RESEARCH ARTICLE OPEN ACCESS

Software Reuse metrics and their impacts

Dr.Sanjay Kumar[1], Satish Kumar[2]
University Campus School, MDU Rohtak, India

ABSTRACT
Reusability is not only the process of using already existing artifacts to develop new software; it is the process of applying existing solution to the new arrived problem. Reusability not only reduces the rework, development time, cost and risk associated with the software product but also enhance the quality and productivity of the product. Software reuse concept is not rigorously adopted by software developer as it should be. The number of factors that work as hindrance in its widespread are reuse libraries, metrics, domain analysis and modelling etc. The concept of reusability can be applied in better way by breaking the product to be developed into its lowest units. This will increase the reusability in all phases and helps to overcome all the barriers. The availability of software reuse metrics plays a vital role in its implementation.
Keywords: *Software reuse, Software quality, Software productivity, Reuse ratio*

Date of Submission: 07-12-2017 Date of acceptance: 14-12-2017

I. INTRODUCTION

Software reuse is well known concept in software development. Every software developer apply it knowingly or unknowing. Reuse of previous knowledge, concepts and experience help the developer to work in batter and confidence manner.

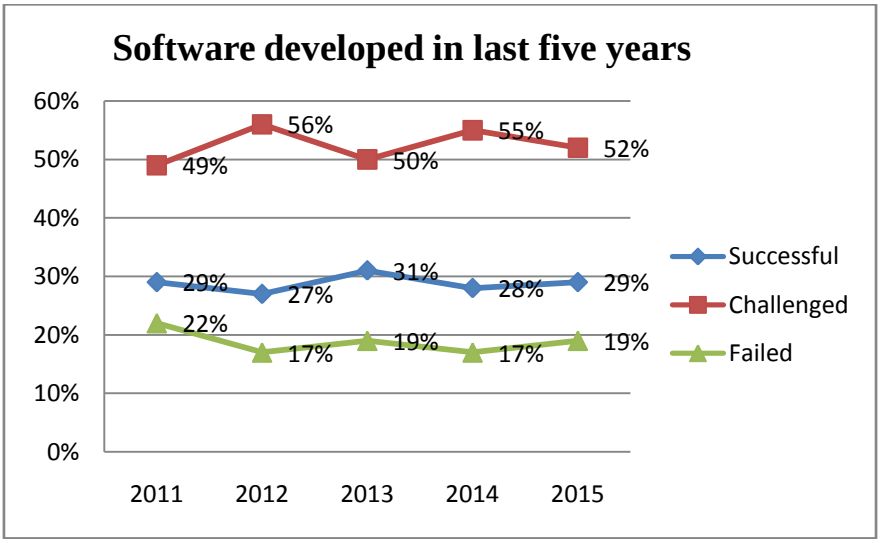


Table 1: *CHAOS Report* by the Standish Group

The survey conducted byStandish Group in 2015 and its reports as shown in fig. 1 states that more than half of the software developed in last five year was challenged. The percentage of successful software is even less than 30, which is great loss to software industry in both cases cost as well as time. The concept of software reusability not only reduces the failure rate, cost and time but also enhances the quality of the product.Frakes [1] conclude from his research and experiment that "using reusable software generally results in higher productivity". Benefits of software reusability is not only limited to the productivity of products but also certified that software reusability increases the quality of the product, because reusable components are more reliable and have less probability of errors as compared to the products or components developed from scratch. The reusable artifacts is the set of already existing components that can be reused, these can be requirement, design, coding components, data, test cases, testing techniques, implementation , acquired knowledge and experience etc. In present

time the reuse of components is more effective in practice, if the developer has practice of reusability [2]. Reusability of assets is different in different phases of software development, in design phase design components are reused, in coding phase coding of already existing components is reused as per their functionality, in testing phase existing test cases are reused and so on [3].
Reusability = Usability + Usefulness

2.1 Software Reuse metrics and their impacts

Quality metrics or other metrics play an important role in quality assurance of any software product. These metrics help the developer to assure whether the available components should be reused or not. They help to decide the reusability of any component is sensible to accept or not. They also help to decide whether the cost of reuse component is lower than the cost of component developed from scratch or not. Literature studies and empirical studies from industries and other institutions with aiming the relation between software reuse and quality attributes regarding usefulness, error density, effort and cost are observed [5, 8, 9, 10].

Aspect	Measurable Impacts
Quality	▪ Error Density ▪ Fault Density ▪ Ratio of major errors to total faults ▪ Rework Efforts ▪ Modules deltas ▪ Developers perception
Productivity	Lines of Code per effort

Table 2: Measurable impacts of software reuse [11].

The table 2 shows the metrics that are commonly used to measure the impact of software reuse on the developed software product. It is observed that reuse of components increases the quality by reducing the error or fault density, reduces the efforts of development. It also increases the productivity of the software product but reduces the development time [11].

The software reuse metrics available so far measure the amount of reused artifacts after the completion of the project or software product but the reuse of coding should not be given equal weightage to the reuse in requirement and design phase. We require such metrics that can measure the amount of reused artifacts in each phase of development. This process helps the developer to identify the percentage of reuse in different phases of development more accurately.

2.2 Proposed Reused metrics

A large number of studies certify that the use of reused artifacts can be done in almost every phase of software development. Starting from requirement phase to the implementation phase, the reuse of already existing components take place. It is observed that, as in phase of software development the developer has to deal with different methods or techniques of software development. So, if the measure of reused components is done at each level or phase, then that provide the better results as compared to the overall measurement of the product. It is because the requirement of requirement phase is different from that of coding phase and that of coding phase is different from testing phase. The consideration of design phase components with the components of coding phase or testing phase in calculating the reuse ratio of the product is not an appropriate method of measurement. So, in this research study it is suggested that reuse percent should be calculated phase wise, for each phase of software development product. It has two major benefits firstly, the developer will be able to know how much reused artifacts is used in that particular phase and secondly this will increase the accuracy in the comparison of reuse components to that of components developed from scratch. This also helps the software developer in the next projects where they are willing to increase the reusability in any phase of software development. The following metrics are proposed to calculate the percentage of software reuse in software development.

To calculate the percentage of reuse components in requirement phase separately.

Reuse Percent in Requirement phase

$$\mathbf{RRP\%} = \frac{\textit{Number of requirement reused}}{\textit{Total number of requirement}} \boldsymbol{X}\ \mathbf{100}$$

The above metrics helps to measure the percentage of reused requirement to the total number of requirements. As the requirement reuse ratio increases that ultimately increases the reusability in the next phases of software development because similar requirement have similar design, coding and testing process.

Reuse Percent in Design Phase

$$\text{RDP\%} = \frac{\textit{Number of modules design reused}}{\textit{Total number of modules designed}} X\ 100$$

Increase of reusability increases the percentage of reusability in coding and testing phases. This also helps the developer to increase the pace of development.

Reuse Percent in Coding Phase

$$\text{RCP\%} = \frac{\textit{Number of modules coding reused}}{\textit{Total number of modules coded}} X\ 100$$

Reusability is generally highest in this phase of software development. In this phase software developer tries to find the existing components with similar coding and functionality. If the component coding similar to 80% or above then that components is taken and modified as per the requirement of product and used in the product development. If the developer does not find any component, then that component is developed from scratch.

Reuse Percent in Testing Phase

$$\text{RTP\%} = \frac{\textit{Number of testing cases reuse}}{\textit{Total number of testing cases}} X\ 100$$

In testing phase test cases are reused. Similar components are tested in similar environment, so this type of situation increases the reuse of testing cases and metrics.

Overall reuse percent or average reuse percent of developed software can be identified or calculated as following

$$\textbf{Reuse Percent} = \frac{\textit{Sum of Reuse Percent in each phase}}{\textit{Number of phases}}$$

For example:

$$\textbf{Reuse Percent} = \frac{\textit{RRP}\%+\textit{RDP}\%+\textit{RCP}\%+\textit{RTP}\%}{4}$$

These metrics measure the amount of reuse components i.e. the requirement, design, test case or the coding more accurately as compared to the already existing metrics.

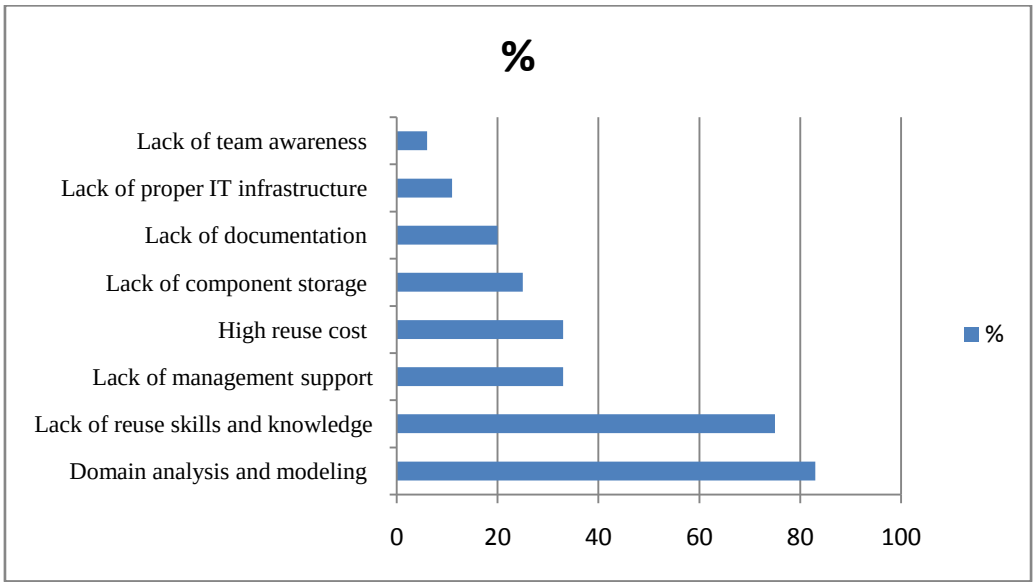


Table 3: Challenges in the software reuse [12]

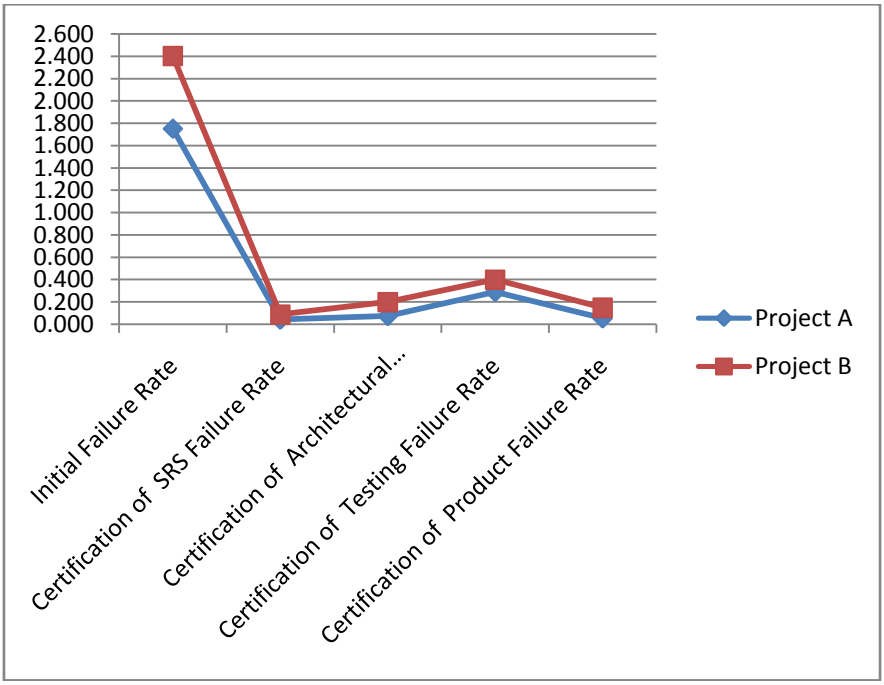
The considerable hindrances in the wide application of reusability are interrelated to each other. These are lack of component storage, documentation, high reuse cost and lack of management support. These challenges or hindrances can be sort out by the wide use of software reuse libraries. Software industry requires well developed and furnished software reuse libraries. Other challenges lack of reuse skills and knowledge, Domain analysis and modelingwill be sort out as the developer get best available components for reuse. This process reduces the effort and increase the interest of developer.

Table 4: Productivity, quality and other benefits of reuse based on HP's two program [5].

Division	A	B
Productivity	51% defect reduction	24% defect reduction
Quality	57 % increases	40% increase
Time to market	Data not available	42% reduction
Relative cost to create reusable code	200%	120-480%
Relative cost to reuse	10-20%	10-63%

III. IMPACT OF SOFTWARE REUSE AND ITS MATRICES

From literature studies it is observed that reusability depends upon many factors but the major factors are usefulness, cost and quality of the existing components. Eunjung Lee [4] states that reusing the already existing components (as shown in table 1) in software development require less effort and reduces the development time up to 42% as compared to development of components from scratch. The improvement and results are further summarized in tabular form in table 1.



The fig. 1: shows the comparison of two similar projects and their certification on basis of these metrics.

Project A is developed for University Campus School by using "Software Quality and productivity enhancement model" whereas Project B is developed earlier of some other school.The comparison in figure 1 shows that the product A has lesser failure rate as compared to product B that results quality of product A is higher than that of product B.

This comparison helps to certify that reuse of artifact

1. Reduces the rework (Past knowledge and experience)
2. Reduces the fault ratio
3. Reduces the product failure rate
4. Increases the reliability
5. Reduce the risks associated with the product development
6. Reduces the development time (Saves time in every phase of productdevelopment).
7. Reduces the cost of development

These characteristics help to state that reusability of artefacts increases the qualityand productivity of software product.

Although, the reuse of components do the cost/ benefits analysis for the software developers but it is also important to consider long term benefits of software reuse and the cost associated with them. So many examples are there that shows the importance of reusability such as:

There are many examples where reusability helps

1. By increasing its reusability by just 1% the Defense Department of U.S. can annually save up to $300 million [6].

2. The Missile Systems Division (MSD) uses the computer software reuse concept to increases its productivity by 50%.
3. American Navy uses the reusable assets that reduces its labor cost to develop and maintain the Restructured Naval Tactical Data Systems (RNTDS) by 26% [7]

IV. CONCLUSION

Although, the reuse of components do the cost/ benefits analysis for the software developers but it is also important to consider long term benefits of software reuse and the cost associated with them. Software reusabilitynot only reduces the rework, fault ratio, product failure rate, risks associated with the product development,development time, costof developmentbut also Increases the reliability and the quality of the product.

REFERENCES

[1]. W.B. Frakes and C. J. Fox, *Sixteen Questions about Software Reuse*, Communications of the ACM, Edition 6, Vol. 38, pp. 75-87, June 1995.
[2]. L. Etzkorn, B. Jagdish and D. Carl, *Design and Code Complexity Metrics for Object-Oriented Classes.* Quality Metric for Object-Oriented Design. Journal of Object-Oriented Programming. April 1999.
[3]. J. S. Alghamdi , *Measuring Software Coupling*, Information & Computer Science Department King Fahd University of Petroleum & Minerals Dhahran, 31261, Saudi Arabia.
[4]. E. Lee, *Software reuse and its impact on Productivity, Quality and Time-to-market*.
[5]. W. C. Lim, *Effect of reuse on Quality*, Productivity and Economics Workshop on Metrics. IEEE 1994.
[6]. N. Paliwal, V. Shrivastava and K. Tiwari, *An Approach to Find Reusability of Software Using Objet Oriented Metrics*, International Journal of Innovative Research in Science, Engineering and Technology ISSN: 2319-8753 Vol. 3, Issue 3, March 2014
[7]. R. Patidar and V. Singh, *An Approach to Calculate Reusability in Source Code Using Metrics*, Int. Journal of Engineering Research and Applications www.ijera.com ISSN : 2248-9622, Vol. 5, Issue 2, pp.34-39, February 2015.
[8]. E. Henry and B. Faller, *Large-Scale Industrial Reuse to Reduce Cost and Cycle Time*. In: IEEE Software, Vol. 12, No. 05, September 1995, pp. 47-53.
[9]. P. T. Devanbu, S. Karstu, W.L. Melo and W. Thomas, *Analytical and Empirical Evaluation of Software Reuse Metrics*, In: Proceedings of the 18th International Conference on Software Engineering (ICSE), Berlin, Germany, pp. 189-199, 1996.
[10]. V. R. Basili, L. C. Briand and W. L. Melo, *How Reuse Influences Productivity in Object-Oriented Systems*, In: Communications of the ACM, Vol. 39, No. 10, October, 1996, pp. 104-116
[11]. P. K. Suri and N. Garg, *Software Reuse Metrics: Measuring Component Independence and its applicability in Software Reuse*, International Journal of Computer Science and Network Security, VOL.9 No.5, May 2009
[12]. SajjadMahmood, Ali Al Zayer, *Challenges of Adopting Software Reuse: Initial Results*. ICSEA 2014 : The Ninth International Conference on Software Engineering Advances
[13]. *CHAOS Report* by the Standish Grouphttps://www.infoq.com/articles/standish chaos-2015.
[14]. Sanjay Kumar, Rahul Rishi and RajkumarYadav,*Metrics to develop high quality software*, Indian Journal of Science andTechnology, Vol 9(48), DOI: 10.17485/ijst/2016/v9i48/105607, December 2016

International Journal of Engineering Research and Applications (IJERA) is **UGC approved** Journal with Sl. No. 4525, Journal no. 47088. Indexed in Cross Ref, Index Copernicus (ICV 80.82), NASA, Ads, Researcher Id Thomson Reuters, DOAJ.

Dr.Sanjay Kumar "Software Reuse metrics and their impacts." International Journal of Engineering Research and Applications (IJERA) , vol. 7, no. 12, 2017, pp. 01-05.