A Review on Reuse of Software Components for Sustainable Solutions in Development Process

Dr. S.SRIDHAR
Professor & DEAN
RV Centre for Cognitive & Central Computing Facility
R.V.College of Engineering, Mysore Road, Bangalore-560 059

Abstract: **Effective reuse of a software product will increase the productivity, reliability and maintainability. It saves the development and verification time and reduces the risk and the cost involved in the software development. From the literature in this field, it is noticed that very few attempts had been made to identify or measure the software reuse process level. Also planning for reuse and determining the suitable component for reuse in a system development process have some significant challenges. To overcome these challenges reuse engineers must apply effective methods to identify high potential and quality reusable software components.**

Keywords : **Reuse of Software Components, Reliability, Productivity, Maintainability**

I. INTRODUCTION

The motivation for finding out the Reusable Software Components (RSC) of high potential and quality is got from business perspective. Since competitive products released earlier in market typically draw the most revenue, time-to-market is a critical factor for the commercial success of a product. We intend to design a conceptual reference framework which presents new ideas of verifying the functionality of the reusable software components before reusing in higher abstraction level. Another motivation is to improve the verification flow for higher assurance of quality reusable software components which can be reused in critical and very large scale system development. In addition, we also derive a set of quality measure for reusable software components which are collected in different level to ensure the quality of the components for high potential reuse.

II. SOFTWARE REUSE

Software reuse evolved from development of function calls in early programming languages and libraries of software routines for performing scientific calculations. In modern-day approaches of reuse covers the entire software life cycle and all software artifacts. Software reuse relies on preplanning to reuse a software component that meets the needs of the organizations involved in software development in new context. Mili et al (2002) defined the software reuse as “Software reuse is the process whereby an organization defines a set of systematic operating procedures to specify, produce, classify, retrieve, and adapt software artifacts for the purpose of using them in its development activities.” Margono and Rhoades (1992) pointed out some of the common reuse problems. Heineman and Councill (2001) proposed the following definition: A software component is a software element that conforms to a component model and can be independently deployed and composed without modification according to a composition standard.

III. CHALLENGES

A summary of the challenges,that we should be following Ivica and Magnus (2002) are component abstraction, functional properties, cost, maintenance needed, configuration management, retrievable and re-implementation. Jiang Guo and Luqi (2000) claimed that reusable software component should have the following functionality: - An automated repositories with a graphical user interface and an effective classification scheme for each domain. Several methodologies have been developed in order to verify the functionality of the components. They are simulation based verification, assertion based verification, formal verification, and coverage based verification (Wile et al , 2005). Min-An Song et al (2006) proposed a functional verification environment for advanced switching architecture. Intra-chip designs, in contrast, the speed of cross-chip connection, or even cross-system connection enhanced at the same rate and gradually became the bottleneck of current computing systems and PCI-Express epitomizes the evolvement of peripheral connecting strategy. Hinton et al (2006) proposed PRISM: A tool for automatic verification of probabilistic systems. The system requirements are stated in quantitative and probabilistic terms, such as reliability and performance requirements.

IV. COMPONENT-BASED DESIGN

Yunja Choi and Christian Bunse (2008) presented component-based design and verification of a μ-controller, a practical engineering method for reducing the complexity of modeling as well as the complexity of verification. Swarna Jyothi et al

(2008) presented reusable verification environment for verification of ethernet packet in ethernet IP core, a verification strategy- an analysis. Hasan and Tahar (2009) presented performance analysis and functional verification of the stop-and-wait protocol in HOL:- A formal verification technologies to support the performance analysis without considering the functionality correctness verification and based on probabilistic model in use. Pham Ngoc et al (2010) presented assume-guarantee tools for component-based software verification. In the proposed method, a verification target is decomposed into components, so that model can check each of them separately. The key idea of the method is to find out the minimal assumptions in the search spaces of the candidate assumptions. Chun-Hsian Huang and Pao-Ann Hsiung (2011) proposed Model-Based Verification and Estimation Framework for Dynamically Partially Reconfigurable Systems. Unified Modeling Language (UML) has been used to analyze dynamically partially reconfigurable systems (DPRS) that can reconfigure their hardware functionalities on-demand at runtime. Tasiran et al (2001) presented functional validation techniques through biased random simulation guided by observability based coverage.

V. ROLE OF NEURAL NETWORK

Shen and Fu (2005) used Neural Network Neural Network for used for Priority Directed Test Generation. The test bench is generated with help of two main modules: the Priority Control module and Neural Learning module. The Neural Learning module analyzes the coverage report and feeds the learning experience to the Priority Control module. The priority control module controls the random test generation by specifying a set of rules and attributes that are set with human interaction. Bayesian Network is used by Fine and Ziv (2003) is to define the relationship between the test directives of a random test generator and coverage report. A learning algorithm trains the data of the Bayesian Network with correct knowledge to direct the random test generator. In contrast, our algorithm starts with totally random data, where the quality of initial values affects only the speed of learning and not the quality of the encoded knowledge. Genetic algorithms are also introduced in the area of coverage driven functional verification as an optimization technique to generate input test directives and coverage report is collected for analysis. For example, genetic algorithm is proposed by Faye et al (2000) where the input test vector is a series of *n* numbers that are optimized and generated by a uniform random number generator.

VI. ROLE OF GENETIC ALGORITHM

James et al (2011) used a genetic algorithm genetic algorithm to improve the assertions coverage points for RTL models. This work tackles only a single assertion at a time, and hence is unable to achieve high coverage rates due to the simple encoding scheme used during the verification process. Several attempts have been made in the area of Reusability Assessment Models. Parvinder et al (2009) proposed reusability evaluation model for assessing the reusability of software components. The authors proposed the Neuro-fuzzy Inference engine can be used to evaluate the reusability. They refined CK metric suit and used it in their model. Parvinder et al (2007) proposed Quantitative Investigation of impact of the factors contribution towards measuring the reusability of software components which help to evaluate the quality of the components. They used Taguchi approach in analyzing the significance of different attributes in deciding the reusability level of a particular component.

VII. COUPLING AND COHESION

Gui and Paul D. Scott (2008) proposed new measure of coupling and cohesion to assess the reusability of components. They had shown that the new measures proposed by them were consistently superior at the time of measuring the component reusability. They used five metrics for coupling and five metrics for cohesion and they were very good predictors for evaluating the component for reusability. Parvinder et al (2009) proposed reusability evaluation system for object oriented software components. Sonia Manhas et al (2010) proposed reusability evaluation model for assessing reusability of software components. The different neural network approaches are used for the modeling of the reusability data. Fazal-e-Amin et al (2011) proposed reusability attribute model for assessing reusability of software components. The proposed model is derived using the GQM approach. Ajay Kumar (2012) proposed a model for classification of the reusability of software components using support vector machine.

VIII. CONCLUSIONS

Thus Reuse of software components is one of the promising aspects in software development process. Effective reuse of a software product will increase the productivity, reliability and maintainability. It saves the development and verification time and reduces the risk and the cost involved in the software development. From the literature in this field, it is noticed that very few attempts had been made to identify or measure the software reuse process level. Also planning for reuse and determining the suitable component for reuse in a system development process have some significant challenges. To overcome these challenges reuse engineers must apply effective methods to identify high potential and quality reusable software components. This review work will be very useful for researchers who are working in this direction.

IX. REFERENCES

[1]. Ajay Kumar (2012), "Measuring Software Reusability using SVM based Classifier Approach", International Journal of Information Technology and Knowledge Management, Vol. 5, No. 1, pp.205-209.

[2]. Chun-Hsian Huang and Pao-Ann Hsiung (2011), "Model-Based Verification and Estimation Framework for Dynamically Partially Reconfigurable Systems", IEEE Transactions on Industrial informatics, Vol. 7, No. 2, pp.287-301

[3]. Faye P., Cerny E. and Pownall P. (2000), "Improved Design Verification by Random Simulation Guided by Genetic Algorithm", Proceedings of ICDA, World Computer Congress, China, pp. 456-466.

[4]. Fazal-e-Amin, Ahmad Kamil Mahmood and Alan Oxley (2011), "Reusability Assessment of Open Source Components for Software Product Lines", International Journal on New Computer Architectures and Their Applications (IJNCAA), Vol. 1, No 3, pp. 519-533.

[5]. Fine S. and Ziv A. (2003), "Coverage Directed Test Generation for Functional Verification using Bayesian Networks", In Proceedings of Design Automation Conference, pp. 286-291, USA, ACM Press, Print ISBN: 1-58113-688-9.

[6]. Gui and Paul D. Scott (2008), "New coupling and cohesion Metrics for Evaluation of Software Component Reusability", In Proceedings of ICYCS, 2008, pp.1181-1186, Print ISBN: 978-0-7695-3398-8.

[7]. Hasan O. and Tahar S. (2009), "Performance Analysis and Functional Verification of the Stop-and-Wait Protocol in HOL". Journal of Automated Reasoning, Vol. 42, No.1, Springer, pp. 1-33, ISSN 0168-7433.

[8]. Heineman G.T. and Councill W.T. (2001), "Component-based Software Engineering Putting the Pieces Together", Addison-Wesley, ISBN: 0-201-70485-4.

[9]. Hinton A., Kwiatkowska M., Norman G. and Parker. D (2006), "PRISM: A tool for automatic verification of probabilistic systems", In Proceedings of International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS06), Vol. 3920, Lecture Notes in Computer Science, Springer, pp. 441-444.

[10]. Ivica Crnkovic and Magnus Larsson (2002), "Challenges of component based development", Journal of Software Systems, Vol. 61, No.3, pp. 201-212.

[11]. James H. Andrews, Tim Menzies and Felix C. H. Li (2011), "Genetic Algorithms for Randomized Unit Testing", IEEE Transaction on Software Engineering, Vol. 37, No.1, pp. 80-94.

[12]. Jiang Guo and Luqi (2000), "A survey of software reuse repositories", In Proceedings of the Seventh IEEE Conference and Workshop on the Engineering of Computer-Based Systems, pp. 92-100.

[13]. Margano J. and Rhoades T. (1992), "Software Reuse Economics: Cost-Benefit Analysis on a Large Scale Ada Project", In Proceedings of International Conference on Software Engineering, Australia, pp. 338-348.

[14]. Mili H., Mili A. and Yacoub S. (2002), "Reused Based Software Engineering: Techniques, Organization and measurement", Wiley.

[15]. Min-An Song, Ting-Chun Huang and Syen Kuo (2006), "A Functional Verification Environment for Advanced Switching Architecture", In Proceedings of the Third IEEE International Workshop on Electronic Design, Test and Applications (DELTA'06), Malaysia, pp.201-204.

[16]. Norman E. Fenton and Martin Neil (1998), "Software Metrics: Successes, Failures and New Directions", Journal of Systems and Software, pp. 1-13.

[17]. Parvinder S. Sandhu, Pavel Blecharz and Hardeep Singh (2007), "A Taguchi Approach to Investigate Impact of Factors for Reusability of Software Components", World Academy of Science, Engineering and Technology, pp.135-140.

[18]. Parvinder S. Sandhu, Harpreet Kaur and Amanpreet Singh (2009), "Modeling of Reusability of Object Oriented Software System", World Academy of Science, Engineering and Technology. No.30, pp. 162-165.

[19]. Pham Ngoc Hung,Viet-Ha Nguyen, Toshiaki Aoki and Takuya Katayama (2010), "Assume-Guarantee Tools for Component-Based Software Verification", In proceedings of Second International Conference on Knowledge and Systems Engineering, 2010.

[20]. Shen H. and Fu Y. (2005), "Priority Directed Test Generation for Functional Verification using Neural Networks", Proceedings of the Conference on Asia South Pacific - Design Automation, China, Vol. 2, pp. 1052-1055.

[21]. Singh Sandhu and Hardeep Singh (2006), "Automatic Reusability Appraisal of Software Components using Neuro-Fuzzy Approach", International Journal Of Information Technology, Vol. 3, No. 3, pp. 209-214.

[22]. Sonia Manhas, Rajeev Vashisht, Parvinder S. Sandhu and Nirvair Neeru (2010), "Reusability Evaluation Model for Procedure Based Software Systems", International Journal of Computer and Electrical Engineering, Vol.2, No.6, pp. 1107-1110.

[23]. Swarna Jyothi L., Harish R. and Manjunath A.S. (2008), "Reusable Verification Environment for verification of Ethernet packet in Ethernet IP core, a verification strategy- an analysis", in IJCSNS International Journal of Computer Science and Network Security, Vol .8 No.11, pp. 226-237.

[24]. Tasiran S., Fallah F., Chinnery D.G., Weber S.J. and Keutzer K (2001), "A Functional Validation Technique: Biased-Random Simulation Guided by Observability-Based Coverage", In Proceedings of International Conference on Computer Design, USA, IEEE Computer Society, pp. 82-88.

[25]. Wile B., Goss J. and Roesner W. (2005), "Comprehensive Functional Verification: The Complete Industry Cycle", ISBN-10: 0127518037.

[26]. Yunja Choi and Christian Bunse (2008), "Towards component-based design and verification of a μ-controller", In proceedings of 11th International Symposium on Component-Based Software Engineering, pp.196-211.