Jurnal
Teknologi

Full Paper

METRIC BASED SOFTWARE PROJECT PERFORMANCE MONITORING MODEL

Mariayee Doraisamy*, Suhaimi Ibrahim, Mohd Naz'ri Mahrin

Advanced Informatics School, Universiti Teknologi Malaysia, Kuala Lumpur, Malaysia.

Article history
Received
26 November 2015
Received in revised form
14 January 2016
Accepted
10 October 2016

*Corresponding author
maarisa75@gmail.com

Abstract

Successful implementation of software projects development is entirely depending upon successful monitoring and control mechanism. Software metrics can deliver the necessary information for monitoring and control the software projects development for its enhancement. However, the current software metrics does not widely address the performance criteria and related metrics for software project management. Largely, metrics are identified in the perspectives of software development only. Hence, the aim of this study is to formulate a Metric based Software Project Performance Monitoring Model which consists of performance criteria and metrics that involves in a software projects development. This model formulation is consists of five processes: metrics integration, metrics validation, metrics description, metrics categorization and metrics threshold. The proposed model is a novel approach and adds significant of knowledge to the software engineering domain especially on software project monitoring and software measurement domain. Generally, this model will be a guideline for software project managers to monitor and control software projects particularly in public sector software projects. In order to demonstrate the applicability of this model, case study was conducted at various departments at Malaysian Public Sector. The results show that the proposed model is very useful for the project managers in monitoring and control software projects.

Keywords: Software projects development, monitoring and control, performance criteria, metrics, metrics validation, threshold,

1.0 INTRODUCTION

Software projects have a high rate of failure. In fact, organizations have tried to reduce the rate through many ways [1]. There is still software projects are delay in delivery, overrun cost, insufficient quality, do not meet user requirements and less customer satisfaction [2]. Wateridge [5], in his research on successful and failure projects had summarized that criteria such as meet user requirements, completed on time, carried out within budget and meet the quality requirements are the major criteria need that need to be consider for measuring the software projects. Additionally, software projects need to be monitor frequently in order to success.

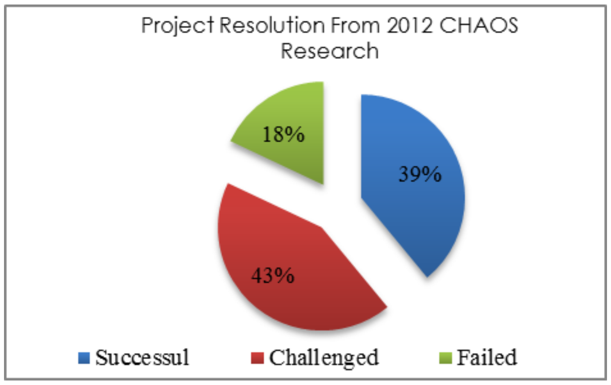


Figure 1 Project Resolution from 2012 CHAOS Research

The 2012 CHAOS [3] indicates project success rates, with 39% of all projects are successful (delivered on time, on budget, with required features and functions) where as 43% of projects were challenged (late, over budget, and/or with less than the required features and functions) and 18% of software projects are failed (cancelled prior to completion or delivered and never used) as shown in the Figure 1. Although many attempts [2][4][6] have been made to solve the problem in the last few decades, but the amount of challenged and failed projects are still higher than the amount of succeeded projects. There are only 39% of succeed projects in the year of 2012. Many actions are being taken by the practitioners and scholars in order to reduce the amount of challenged and failed software projects development. However the result is not inspiring.

The same scenario goes to public sector software projects development. For example, Malaysian Administrative Modernization and Management Planning Unit (MAMPU) was conducted a survey in 2010 on software projects development at Malaysian public sector. This study indicated that many outsource projects listed as challenged and failure projects [10]. In fact, the developments of some software projects are failed in the beginning stage itself. For example, the Health Ministry of Malaysia has ended its contract with one of the software company. This software company failed to develop two proposed software projects namely Pharmacy Enforcement Management System (SPPF) and Pharmacy Management System (SPF). Almost RM2.59 million in expenses was not considered value for money to the government [11].

Software projects development need to be monitor frequently in order to have successful software project. In the context of Malaysian Public Sector, a survey was conducted in the year 2013 among government ICT officers on software projects monitoring. Almost 65.3% respondents agreed that there is a lack of having effective monitoring and control of software projects at Malaysian Public Sector [12]. This result shows that there is a need to have effective monitoring mechanism in order to reduce the software projects failures at Malaysian Public Sector.

Well established monitoring methods such as Earned Value (EV) is added value on monitoring software projects by looking at project duration and cost [24]. In addition to this, there are many models for monitoring processes of software projects were introduced such as system dynamics model [7], scenario model [8], PERT method, Use Case Point [9], Model-Driven, Bayesian Based and Shared Mental [25]. Eventually many studies are being conducted in the field of software projects monitoring yet there is always a room to explore to enhance the existing studies on determining a software projects success.

In line with this, metrics are vital to determining the software projects success. Generally, software metrics can deliver the necessary information for managerial understanding in managing and control the software projects development for its enhancement [18]. However, the current software metrics does not widely address the performance criteria and related metrics for software project management. Largely, most of the metrics are identified in the perspectives of software development only. The existing software project monitoring literatures are merely focused on monitoring the cost and schedule elements. We believe that software projects success can be achieved by using the performance criteria and metrics which influence software projects development. Accordingly, we can make the software projects development moves towards success. Thus, a development of Metric based Software Project Performance Monitoring Model could guide the software project managers to monitor and control the performance of software projects towards success. By using this proposed model, project managers could monitor and manage the performance for each element that involved in the software projects development. Besides this, software project manager also could view the performance of the each and every element in a software projects development. Consequently, this proposed model encompasses 14 identified performance criteria and 143 related metrics that can be monitored during the development of any software projects specifically outsource projects. This paper delivers insights the formulation of Metric based Software Project Performance Monitoring Model by systematically.

This paper begins with the discussion by providing step by step instructions on Formulation of Metric based Software Project Performance Monitoring in the Section 2.0. Section 3.0 explains the results and discussion. This paper ends with overall conclusion of these activities by summarizing the entire formulation process and describing the model evaluation and the results in the Section 4.0.

2.0 METHODOLOGY

The model formulation is divided into several important activities which are metrics integration, metrics description, metrics validation, metrics categorization and metrics performance threshold value. These activities produce a finalized metrics which will be significance for monitoring and control software projects.

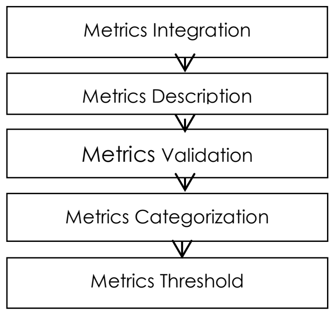
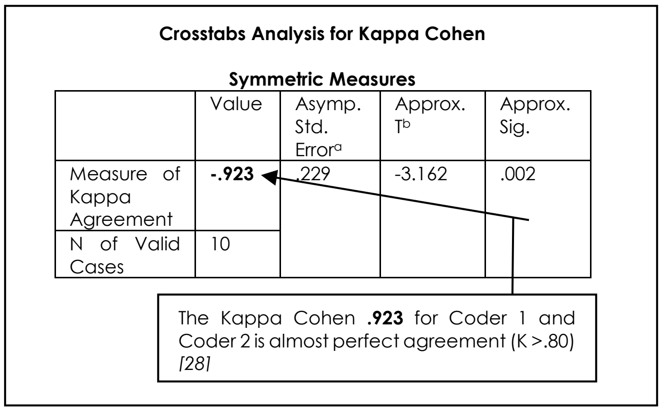


Figure 2 Activities involved in formulation of Software Project Performance Model

Figure 2 shows the activities in the formulation process. This research begins with Systematic Literature Review (SLR) [13]. A list of performance criteria and metrics that influence the software project monitoring were identified from this SLR. This SLR study identified a number of 14 performance criteria and 110 related metrics based on 43 selected studies. This followed by the second activity which is identifying the performance criteria and metrics that influence the software project monitoring from the industrial perspectives. Here, we had conducted structured interviews with software project managers from various departments at Malaysian Public Sector.

The data collection phase was conducted in two different phases. In order to triangulate the first phase data, we conducted the second phase of structured interview sessions at various departments in Malaysian Public Sector at the different time and different places. A total of 37 software project managers were involved in these phases. We used purposive sampling for data collection. Additionally, we have identified software project managers from all the ministries at Malaysian Public Sector. Thus, at least an experienced software project manager from one ministry was involved in these data collection phases. This is to ensure that collected data are from various types and environment of software projects development at Malaysian Public Sector. Data were collected, transcribed and analysed using NVIVO 10. In this phase, we had collected 13 performance criteria and 87 related metrics. Consequently, we tested the reliability of our transcribed data by performing peers review data transcription [30]. Subsequently, the reliability (inter-coder reliability or inter-rater reliability) was determined using Kappa Cohen in this study. Two researchers (Coder 1 and Coder 2) were chosen to identify number of codes by reading the transcribed data randomly. These two researchers identified a number of codes from the transcription documents based on the given coding scheme. The Kappa Cohen statistics shows high reliability of data transcriptions as described in Figure 3. The Kappa Cohen **0.923** for Coder 1 and Coder 2 is almost perfect agreement. This value ensured the accuracy of transcribed process in this study.



Crosstabs Analysis for Kappa Cohen

Symmetric Measures

	Value	Asymp. Std. Error^a	Approx. T^b	Approx. Sig.
Measure of Kappa Agreement	**-.923**	.229	-3.162	.002
N of Valid Cases	10			

The Kappa Cohen **.923** for Coder 1 and Coder 2 is almost perfect agreement (K >.80) *[28]*

Figure 3 Crosstabs analysis for kappa cohen

Consequently, the triangulation results show that identified data having very higher similarities and enhance confidence in the ensuing the findings. We further our discussion with model formulation activities.

2.1 Metrics Integration

Basically, in this phase metrics from SLR and structured interviews were gathered. These gathered metrics were mapped and integrated in deriving the final metrics using Constant Comparative technique which is core to the Grounded Theory method.

Constant Comparative is a process of constantly comparing occurrence of data that labelled in a category with other same category to see they are fit and workable or not [14]. The amount of data that collected is compared and examine by explicitly and implicitly using this constant comparison technique. Thus, each and every metrics that identified in this study went through comparison and analysis based on the Constant Comparative process.

In line with this, each metric are analyzed implicitly and explicitly by looking at the terms, meanings, logics and structures in detail [15]. Furthermore, the similarities and differences of each metric are also analyzed. These Comparisons highly considered for increasing the internal validity of the findings. Based on our study, we derived a five-step analysis procedure. Our study comprises five steps which are:

a. Compare the metric by its phrases, meaning, logics and sentences structures by individually. (Internal validity)
b. Compare the similarity between two metrics that identified based on the SLR and structured interviews. (External validity)
c. In some cases, metrics are not available in either SLR or structured interviews. Thus, we analysed the practicality of that metric for the monitoring the software project. (External validity)
d. Create a category of metric.
e. List down final metrics.

A number of 14 performance criteria and 141 metrics are derived from this integration process. These performance criteria and metrics are basically used for monitoring software project in the industry. Next, these identified metrics were validated by using experts.

2.2 Metrics Validation

These identified metrics were validated by experts. Experts from project management field are invited to involve in this validation process. Expert judgement enables to acquire opinions from the real people in the industrial [16]. Metrics validation process involved three important issues which are metrics categorization, metrics significant and metrics valid measurement. These three important issues were derived based on the software metrics validation methodologies in software engineering that proposed by K.P Srinivasan [17].

Experts were given 2-3 weeks' time to review and validate these identified metrics. Experts validated the identified metrics by reviewing three important criteria such as:-

- Does the metrics titled correctly according to SLR and interviews metrics?
- Does the metrics are useful for monitoring and controlling software projects?
- Does the metrics having a valid measurement types?

We established the experts' criteria in order to execute this metric validation. These criteria were

Table 1 Detail of expert data analysis

Performance Criteria	**Metric Name**	**Excluded /Included**	**Reason for excluding / including**	**New Metrics**
Team members	Team member residency (Total Distance)	Excluded	This metric is not included in this proposed model because at public sector scenario team member travelling cost is not in counted as they are paid by monthly salary.	
Organization	Software project contributes to organization ICT Strategic Plan (Yes / No)	Included	Monitoring the ICT Strategic Plan is very important in public sector projects. Thus, under this circumstance, this metric is modified into new two metrics.	Existence of project in ICT Strategic Plan
				Number of projects implemented from the ICT Strategic Plan

established in order to have reliable metrics for monitoring the performance of software projects. Ultimately, these experts involved in software projects management. Below are the expert criteria:

- Software Project Managers or Software Project consultants at the Malaysian Public Sector.
- Appointed as Project Management Experts at the Malaysian Public Sector by Public Service Department, Malaysia.
- Experiences more than 10 years in the managing software projects at the Malaysian Public Sector.
- Certified project management consultants at the Malaysian Public Sector.

Software Project management experts from Public Sector of Malaysia were contacted via email and phone calls to get their commitments for validation purpose in this study. Initially, five experts were chosen based on the above determined experts criteria. Unfortunately, one of the identified experts could not able to take part in this validation process due to high commitments and workloads. The Metric Validation Form was distributed to the experts in the introduction session. We further our work by collecting Metric Validation Form from the experts. The experts took appropriately a month to complete this form. We collected the form by softcopy (email) and hardcopy (manual form). We continue our work by analyzing the experts' comments on these metrics. All the 141 metrics was reviewed by the experts. Experts commented on some of the metrics by of its name, measurement way and the structure. Out of these 141 metrics, one of metric is excluded from the proposed model based on the experts' reviews as described in the Table 1.

Three out four experts were not agreed for this metric which is *team member residency (Total Distance)*. Besides this, *Software project contributes to organization ICT Strategic Plan* metric was divided into two more new metrics such as *Existence of project in ICT Strategic Plan* and *Number of projects implemented from the ICT Strategic Plan*.

Altogether a total of 143 metrics were identified in this study. Table 2 shows the final identified metrics for Project Manager Performance Criteria. Next, the final identified metrics are described as explained in the next Section 2.3.

2.3 Metrics Description

Software metrics purpose is to provide a quantitative assessment of the elements or attributes. Software metrics will be meaningful if it includes a description of how data are to be presented, interpreted and used regards software projects [18]. Thus, in this study metrics were defined based on the ISO/IEC TR 9126:2003 standards. ISO/IEC TR 9126:2003 [19] is a standard provides a complete and comprehensive report on metrics that involved in software product and process [20]. Moreover, most of well -established software organizations are using this standard as a reference guide in their software development processes [20]. Below are the attributes that included as a metrics description.

- Metrics name
- Purpose of the metrics
- Method of application
- Measurement formula and data element computational
- Interpretation of measured value
- Metric scale type
- Measure type
- Input to measurement
- Target audience

The above attributes are used to formulate the metrics descriptions. Each identified metrics are scrutinized and detail it in a comprehensive way. As shown in Table 2, each identified metrics for project manager performance criteria were reported by explaining the metric purpose, metrics method application, measurement formula, interpretation of measurement value, scale, measure type, source for the measurement and target audience.

By detailing these metrics, we get a clear guide on how to use of these metrics in monitoring every software projects basically. Metrics description is very important for the project managers to understand better about metrics and what, when and how they should use these metrics.

All the described metrics were reviewed by the software projects experts. This metrics description was reviewed by the same software project experts who involved in the metrics validation. This is to make sure the continuity of the metrics identification and description. This ensures the reliability of identified metrics that identified in this study. Unfortunately, only two experts (Expert B and Expert C) managed to take part in this metrics description. The other two experts (Expert A and Expert D) were having a tight schedule with their workloads. Thus, these two experts are unable to take part in the review process. The review was conducted for two rounds.

The first round of this review was conducted for three hours in a discussion mode. The experts reviewed all the described metrics and recommended some changes on these below stated subjects such as:

- Target audiences
- Input measurement documents
- Measurement formula
- Metrics scales

The output of the first round review was considered important because of the expert's experiences and credibility in managing, monitoring and producing software projects for 20 years at Malaysian Public Sector. Then, the metrics descriptions were amended according to the first review and send for the second review with same experts for the validity purpose. The experts are agreed on these metrics description without any changes for the second review as shown in the Appendix A. Finally, the metrics were ready with complete descriptions. Figure 4 shows the review process in a graphical form.

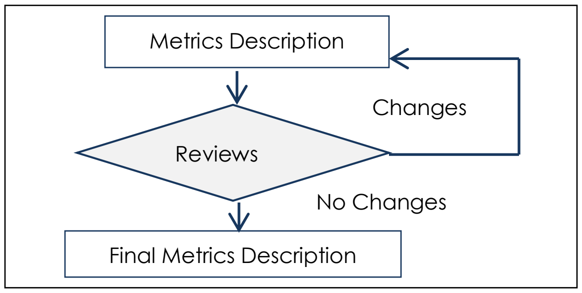


Figure 4 A review process of metrics description

We continue with the next activity which is metric categorization which explained o the next section.

2.4 Metrics Categorization

The formulation activities were continued with metrics categorization. The validated metrics were categorized according to Project Management Iron Triangle model. This model has three important elements which are Cost, schedule and Quality. Figure 5 shows how these performance criteria and the related metrics were categorized.

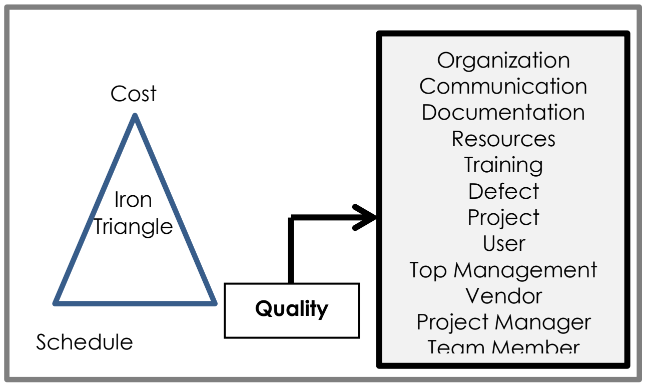


Figure 5 Metric categorization based on Iron triangle Model

The purpose of metrics categorization is to answer the research gap which claimed there are no commonly agreed practices of performance criteria and metrics for managing software projects in the perspectives of public sector. We believe that software project also can be monitored by not only looking at the cost and schedule but other elements under quality domain such as top management, vendor, project manager, team member, documentation, communication, training, defect, organization, user, resources, and defect. Thus, we described how these performance criteria give significance to the software projects towards success using case study evaluation in the

Table 2 Final metric lists for project manager performance criteria

Number	Metric Name
	PROJECT MANAGER
PM1	Type of skill or expertise
PM2	Number of skills or expertise
PM3	Number of meetings with users
PM4	Number of stakeholders meetings
PM5	Number of meetings with vendor
PM6	Total number of projects
PM7	Total number of successful software projects
PM8	Total number of high impact successful projects
PM9	Total number of unsuccessful software projects
PM10	Time taken to identify and solve the problem
PM11	Total time taken to complete each task
PM12	Total time spend for the projects in a day
PM13	Project manager appraisal (% of performance)
PM14	Total number of tasks
PM15	Number of completed tasks on time
PM16	Number of project plans per project

Section 3.0. We further our work with the threshold activity as in the next section.

2.5 Setting Metrics Threshold Measurement Scale (MTMS)

Setting Metrics Threshold Measurement Scales (MTMS) is the final process in this model formulation. In this phase, we established Metrics Performance Threshold Scale for our validated metrics. This measurement scale was established for the evaluation purpose. Threshold measurement guides the software project manager to count the performance of software projects by quantitatively.

This MTMS was developed based on the literature [21]. There are number of papers that discussed about the threshold setting for object oriented metrics [22] and security metrics [23]. Additionally, our study adopted this threshold rating scales from the existing security metrics study that proposed by Shareeful and his co-authors [21]. The proposed scales are more suitable and meaningful for the software project metrics threshold in the context of the public sector environment.

Finally, the Metric-based Performance Model for Software Project Monitoring and Control is consists of performance criteria, metrics with descriptions and metrics threshold measurement scale was formulated. This model is a novel approach for software project monitoring and control. Software project managers are the targeted users for this model. This model can be used by software project managers for monitoring and control the software projects. The Metric-based Performance Model for Software Project Monitoring and Control delivers sights on all the elements that included in this model development. Besides these elements, the conceptual model also shows the input and output of the model. This model can be used during the project management life cycle phases. Figure 6 illustrates the proposed Metric-based Software Project Performance Model.

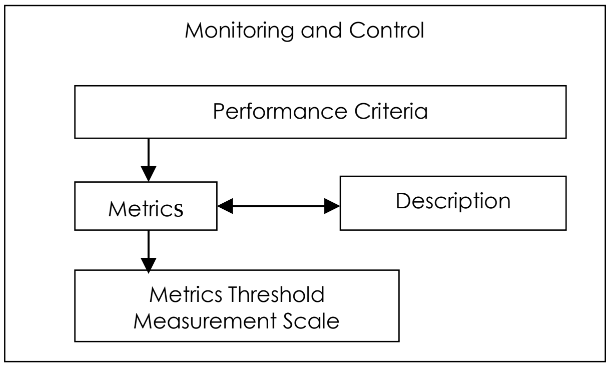


Figure 6 A conceptual model of metric-based software project performance model

3.0 RESULT & DISCUSSION

Case study methodology is used to evaluate the proposed model by empirically. Case study methodology is well suited for many kinds of software engineering research, as the objects of study are contemporary phenomena, which are tough to study in remoteness. Besides this, case study methodology was originally used primarily for exploratory and descriptive purposes. According Klein and Myers [29] define three types of case study depending on the research perspective which are positivist, critical and interpretive. This study is more to positivist case study which searches confirmation for our model evaluation, measures metrics, test hypothesis and draws inferences from selected software projects.

The objectives of this case study are:-

- To show the relationship between the number of metrics and the software project success.
- To show the relationship between the performance level of each criterion and the software project success.
- To show that the proposed model is useful for the software project managers in monitoring and control the software projects.

This case study is based on the seven stages to tail in as suggested by Kitchenham [30]. Six software projects were selected as unit of analysis in this case study evaluation. The evaluation was conducted in two different environments which are large scale and small scale. These software projects were gathered based on the criteria which are successful, problematic and failure software projects in large as well as small project environments. Emailed were sent to the head of department of two organizations at Malaysian Public Sector to gather software projects details. These two organizations which known as A and B were agreed to participate in this model evaluation. Organization A is for large scale environment and Organization B is for small scale environment. Both organizations provided us the detail of six software projects according to the given criteria. These identified software projects are illustrates as below in the Table 3.

Table 3 Unit of Analysis in our multiple case studies

Software Project Environment	**Successful (on time and on cost)**	**Problematic (delay in time and cost)**	**Failure (abounded or neglected)**
Large Scale	Software Project A	Software Project B	Software Project C
Small Scale	Software Project D	Software Project E	Software Project F

Hypotheses testing were conducted in order to conclude if there is a significant effect on identified metrics for monitoring the performance of software project. The derived null hypotheses were disapproved in this study. The generated null hypotheses in this case study were described as below:-

H0$_1$: The less the metrics used for monitoring then higher the software project success.

H0$_2$: The lower the performance level (%) of the each criterion then the higher is the software project success.

This case study evaluation started with determining threshold value for validated metrics and followed by model evaluation. The model evaluation was conducted by the software project managers of the selected software projects. Then, the hypotheses were tested.

3.1 Determining Threshold Value for Validated Metrics

In this section, threshold value for each and every identified metrics in this study was derived. A session was conducted in order to determine the threshold value for the metrics. The details of this session and the threshold are explained below.

3.1.1 Session Details

The researcher played as a moderator to reach the consensus for each and every metrics threshold in the session. Three software project managers were gathered for this session. Well experienced project managers participated in this session. These three software project managers are having more than 15 years experiences in managing software projects at Malaysian Pubic Sector. Furthermore, these software project managers are having experiences in both large and scale software projects development at as well. This is to ensure that proposed model can suits for large and small scales software projects environments. Software project managers from two IT Departments as identified as a Department A and Department B were chosen as session participants. The two departments details are described as below:-

a. Large Scale Software Projects Environment

Department A was identified for the large scale projects environment. Department A is basically handling many large scale projects for Malaysian Public Sector. Each software project managers in this department is monitoring and controlling many high impact software projects that involves at Malaysian Public Sector. These identified software project managers are also became software projects consultants for many other IT agencies at Malaysian Public Sector. Thus, two well experienced software project managers from this department were gathered to take part in a threshold activity.

b. Small Scale Software Project Environment

Department B was identified for small scale software projects environment. This department are basically handling a few software projects for their internal use of IT departments at the Malaysian Public Sector. Their software projects are not a high impact projects. These software projects are small scale projects. Only limited number of users is using these small scale software projects at the agency. Thus, a well experienced software project manager was invited from this department to take part in this session.

3.1.2 Sessions Output

As explained earlier, a group of three software project managers were involved in this session. The software project managers and the moderator were sitting together in this session and discussed about threshold for each metrics according to the software project environments. The results are promising; each participant has been able to set the threshold value for each metrics in the model. The proposed model seems to be a helpful model as it can be used for different software project environments in order to monitor and control the software projects. In addition, the experiences of the session participants in monitoring and controlling on software projects confirm that these agreed metrics threshold will be very useful for other software project managers to monitor and control the software projects basically.

The execution of the session was divided into introduction part and the discussion part. The purpose of the introduction is to give the participants the basic knowledge on performance criteria and the related metrics as well as how these metrics can be used for monitoring and controlling the software projects in real environment. Upon an execution of the introduction section, a discussion on the metrics threshold was furthered with software project managers in the session. The threshold was set based on their experiences and knowledge in handling many software projects at Malaysian Public Sector for more than 15 years.

This session was held for the duration of three hours in order to set the threshold for each identified metrics. Each and every metrics was reviewed and assigned the threshold value based on the projects environment. The input of this session was analyzed by the software project managers and moderator until we get the consensus for the threshold value for each metrics. Any arguments in determining the threshold were got back to the consensus finally by the moderator and the software project managers at the end the session. In order to increase our threshold consistency we strictly follow:-

- Describe each and every metrics in detail on how it is giving significant for monitoring software projects.
- We compare the threshold value for each types of software projects environment before the get consensus value for metrics.

Appendix B and Appendix C describe the threshold value for large scale and small scale software projects. The input from the session were collected and documented for the next phase of evaluation. Next, we move on model evaluation using multiple case studies at selected software projects in Malaysia Public Sector.

3.2 Model Evaluation

The proposed model was evaluated using with six types of real projects in a two different software projects environment as described above. This is to ensure that the model can be used for all types of project environments. Project manager of each identified software project was chosen in order to participate in this evaluation. The metrics list with threshold value was distributed to each project manager of the selected software project. They evaluated their software projects using these metrics based on their experiences on handling the selected software project. The objectives of this case study evaluation are explained in detail as below.

3.2.1 To Show The Relationship Between The Number Of Metrics And The Software Project Success.

As discussed earlier, one of the objectives of this case study is to show the relationship between the number of metrics and the software project success. Figure 6 illustrates the number of used metrics in the identified six software projects in both large and small scale environment in this case study. The finding shows that *the more the metrics used for monitoring the higher the software project success*. This is evidenced in Software Project A and Software Project D. These two software projects are successful projects. The project manager from these software projects used more metrics that listed in this model in their software project A (120 metrics) and D (123 metrics) as well. The failure software projects which were Software Project C (68 metrics) and Software Project F (64 metrics) used very less metrics that listed in this model.

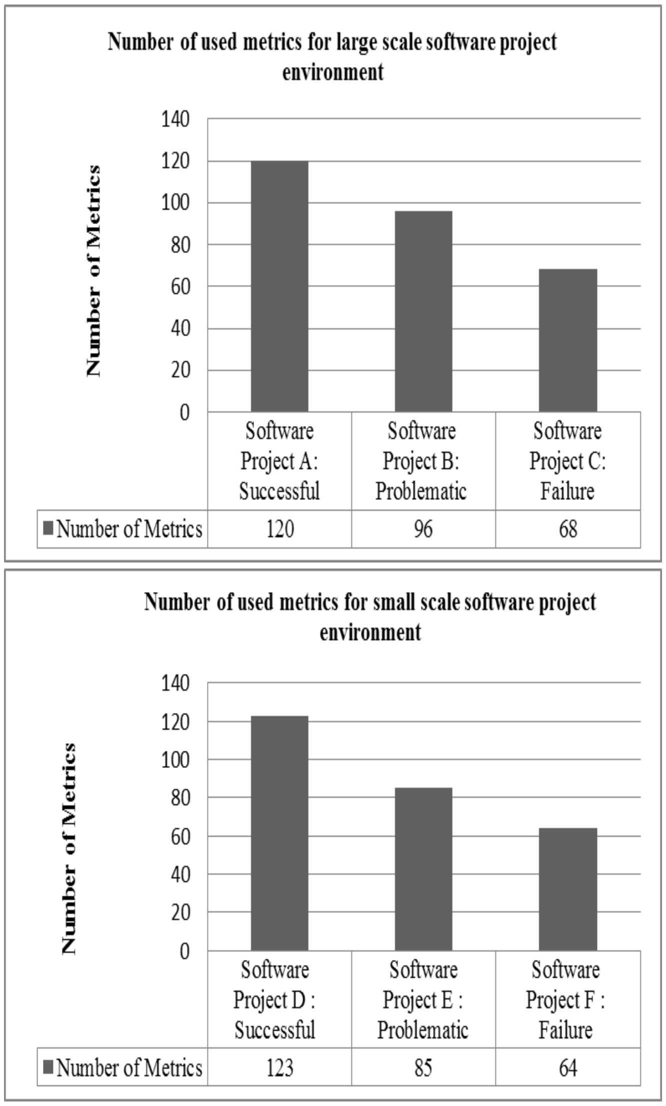


Figure 6 Number of metrics used for large and small scale software projects

Whereas the problematic software projects which were Software Project B (96 metrics) and Software Project E (85 metrics) used average metrics. Consequently, successful software projects were used more metrics compare to problematic software projects as well as the failure software projects.

In summary, there is a significant positive relationship between the number of metrics and the software project success. The more the number of metrics used from this proposed model then the higher the software project success. Thus, the null hypothesis **(H0$_1$)** "***The less the metrics used for monitoring the higher the software project success"*** was successfully rejected in this study.

3.2.2 To Show The Relationship Between The Performance Level Of Each Criterion And The Software Project Success.

As discussed earlier, one of the objectives of this case study is to show the relationship between the performance level of each criterion and the software project success. Table 4 illustrates the performance level of each performance criterion for large scale software project that involved in this case study.

Table 4 Performance level of each criterion for three large scale selected project
s

ID	Performance Criteria	Software Project A	Software Project B	Software Project C
PM	Project Manager	79.7	46.9	23.4
S	Schedule	100	25	0
C	Cost	100	55	40
TM	Team Members	73.6	38.8	16.7
U	User	75	57.1	32.1
R	Resources	100	50	50
D	Defect	100	100	0
TP	Top Management	58.3	50	8.3
P	Project	100	78.1	53.1
D	Documentation	79.5	61.4	0
C	Communication	90.9	68.2	18.1
V	Vendor	91.2	41.2	23.5
T	Training	78.6	39.3	28.6
O	Organization	100	83.3	66.7

The performance level of each criterion was grouped into three rating scales. These three ratings scales are:-

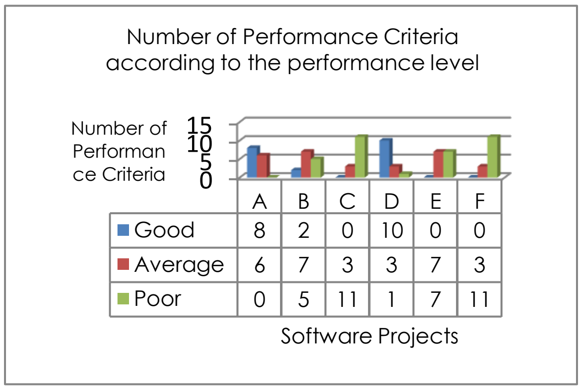
- Good Performance Level (80% and above)
- Average Performance Level (Between 50% to 79%)
- Poor Performance Level (Below 50%)

Consequently, Software Project A has eight (8) criteria above 80% performance level which are Project Manager, Schedule, Cost, Resources, Defect, Project, Communication, Vendor and Organization. Table 5 illustrates the performance level of each performance criterion for small scale software project that involved in this case study.

Table 5 Performance level of each criterion for three small scale selected projects

ID	Performance Criteria	Software Project D	Software Project E	Software Project F
PM	Project Manager	85.9	25	9.4
S	Schedule	97.7	15.9	0
C	Cost	100	60	30
TM	Team Members	65.3	13.8	6.9
U	User	85.7	21.4	7.1
R	Resources	100	50	50
D	Defect	100	50	0
TP	Top Management	44.4	19.4	2.8
P	Project	100	62.5	50
D	Documentation	88.6	68.2	0
C	Communication	72.7	47.7	4.5
V	Vendor	77.9	29.4	17.6
T	Training	82.1	50	14.3
O	Organization	100	66.7	66.7

Performance criteria such as Training, Documentation, Top Management, User and Team Members are having average performance level. This is similar to another successful software project which is Software Project D. There are 10 criteria above 80% performance level which are Project Manager, Schedule, Cost, Resources, User, Defect, Project, Documentation, Training and Organization. Performance criteria such as, Communication, Vendor and Team Members are having average performance level. A Top Management performance criterion is having poor performance level which is 44.4% only. Perhaps, Software Project D is a small scale software project thus the top management did not involve much and leave everything to the software project manager to manage. This is shows that successful project like Software Project A and Software Project D have obtained higher performance level for many criteria as illustrates in the Figure 7.



	A	B	C	D	E	F
🟦 Good	8	2	0	10	0	0
🟥 Average	6	7	3	3	7	3
🟩 Poor	0	5	11	1	7	11

Figure 7 Number of performance level for each criterion software projects

Additionally, Software Project B has only two criteria achieved good performance levels which are Defect and Organization. Most of performance criteria such as Schedule, Project Manager, Team Members, Training and Vendor are having poor performance level. Some of these poor performance level criteria are related to human involvement to the particular software project. Besides this, this project was a problematic project. It is overrun the actual planned schedule. This can be seen when the performance level of schedule is only 25% for this software project. Software Project E has average and poor performance level criteria only. This project is a problematic project. It is overrun the actual planned schedule. The performance level of Schedule criteria is just 15.9% only. Besides this, most of these criteria are related to human involvement to the particular software project. The cooperation among the project manager, users and vendors is less in this project. This can be seen when most of the metrics thresholds in the Project Manager, Team Members, Users, and Vendor performance criteria are below the average scale only.

Subsequently, there are no criteria in Software Project C and Software Project F obtained good performance level. In fact, the number of criteria that obtained average level are only three (3) for both software project respectively.

Most of criteria are having poor performance level. These software projects are not implemented at all. The development of this software project was unfinished. It was failure project in the beginning of development itself. Most of the metrics thresholds for the performance criteria are not compliance. The user involvement, frequent changing requirements from the top management, inabilities of project manager, less committed team members and less cooperation from users caused the failure of this software project.

Overall, most of the performance criteria of Software Project A and Software Project D having higher performance level compare to other our software projects (Software B, Software Project C, Software Project E and Software Project F). Thus, the higher the performance level of each criterion in a software project then higher is the software project success.

In summary, there is a significant positive relationship between the performance level of each criterion and the software project success. The higher the performance level (%) of the each criterion in a software project then the higher is the software project success. Hence, the null hypothesis **(**$H0_2$) "*The lower the performance level (%) of the each criterion then the higher is the software project success*" was successfully rejected in this study.

3.2.3 To Show That The Proposed Model Is Useful For The Software Project Managers In Monitoring And Control The Software Projects.

Consequently, to answer the third objective of this case study, a set of questionnaire were given to each software project managers who are involved in this case study as shown in the Table 6.

All six software project managers were strongly agreed that this proposed model is useful for them in monitoring and control the software projects. Besides this, they were strongly recommended that this proposed model can be used in future for monitoring and control the software projects.

Furthermore, all six software project managers were agreed that this proposed model enhanced the effectiveness of the software project managers in monitoring and control the software projects. Four of the software project managers were strongly agreed

Table 6 Proposed model evaluation results

Id	Items	Software Project Managers					
		1	2	3	4	5	6
1	Do you think this Metric-based Performance Model helpful in monitoring and control of your software projects?	SA	SA	SA	SA	SA	SA
2	Do you think this Metric-based Performance Model can enhance effectiveness in monitoring and control of your software projects?	A	A	A	A	A	A
3	Do you think this Metric-based Performance Model can ease the process of monitoring and control of your software projects?	A	A	SA	SA	A	A
4	Do you think this Metric-based Performance Model fits well for monitoring and control of your software projects?	SA	A	A	SA	A	A
5	Do you recommend the usage of Metric-based Performance Model for monitoring and control of software projects in future?	SA	SA	SA	SA	SA	SA

SA- strongly Agree A- Agree

that proposed model fits well for monitoring and control the software projects. Subsequently, three more software project managers were agreed that this proposed model can ease the process of monitoring and control the software projects.

In relation to these software project managers' feedbacks, it is summarized that this proposed model is useful for the software project managers mainly to monitor and control the software projects and recommend using this proposed model for monitoring and control of software projects in future.

3.3 Validity

This section explains the integrity of this case study research in term of validity and reliability. Case study researchers have developed a number of different approaches for increasing the integrity of qualitative research [27]. Below are the detail explanations of validity approaches.

3.3.1 Construct Validity

Construct validity achieved by developing the constructs in this case study research. The 14 identified performance criteria and the related metrics are the constructs for this case study. These constructs were derived through multiple sources of evidence. Consequently, SLR, structured interviews, expert validation are the sources of evidence for this case study. The constructs of this case study are basically developed through a real data from practitioners as well as from the literatures.

3.3.2 Internal Validity

Internal validity is to ensure that the study measures or tests what is actually intended as described by Andrew K. Shenton (2003)[26]. In this case, this case study uses the experienced software project managers as participants of this model evaluation. They handle many software projects in the real industry. Subsequently, these selected six software project managers are the people who handled the selected six software projects in the industry. Besides, the selected six software projects are the real projects at two different agencies at industry. Currently, four of these software projects which are Software Project A, B, D, E are being used in the agencies. However, the other two software projects C and F are not in use due to incompleteness and failures. Thus, this shows that this case study is tests in the actual contexts which increase the validity of this study.

3.3.3 External Validity

External validity is mainly concerned to what degree it is possible to generalize the findings in the particular research. This can be achieved through the use of a case study [27]. Subsequently, this study used multiple cases in generalizing the findings. Six software projects from Malaysian Public Sector were chosen as a unit of analysis in this case studies. Respectively, three software projects from the large scale and three more software projects from small scale software project environment. Based on these findings from two different software project environments, it is likely to generalize the findings of this case study to other settings or backgrounds.

3.3.4 Reliability

Accordingly, to increase the possibility that data collected are reliable and consistent across different time, we strictly follow:-

- Respondents for threshold session are senior software project managers from Malaysian Public Sector. These three senior software project managers are having vast experiences in a software project management for more than 15 years. They consult many of government agencies software projects development. Thus, they are the right participants for this threshold session.

- Data are collected in a same way using same instruments at each time. All the six selected project managers were used the same metrics lists as an instrument in this case study.

- The respondents of this case study are having the same background of working environment which is Malaysian Public Sector. Besides, all six case study respondents are software project managers. These software project managers are in the professional group at Malaysian Public Sector.

4.0 CONCLUSION

As a conclusion, this paper explained in detail the process of model formulation. It explained the entire processes that involved in this model formulation. It discussed how identified metrics are integrated and mapped using one of the core techniques in Grounded Theory which is Constant Comparison. This process followed by metrics validation by project management experts from Malaysian Public Sector. Then, these integrated metrics were described according to ISO/IEC TR 9126:2003. We continued with metrics categorizations based on PMBOK Iron Triangle Model.

Finally, the Metric Threshold Measurement Scale was created. This proposed model was evaluated and validated by conducting multiple case studies evaluation. The multiple case studies evaluation started with identifying threshold for each and every metrics using Metric Threshold Measurement Scale by the software project managers and followed by evaluation with selected software projects at various ministries at Malaysian Pubic Sector.

We also validate this model in a different project environments in this multiple case studies evaluation such as large scale outsource projects and small scale outsource projects. This is to examine our model suitability and flexibility in any types of software project environment at the public sector. This model can assist the software project managers in monitoring and controlling the software projects.

The Model comprises 14 performance criteria and 143 related metrics for the software projects monitoring and control. These metrics involves with threshold value to perform the performance scores. These processes required additional strength and time. Thus, this proposed model can be transformed into an automated tool in future, which comprises the processes and tasks that need to be performed by the project managers during the monitoring time. Moreover, automated tool may reduce the human errors in these processes as well as ease and fasten the processes.

Consequently, this Model can be enhanced by looking at the validated metrics impacts. Each identified metrics in this study can explored and analyzed whether it is giving high, low or no impacts on the software projects performance towards its success. Besides this, these identified metrics also can be prioritized according to its impact level in the software projects management. This may helpful for the project managers to deepen the software projects monitoring process.

Furthermore, to increase the reliabilities of these validated metrics, a statistical analysis can be performed in future. The statistical techniques such as co-relation and multi-regression can be applied into these validated metrics through survey questionnaires to the project managers.

Subsequently, this study determined the threshold value for the small and large scale software project environments in the context of Malaysian Public Sector. This threshold value can be enhanced for medium scale software projects in future. Project managers from this medium scale software project environment also will be benefited in future.

Additionally, in future, these validated metrics can be categorized according to software project management life cycle phases such as project initiation, project planning, project execution and project closure. By categorizing these metrics, project manager can monitor the software project in detail by each phase of software projects. This can be increased the success rate of software projects in future.

Acknowledgment

This research is funded by the Research University grant of Universiti Teknologi Malaysia (UTM) under the Vot no. 08H28. The authors would like to thank the Research Management Centre of UTM and the Malaysian ministry of education for their support and cooperation including students and other individuals who are either directly or indirectly involved in this project.

References

[1] James J. Jiang, Gary Kleim, Hsin-Ginn Hwang, Jack Huang, Shin-Yuan Hung. 2004. An exploration of the relationship between software development process maturity and project performance, *Information & Management*. 41: 279-288.

[2] Masateru Tsunoda, Tomoko Matsumura, Ken-i-chi Matsumoto. 2010. Modeling Software Project Monitoring with Stakeholders. 9th *International Conference on Computer and Information Science, 2010, IEEE.*

[3] The Standish Group International 2014. March 2015. *Chaos Report.*[Online] www.projectsmart.co.uk/docs/chaos-report.pdf.

[4] Reza Aliverdi et. al. 2013. Monitoring project duration and cost in a construction projects by applying statistic quality control charts. *International Journal of Project Management*. 31: 411-423,

[5] Wateridge et. al., 1998.How can IS/IT projects be measured for success. *International Journal of Project Management* 16(1): 59–63

[6] Rosana Stoica and Peggy Brouse, 2013. IT project Failure: A Proposed Four-Phased Adaptive Multi-Method Approach. *Procedia Computer Science*. 16: 728-736

[7] Oorschot et.al, 2009. Dynamic of Agile of Software Development. *International Conference of the System Dynamics Society*

[8] Barros et. al. 2000. Applying System Dynamics to Scenario Based Software Project Management. *International System Dynamics Conference*

[9] Jinhua et. al. 2008. Monitoring Software Projects With Earned Value Analysis And Use Case Point. Computer And Information Science, 2008. *ICIS 08. Seventh IEEE/ACIS International Conference* . 475 – 480

[10] Subramani Nagaiah, 2011. Seminar Personel ICT. Malaysian Administrative Modernization and Management Planning Unit (MAMPU), Malaysia

[11] Bernama Kuala Lumpur. (12 February 2013). [Online].http://www.mysinchew.com/node/78889.

[12] Mariayee Doraisamy et.al, 2014. The Need for Software Project Monitoring Methodology: An Empirical Investigation at Malaysian Public Sector. *International Journal of Innovating Computing*. 4(1)

[13] B.Kitchenham. 2007. " Guidelines For Performing Systematic Literature Reviews In Software Engineering (Version 2.3), Software Engineering Group, School Of Computer Science And Mathemathics. Keele University And Department Of Computer Science, University Of Durham,

[14] Hennie Boeije. 2002. A Purposeful Approach to the Constant Comparative Method in the Analysis of Qualitative Interviews, *Quality & Quantity* 36: 391–409

[15] LaRossa, R. 2005. Grounded Theory Methods and Qualitative Family Research. *Journal of Marriage and Family*. 67: 837–857

[16] Ming Li and Carol S.Smidths. 2003. A Ranking of Software Engineering Measures Based on Expert Opinion. *IEEE Transactions on Software engineering*. 29(9)

[17] K.P Srinivasan and T. Devi. 2014. Software metrics Validation Methodologies in Software Engineering. *International Journal of Software Engineering & Applications (IJSEA)*. 5(6),

[18] Rita J. Costello and Dar-Biau liu. 1995. Metrics for Requirements Engineering, *Journal of Systems Software*. 29:39-63

[19] The ISO/IEC 9126 Standard. 2010. *International Journal of Software Engineering & Applications (IJSEA)*.1(3). DOI: 10.5121/ijsea.2010.1302 17

[20] Yiannis Kanellopoulos et.al, 2010. Code Quality Evaluation Methodology using the ISO/IEC 9126 standard. International Journal of Software Engineering& Applications (IJSEA). 1(3): 17-36.

[21] Shareeful Islam and Paolo Falcarin. 2011. Measuring Security Requirements for Software Security, Cybernatic

Intelligent System (CIS). 2011 IEEE 10th International Conference . 70 – 75

[22] Kecia A.M.Ferreira et. al. 2012. Identifying Thresholds for Object Oriented Software Metrics, *The Journal of System and Software*. 85: 244-257

[23] Leanid Krautsevich et. al. 2010. Formal Approach To Security Metrics. What Does "More Secure" Mean For You? , *ECSA*, August 23-26, 2010, Copenhagen, Denmark, ACM.

[24] Reza Aliverdi et.al. 2013. Monitoring Project Duration And Cost In A Construction Projects By Applying Statistical Quality Control Charts. *International Journal of Project Management*. 31: 411-423

[25] Matthew Bass. 2006. Monitoring GSD Projects via Shared Mental Models: A Suggested Approach. *GSD'06 Proceedings of the 2006 International Workshop on Global Software Development for the Practitioner*, 34-37. ACM,

[26] Andreas M. Riege. 2003."Validity And Reliability Tests In Case Study Research: A Literature Review With "Hands-On" Applications For Each Research Phase", *Qualitative Market Research: An International Journa*l. 6(2): 75 – 86.

[27] R K Yin, 1994. Enhancing The Quality Of Case Studies In Health Services Research. *Health Service Res*. 1999 Dec; 34(5 Pt 2): 1209–1224.

[28] Anthony J. Viera et. al. 2005. Understanding Interobserver Agreement: The Kappa Statistic. *Research Series, Family Medicine*. 37(5): 360-363.

[29] Heinz K. Klein, Michael D. Myers. 1999. A set of Principles for Conducting and Evaluating Interpretive Field Studies in Information Systems. *MIS Quarterly*. 23(1): 67-93.

[30] B.A. Kitchenham, L.M. Pickard. 1998. Evaluating Software Engineering Methods And Tools – Part 10. Designing And Running A Quantitative Case Study.*Software Engineering Notes*. 23(3): 20–22.

Appendix A Metrics Description for Cost Performance Criteria

Metric ID	**Metric Name**	**Purpose of the metrics**	**Method of application**	**Measurement formula and data element computations (X)**	**Interpretation of measured value**	**Metric scale type**	**Input to measurement**	**Target audience**
C1	Exists of Actual cost plan	Does the actual cost plan is exist?	Determine the existence of actual cost plan.	X=A or X=B A= Actual cost plan is exists B = Actual cost plan is not exists	Actual cost plan should exist.	Nominal	Project Management Plan	Project Manager
C2	Exists of Estimated cost plan	Does the estimated cost plan is exist?	Determine the existence of estimated cost plan.	X=A or X=B A= Estimated cost plan is exists B = Estimated cost plan is not exists	Estimated cost plan should exist.	Nominal	Project Management Plan	Project Manager
C3	Exists of cost per activities plan	Does the cost per activities plan is exist?	Determine the existence of cost per activities.	X=A or X=B A= Cost per Activities Plan is exists B = Cost per Activities Plan is not exists	Cost per activities plan should exist.	Nominal	Project Management Plan	Project Manager
C4	% of changes in cost per activities	How much is percentage changes in the cost per activities?	Count the percentage of changes in a cost per activities.	X=A/B *100 A=Changes on actual cost per activities B= Actual cost per activities	The less percentage of changes is good.	Ratio	Project Management Plan	Project Manager
C5	% of usage Cost per activities	What is the percentage usage on cost per activities?	Count the percentages usage on cost per activities.	X=A/B *100 A=Total usage of cost per activities B= Actual cost per activities	The higher percentages of usage is according to cost per activities plan is good.	Ratio	Project Management Plan	Project Manager

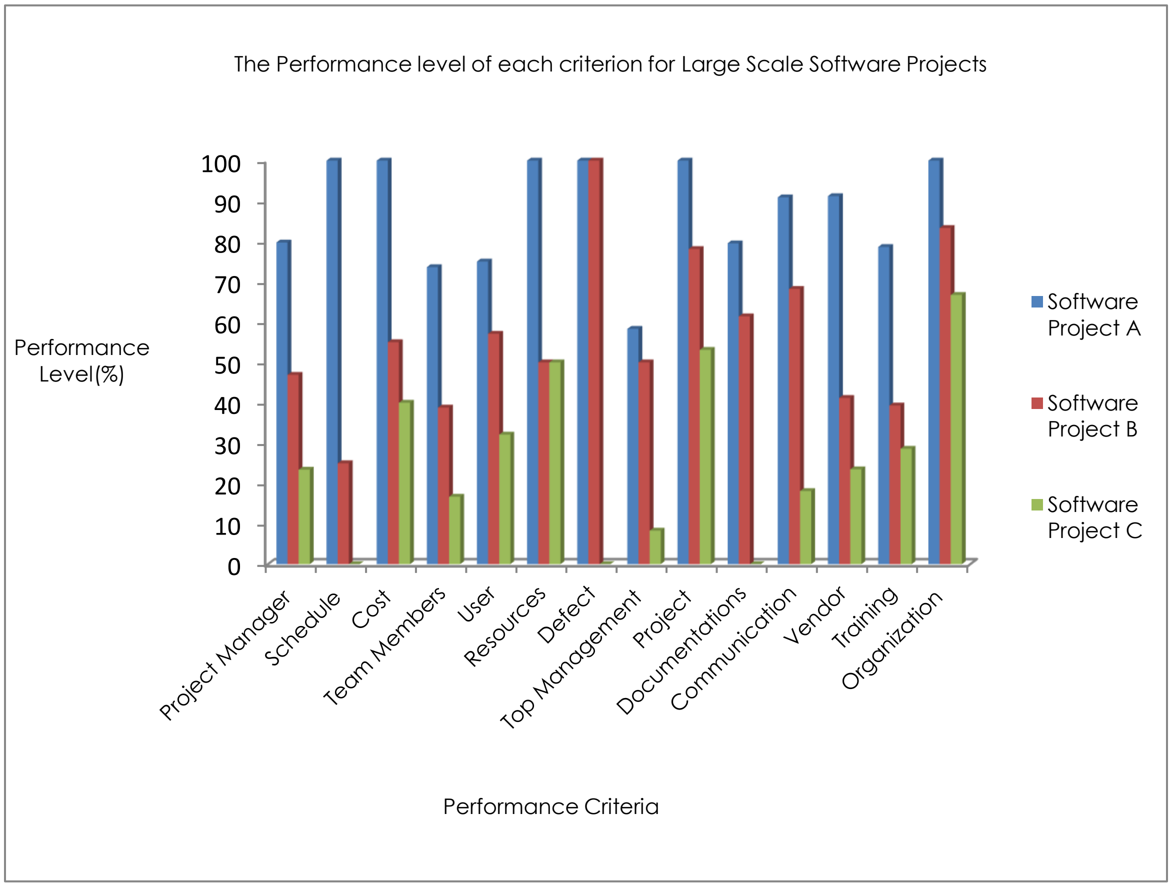
Appendix B Metric Performance Threshold Value Rating Scale for **Schedule Criteria - ** Large Scale

Metric ID	***Metric Name***	**Metric Performance Threshold Value**					**Threshold Indicator**
		Unknown or nil compliance ***(0)***	*weak compliance* ***(0.25)***	***Average*** *compliance* ***(0.5)***	*Above average compliance* ***(0.75)***	*strong compliance* ***(1)***	
S1	Total number of tasks per project	**No tasks**	**7-8 tasks**	**1-2 tasks**	**3-4 tasks**	**5-6 tasks**	Projects should have only average tasks only.
S2	Total number of tasks implemented per day/week/month	**0 tasks per month**	**1 tasks per month**	**2 tasks per month**	**3 tasks per month**	**More than 4 tasks per month**	The more number of tasks implemented in a month is good.
S3	Total number of functionalities implemented per week	**Not implemented any functionalities**	**-**	**At least 1 functionalities**	**2-3 functionalities**	**More than 4 functionalities**	The more functionalities are implemented in week is good.
S4	% of project delivered on-time as estimated	**Less than 60%**	**60-74%**	**75-84%**	**85-95%**	**More than 95%**	The higher the percentages of project delivered on-time as estimated is good.
S5	% of project delivered on-time as actual	**Less than 60%**	**60-69%**	**70-79%**	**80-90%**	**More than 90%**	The higher the percentages of project delivered on-time in actual is good.
S6	Total number of tasks finished at specified quality	**0 tasks**	**At least one task**	**2 tasks**	**3-4 tasks**	**5-6 tasks**	The more number of tasks finished at specified quality is good.
S7	% of project delivered according to each planned activities	**Less than 60%**	**60-74%**	**75-84%**	**85-95%**	**More than 95%**	The higher the percentages of project delivered according to each planned activities is good.

Appendix C Metric Performance Threshold Value Rating Scale for **Schedule Criteria** - Small Scale

Metric ID	***Metric Name***	**Metric Performance Threshold Value**					**Threshold Indicator**
		Unknown or nil compliance ***(0)***	*weak compliance* ***(0.25)***	***Average** compliance* ***(0.5)***	*Above average compliance* ***(0.75)***	*strong compliance* ***(1)***	
S1	Total number of tasks per project	**1 tasks**	**2 tasks**	**3 tasks**	**4 tasks**	**5 tasks**	Projects should have only average tasks only.
S2	Total number of tasks implemented per day/week/month	**0 tasks per month**	**-**	**1 task per month**	**2 tasks per month**	**More than 3 tasks per month**	The more number of tasks implemented in a month is good.
S3	Total number of functionalities implemented per week	**Not implemented any functionalities**	**-**	**At least 1 functionalities**	**2 functionalities**	**More than 2 functionalities**	The more functionalities are implemented in week is good.
S4	% of project delivered on-time as estimated	**Less than 60%**	**60-74%**	**75-84%**	**85-95%**	**More than 95%**	The higher the percentages of project delivered on-time as estimated is good.
S5	% of project delivered on-time as actual	**Less than 60%**	**60-69%**	**70-79%**	**80-90%**	**More than 90%**	The higher the percentages of project delivered on-time in actual is good.
S6	Total number of tasks finished at specified quality	**0 tasks**	**0 tasks**	**1 task**	**2-3 tasks**	**4-5 tasks**	The more number of tasks finished at specified quality is good.
S7	% of project delivered according to each planned activities	**Less than 60%**	**60-74%**	**75-84%**	**85-95%**	**More than 95%**	The higher the percentages of project delivered according to each planned activities is good.

Appendix D The performance level for each criterion for large scale software projects


The Performance level of each criterion for Large Scale Software Projects
Performance Level(%)
Project Manager
Schedule
Cost
Team Members
User
Resources
Defect
Top Management
Project
Documentations
Communication
Vendor
Training
Organization
Performance Criteria
Software Project A
Software Project B
Software Project C