found in existing literatures, hence the outcomes of the model are more inclined to hypothetical figures and less possibilities of applicability with real-time requirements, and vi) Software reusability as well as software consistency is not found to be combine studies. Modern day software development methodologies encounter more dynamicity, uncertainties, and unforeseen possibilities of failures of projects. Such issues cannot be addressed by theoretical and hypothetical framework of software consistency, which is found to be less connected with software reusability in existing system. Hence, all the above problems are highly critical and invoke various issues while attempting novel modeling of software reusability management. There is a need of designing a model considering presences of various uncertainties are errors to closely check the efficiency level of the solution. Such critical emphasis was never found in the literatures till date and hence calls for addressing the same. The next section will present a solution towards this problem:

IV. PROPOSED SYSTEM

The prior model named as Dynamic Reusability Model (DyRM) created a fundamental base for establishing a relationship among the CK metrics (CBO, RFC, WMC, DIT, and NOC) and design reusability [20]. DyRM was basically designed for the purpose of evaluating the impact of design reusability in software engineering under three real-time constraints e.g. quantity, work schedule, and cost of new development. Owing to various possible uncertain scenarios (e.g. requirement volatility, change management, skill gap, attrition rate), there is a possibility that the outcome of DyRM model may be associated with significant errors that are hard to find. Hence, there is a need of an optimization technique to our prior DyRM model for achieving following benefits e.g. i) Inconsistency reduction in DyRM leading to better predictability of the reusability outcomes, ii) Parallel computation of multiple and heterogeneous constraints-based reusability estimation in software project developments, and iii) Predictive optimization with assured consistency and robustness in future use. Hence, the development of a novel predictive optimization technique over DyRM model for enhancing the reusability management of software projects to greater extent. It is also applicable to overcome all the unseen constraints to a large extent that are not considered in this model. Hence neural network multi-layered perceptron is applied for developing the proposed predictive optimization principle. The technique allows for multiple forms of input to the processor in the form of real time constraints, which after processing gives the output of consistency score and uniformity score. The proposed system oDyRM is implemented in a typical way as exhibited in Fig.1

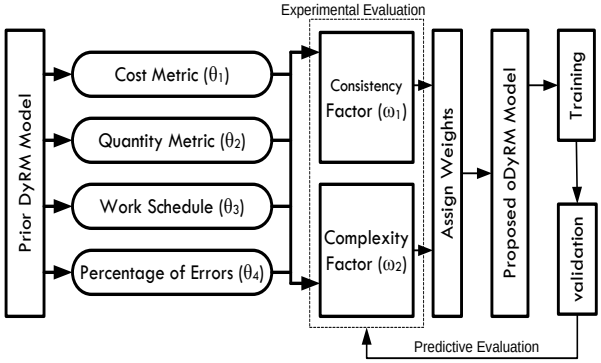

Fig. 1. Schema of Proposed Implementation (oDyRM)

V. RESEARCH METHODOLOGY

The proposed system adopts analytical research methodology, where the prime target to check for level of consistency for prior DyRM model after being subjected to predictive optimization. Similar assumptions and problem formulation discussed in design principles of DyRM model is continued in proposed system also. Apart from inclusion of older forms of 3 inputs of design attributes (or constraints) i.e. i) quantity, ii) work schedule, and iii) cost of new development, the proposed model also considers a new input attribute i.e. percentile of error for performing analysis for optimal consistency. The rationale behind selection of 4th design attribute of error is – DyRM chooses only three input attribute which definitely lowers the scope of design reusability value in sophisticated software projects. There is also a possibility of many other input attributes (or constraints) that equally impacts design reusability computation e.g. skill gap, requirement volatility, change management, etc. Such abstract parameters combine to have negative impact of design reusability computation and hence may eventually affect the optimization process. Therefore, we consider such parameters as percentile of error, whose values are defined between 1-4 depending on total numbers of inputs. The outcome of optimization is valued with respect to consistency factor and complexity factor. Fig.2 shows. The objective is achieved by developing a multi-layered perceptron for predicting the consistency factor in reusability model. Investigation was conducted to determine the strength of design metrics in the form of consistency after 100 iteration rounds testified with various parameters like quantity, work schedule, cost of new development, and percentage of errors. A total of 300 permutations of 4 input variables were developed. Out of these data sets, certain data sets (80% of total data) were used for training and the remaining data sets were used for validation. The input and output vectors have been normalized in the range (0, +1) using suitable normalization factors or scaling factors. The following input parameters were selected to predict the consistency factor.

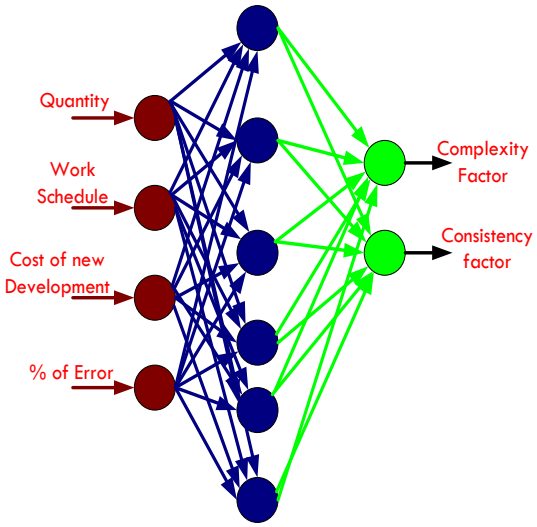

Fig. 2. Inputs and Outputs of oDyRM

- **Cost (θ_1)**: This metric estimates the cost for new development of the reusable design of projects from prior DyRM model. The values are further randomized within a scope to generate cost involved various projects.
- **Quantity (θ_2)**: This metric estimates the number of the projects with reusable designs (with higher values of θ_1) and further randomized statistically.
- **Work Schedule (θ_3)**: This metric evaluates the optimal time required to dispatch a particular projects (with higher values of θ_1 and$\theta_{2)}$.
- **Percentage of Errors (θ_4)**: This is the newly introduced metric that introduces random errors that may be possibly caused due to various extrinsic factors in software development methodologies.

Based on the input parameters selected, the proposed model was formulated as

$$IP_{oDyRM} = [\theta_1, \vartheta_2, \theta_3, \theta_4] \quad (1)$$

The following output parameters were selected to be predicted from the network model e.g.

- **Consistency Factor (ω_1)**: This factor statistically evaluates the extent of consistency after adopting inputs specified in eq.(1). The mathematical representation of novel consistency factor is,

$$\omega_1 = \sum \frac{IP_{oDyRM} - 0.1}{0.8} . \Delta wt \quad (2)$$

The above eq.(2) considers inputs from eq.(1), generated weight *wt* and considers the lower limit of 0.1 and higher limit of 0.8 in probability theory. The system chooses to consider 0.8 as accomplishing higher consistency factor of 0.9 and 1 is impractical assumption in probability theory.

- **Complexity Factor (ω_2)**: This factor checks the uniformity of the generated values after performing validation to check the consistency of the consistency factor (ω_1).

VI. Algorithm Implementation

The system mainly attempts to ensure better reusability management by enhancing consistency factor. The implementation of the proposed system starts by evaluating cost metric, quantity metric, work schedule metric, and percentage of error metric from prior DyRM model, which is statistically subjected to evaluation of consistency factor. As the enhancement of prior DyRM model is carried out using multi-layered perceptron, hence, it is important to assigned weight. Adopting the techniques of multi-layered perceptron, the proposed system has only three layers (input, hidden, and output layer). Hence, the configuration of the oDyRM model is: 4-24-2, where 4 is the number of inputs nodes, 24 is the number of hidden nodes and 2 is the number of output nodes. The calculations of the weight to be allocated is done as number of weights to be determined (4x24) + (24x2) =144. The computational algorithm used for the training is as follows:

Algorithm for Optimizing ω_1 andω_1

- **Input:** IPoDyRM, wt,
- **Output:** consistency adoptability factor (ω1), complexity factor (ω2)
- Initialize IPoDyRM, wt, Neurons
- Data Obj ← Read(Datafilefor Training)
- Compute, Min/ Max (Data Obj), → Normalization for Min/ Max ← output
- Set Epoch, Initialize Neural Network NLf() / Layers, Train, Test , error
- Evaluate consistency adoptability factor (ω1), complexity factor (ω2).

The algorithm is designed using the similar concept in multilayer perceptron, where the inputs are given as Cost Metric (θ_1), quantity metric (θ_2), work schedule metric (θ_3), and percentage of error metric (θ_4). The inputs are fed for processing and furnished an output as consistency adoptability factor (ω_1) and complexity factor (ω_2). The training was for 2000 iteration and checked for the curve-fitting using neural network in numerical computing simulation for both experimental and predicted data. The algorithm mainly took less than 3.5 seconds to execute and outcomes are discussed in next section.

VII. Results and Discussion

Firstly, the technique of accomplishing the data as well as processing the data for claiming the optimization in design reusability concept is discussed in this section. Case studies of two software projects of Enterprise Resource Planning (ERP) which are mainly open source are taken namely Apache OFBiz [21]. The OFBiz is configured on user machines. The classes were re-configured related to asset management, human resource, accounting, and inventory management etc. to have real-time environment of ERP application. A plug-in Metrics

1.3.6 [22] is used for extracting the CK metrics which is used in prior DyRM model i.e. CBO, RFC, WMC, DIT, and NOC. The DyRM model is used to estimate design reusability under multiple iterations along with new inclusion of 4th new input parameter i.e. percentage of error. The outcomes are recorded in comma delimited file to use as an input to Statistical analysis applications such as SPSS etc. The t-test and analysis of variance is performed in SPSS to observe the statistical outcome, which is further arranged in the form of 4 input parameters in numerical computation to carry out training and validation phase. The system considers the input from the SPSS where the empirical analysis for the 4 design parameters of the oDyRM model is considered (cost, quantity, work schedule, and percentage of error). Table 1 highlights the outcome after 2000 epochs of training period. This optimization process uses all forms of non-linear input data that may eventual lead to non-linear optimization problems. Hence, the training is carried out using Levenberg-Marquardt algorithm for solving non-linear squares problems.

TABLE. I. DETAILS OF SCALING FACTORS

Nature of vector	Parameter	Minimum Value	Maximum Value	Scale Factor
Input Vector	Cost Metric	0.3	0.5	0.7
	Quantity Metric	2	3	4
	Work Schedule	1	3	3
	Error	1	4	4
Output Vector	Consistency Factor	0.6	0.8	1.3
	Complexity Factor	4.9	13.8	18.7

The table highlights basically two types of information i.e. i) experimental data (before training) and ii) predicted data (after training). The experimental data is being calculated using SPSS, which after feed to the training module generates the predicted data in numerical computation. Both experimental and predicted data shows the higher rate of data consistency as a part of validation test in multi-layer perceptron based error reduction. The higher consistency factor (near to 1) accomplished by the training state confirms the robustness of the proposed optimization model using probability theory. In order to accomplish the optimal outcomes, the proposed system used a scale factor which is calculated as (MinVal+MaxVal)-1, which in other sense refers to the degrees of freedom to evaluate the outcomes. Table 1 gives the details of the input vectors and output vectors. The model considers neural network parameters as number of nodes, nodal properties corresponding to the input as well as output vectors along with hidden layers. The training process adjusts the epoch in run time as per the error. It was observed that there are 24 neurons present in hidden layer. Hence, the configuration chosen for the proposed model is 4-24-2. The proposed system uses backpropagation algorithm to evaluate its weight depending on the used gradient search technique after generation of the weight-factors in SPSS. We define the number of weight as 144 and scale the weights using SPSS. After the training is accomplished, it is found that expected outcomes as well as actual outcomes are highly matching with each other, as seen in Fig.3.

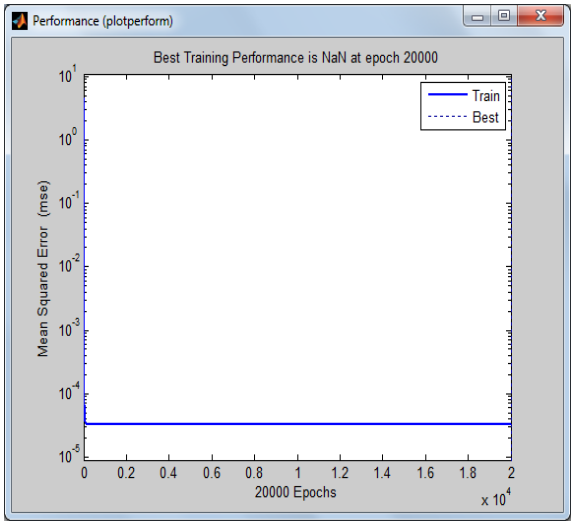

Fig. 3. Performance of oDyRM

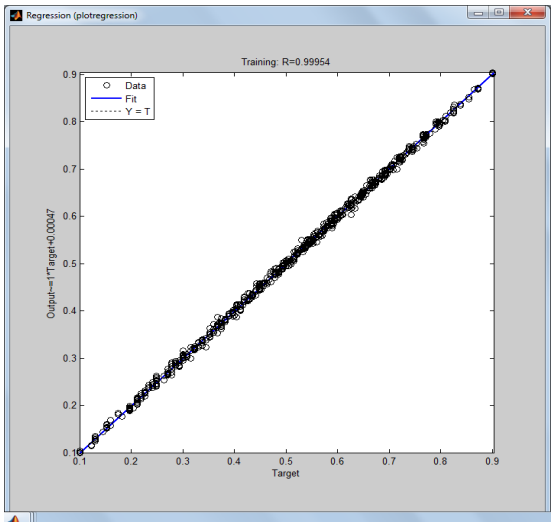

Fig. 4. Regression Analysis of Proposed eDyRM

Fig.3 highlights the regression analysis performed on proposed eDyRM model. The outcome is found to have the best fit with the training data as well as better uniformity in the error outcomes as linear and deterministic trend of error curve. The regression analysis is also performed for RMS value (Fig.4) with the increasing number of epoch. The duration of the epoch is from 100-2000.

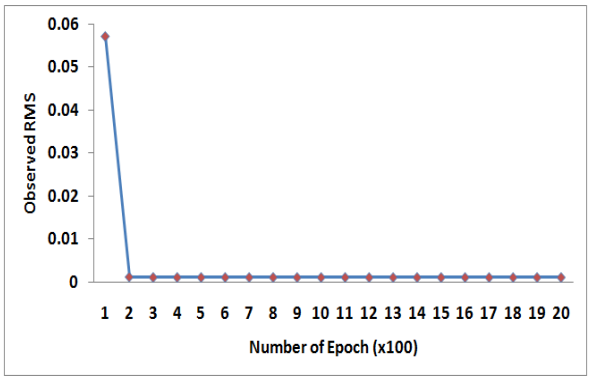

Fig. 5. Analysis of RMS

The optimization principle adopted in the proposed oDyRM model ensures the better solution towards enhancing normal gradient descent as well as drastic minimization of RMS values (Fig.5). Hence, the proposed technique has the faster convergence rate irrespective of the variances in the input data size. The proposed technique is also flexible for incorporating more empirical methods for testifying the reusability metrics pertaining to OO design optimization in software engineering. The extent of design complexity is always a matter of worry in when it comes to design reusability. Even with higher values of epoch (>2000), the outcomes were found with similar trend of consistency.

VIII. CONCLUSION

Design reusability is one of the prime concerns in software engineering especially when working on complex Object Oriented software components. The proposed paper has enhanced the prior model DyRM. The enhancement includes evaluating the consistency factors and consistency factor of proposed optimized model oDyRM using multilayered perceptron approach of neural network. The system takes the input of cost of new production to generate reusable design components, quantity of the projects to be delivered, flexibility in the work scheduling, and percentage of possible errors. The outcome of the study shows that the model is able to check for error, reduce it recursively till it gets best curve fitting for the trained and real-data. Hence, the proposed model can be used by any technical architect to evaluate the possible risk or gain to adopt a particular software development methodology to measure the level of effectiveness in design reusability and software consistency.

REFERENCES

[1] Wentzel, K.D.: Software Reuse - Facts and Myths. Proceedings of the 16th international conference on Software engineering. IEEE Computer Society, 267-268 (1994)

[2] Vanmali, M., Last, M., Kandel, A.: Using a Neural Network in the Software Testing Process. International Journal Of Intelligent Systems, 17, 45–62 (2002)

[3] Musilek, P., Meltzer, J.: Assessing Empirical Software Data With Mlp Neural Networks. ICS AS CR, (2005)

[4] Adebiyi, A., Arreymbi, J., and Imafidon, C.:Security Assessment of Software Design using Neural Network. International Journal of Advanced Research in Artificial Intelligence. 1, 4, (2012)

[5] Nair, T.R.G and Selvarani R.: Estimation of Software Reusability: An Engineering Approach. Association for Computing Machinery (ACM) – SIGSOFT. USA, 35, 1 (2010)

[6] Selvarani, R., Nair, T.R.G.: Software Reusability Estimation Model Using Metrics Governing Design Architecture, International Book: "Knowledge Engineering for Software Development Cycles: Support Technologies and Applications", Engineering Science Reference, IGI Publishing, USA (2009)

[7] Das, S., Dewanji, A., Sobti, A.: Software Reliability Modeling with Periodic Debugging Schedule. Indian Statistical Institute (2013)

[8] Gargoor, R. G. A., Saleem, N. N.:. Software Reliability Prediction Using Artificial Techniques. IJCSI International Journal of Computer Science Issues. 10 (4) (2013)

[9] Strong, K.: Using FMEA to Improve Software Reliability. In Pacific Northwest Software Quality Conference (PNSQC) (2013)

[10] Wason, R., Ahmed, P., & Rafiq, M.Q.: Automata-Based Software Reliability Model: The Key to Reliable Software. Int. J. of Software Engineering & Its Applications. 7(6), 111-126 (2013)

[11] Anjum, M., Haque, M.A., & Ahmad, N.: Analysis and ranking of software reliability models based on weighted criteria value. International Journal of Information Technology and Computer Science (IJITCS), 5(2), 1 (2013)

[12] Yakovyna, V., Fedasyuk, D.,.Nytrebych, O., Parfenyuk, I., Matselyukh, V.: Software Reliability Assessment Using High-Order Markov Chains. International Journal of Engineering Science Invention. 3(7), 1-6 (2014)

[13] Sabbineni, S., & Kurra, R.:Estimation of Reliability Allocation on Components Using a Dynamic Programming. International Journal of Computer Science Issues (IJCSI). 10(3) (2013)

[14] Sheakh, T.H.: An Improvised Algorithm for Improving Software Reliability. International Journal of Computer Applications. 79 (17) (2013)

[15] Kumar, D., Dinker, A. G.: Enhancement of Reliability in Object-Oriented Software Reliability Model. International Journal of Advanced Research in Computer Science and Software Engineering. Vol. 4 (2014)

[16] Kapila, H., & S.Singh.:Analysis of ck metrics to predict software fault-proneness using bayesian inference. International Journal of Computer Applications. 74 (2013)

[17] Antony, P.J.:Predicting Reliability of Software Using Thresholds of CK Metrics. International Journal of Advanced Networking & Applications. 4(6) (2013)

[18] Fetaji, B., Reci, N., & Fetaji, M.:Analysing and Devising a Model for Trustworthy Software. Recent Advances in Electrical and Computer Engineering. (Retrived 2015)

[19] Singh, P. K., Sangwan, O.P., Singh, A.P., and Pratap, A.: A Quantitative Evaluation of Reusability for Aspect Oriented Softwareusing Multi-criteria Decision Making Approach. World Applied Sciences Journal. 30 (12) 1966-1976 (2014)

[20] Selvarani, R., and Mangayarkarasi, P.:A Dynamic Optimization Technique for Redesigning OO Software for Reusability. SIGSOFT Softw. Eng. Notes. 40 (2) 1-6. (2015)

[21] "Apache Ofbiz®", http://ofbiz.apache.org/, (Retrived, 05th Jan, 2017)

[22] "Metrics 1.3.6-Getting Started", http://metrics.sourceforge.net/, (Retrived, 05th Jan, 2017)