Development of software metrics for improving the quality of the under graduate student projects in computer science /information science/information technology/computer engineering

Saravanan Venkataraman Tirumalai
Department of Computer Science
Majmaah University
Al Majmaah, Kingdom of Saudi Arabia

Abdullah Al Hussein
Department of Computer Science
Majmaah University
Al Majmaah, Kingdom of Saudi Arabia

Manjunatha Siddappa Lakshmana
Department of Computer Science
Majmaah University
Al Majmaah, Kingdom of Saudi Arabia

Summary
At present, many Universities educates good number of undergraduates in field of Computer Science/Information Sciences/Information Technology/Computer Engineering across the Kingdom of Saudi Arabia. The students' carryout the graduation project as a part of study plan in the curriculum. Most of the Universities offer the graduate project in last two semester of the program. However, the quality of the student projects is of greater concern in prospective of entrepreneurship or good job in software industry. In general, most of the students do not take the project work seriously and end up with low quality. This may leads to the students with surficial knowledge in the subject and find difficult in getting a suitable job in the industry. This research project measures the quality of the graduate project by introducing a software metrics tool to be followed by the instructor as the work progresses by the student. 44 software metrics were developed and tested with a prototype. These metrics are attached with the under graduate student projects for evaluation. This model allows the student to carry out more quality works and thereby increase the knowledge level that leads to better job prospects

Key words:
Software Metrics, Student Project, Recommendation System

1. Introduction

The enrollment number of students in higher education is increasing every year. On the other hand, the Universities/Colleges role is also increasing in producing quality graduates. The prime priority of the instructor is to help student to acquire good knowledge and understanding ability in the course, which he/she undergoes study. In addition to his/her learning, a good project gives completeness of graduation. A good project helps to display his/her ability in the competing world. Many higher education institutions in the Kingdom of Saudi Arabia are enforcing quality standards in the teaching learning process. The challenge at present is to enforce a standard for increasing the quality of the undergraduate projects in Computer Science/Information Sciences/Information Technology/Computer Engineering. This challenge is achieved in this research project by means of development of software metrics and a recommendation system to improve the quality of the undergraduate student projects in computer science/information science/information technology/computer engineering. In this research, we developed 44 software metrics, which plays an important role in developing standard in evaluating the graduation project. This research also helps to improve the quality of the software projects. Based upon the implementation results, this project can be extended to hardware oriented or hardware based projects.

2. Literature Review

In order to meet the objective of the project, a detailed review of literature has been carried out in the following context:

a. Software Metrics

Abdullah Saleh Al Sadaawi [1] discussed the universal basic education followed in Saudi Arabia. The kingdom initially employed a rapid quantitative educational strategy, later developed a qualitative focus to improve standards of education delivery and quality of student outcomes. The author points out the growing demand for national assessment standards for all key subject areas to monitor students' learning progress. This study acknowledges extant research on this important topic and offers a strategy of national assessment to guide educational reform.

Manuscript received October 5, 2017
Manuscript revised October 20, 2017

Bavota, G et al.,[2] presents guidelines on how to build recommendation systems and how to evaluate them. The authors also highlight some of the challenges exist in the construction of recommendation systems. The author mainly focuses on refactoring the software process in large software systems. Various approaches for refactoring are discussed in detail by the author.

Calvin Selig and Sallie Henry [3] proposed a design tool to quantitatively evaluate student projects. The authors are using a reduced life cycle to increase the efficiency of the software development process by implementing metrics analysis as early as possible. The student assignment / home work is tested with the developed software and this software generates quality metrics. These metrics are used to check the quality of the work submitted by a student. The authors uses ADLIF(Ada Like Design Language based on Information Flow) to create an intermediate step between the general design and coding phases in the life cycle of a system.

Fahad Al Faadel et al., [4] investigates the reasons for the success and failure of IT projects in Saudi Arabia. Common reasons for failure of IT projects in Saudi Arabia were identified, with characterizing organizational culture and conflict of interest and the instability and lack of clarity of the set of requirements as the most important causes for failure.

Ghaleb Hamad Alnahdi [5] discusses the possibility of adapting the suggestions by Hargreaves and Shirley in their book "The Fourth Way." This paper discusses the topic of educational change and reform through three main points. First, it reviews the most important advantages and disadvantages that characterize the three period. Second, it will extract the main principles proposed as the fourth way (the principles of how education should be changed in the future) and discuss whether or not officials in Saudi Arabia will be able to apply it.

Gunnar Schröder et al.,[6] focuses on clearly defining the goal of an evaluation and how this goal relates to the selection of an appropriate metric. The author discusses several well-known accuracy metrics and analyze how these react different evaluation goals. The author also presents some less well-known metrics as well as a variation of the area under the curve measure that are particularly suitable for the evaluation of recommender systems in e-commerce applications. Improved metrics for the recommender systems were presented with the results and justification.

Iman Avazpour et al., [7] reviews a range of evaluation metrics and measures as well as some approaches used for evaluating recommendation systems. The authors also provided a detailed discussion on the quantitative evaluation techniques used to compare recommendation systems. The key dimensions of the recommendation systems namely correctness, coverage, diversity, trustworthiness, recommender confidence, novelty, serendipity, utility, risk, robustness, learning rate, usability, scalability, stability, privacy and user preference are discussed in detail. The authors also discuss the various categories of the dimensions such as a recommendation-centric, user-centric, system-centric and delivery-centric.

The Infosys white paper [8] illustrates a comprehensive measurement model, which can be adopted to inculcate a culture of continuous improvement in the overall software-testing life cycle. This paper also illustrates the importance of measurement and metrics in the whole software testing process and provides an insight into various process/product efficiency and effectiveness metrics.

Jayabalaraja et al [9] aimed to identify the competence of the developers in their selected skill set relevant to the assigned tasks. It provides the developers those who are worked in the minimum skill set components and identified as a weak set of employees through equivalence algorithm of the rough set theory. The functional attributes are observed and analyzed to find out the maximal error process which leads to the identification employees set those are made more modification with low level expert knowledge in the working area or project. The observation, analysis and the experimental procedures using Quick reduct were presented in this paper.

Marko Gasparic [10] presents state-of-the art recommendation systems for software engineering. The typical functionalities offered by existing recommendation systems and its technical design are discussed in detail. The author proposes a tool that will show a content relevant information and recommendations to the developers during the programming tasks. The author also presents a detailed architecture, which consists of logical view, process view, development view, physical view and scenarios. The user interface developed by the author is presented very clearly in this thesis. The author plans to modify the measurement framework to improve the performance and enable new functionalities as an extension work.

It is very clear from the above reviewed literatures that, no authors has developed a software metrics/recommendation systems to crosscheck the standards of the student software projects as stated in the summary of the research work.

3. Methodology

The project methodology contains 12 phases as listed below

i. Various kinds of software/hardware project domains such as database projects, networking projects, and application projects were identified.
ii. A questionnaire designed to meet the required recommendation system based on various domains in order to evaluate the exist practices in assessment of graduation project.
iii. As defined in the plan of the proposal, the team visited the following Universities and collected the data for initial study.
    i. King Saud University, Riyadh
    ii. Imam University, Riyadh
    iii. King Faisal University, Al Asha
    iv. King Khalid University, Abha
    v. King Abdul Aziz University, Jeddah
iv. 96 samples were collected from the above mentioned university in person and through google spread sheet. Detailed analysis was carried out in the data set using SPSS and the detailed results are tabulated in the subsequent sessions.
v. A prototype was developed by considering these software metrics. Microsoft .Net with MS-Access was used to develop the prototype. This prototype is developed as a recommendation system by referring the literatures [8] and [9].
vi. This recommendation system is developed by considering the software engineering life cycle stages (Analysis, Design, Coding, Testing, and Implementation). Metrics will be developed in all these life cycle stages.
vii. As the student completes a life cycle stage, the supervisor review the software metrics provided by the system.
viii. The recommendation system takes the input from the supervisor, evaluates it and gives the qualitative and quantitative output.
ix. The qualitative output gives the students a set of missing concepts/analysis data/design data etc.,
x. The quantitative output gives the students a numeric value. The lesser the value, the lower the quality.
xi. The quantitative output has an optimal values as fixed by the supervisor. 70% is considered as a optimal value to test the developed prototype. The student needs to repeat the tasks until the optimal value is arrived.
xii. The students incorporate the qualitative outputs positively after the discussion with the supervisor.

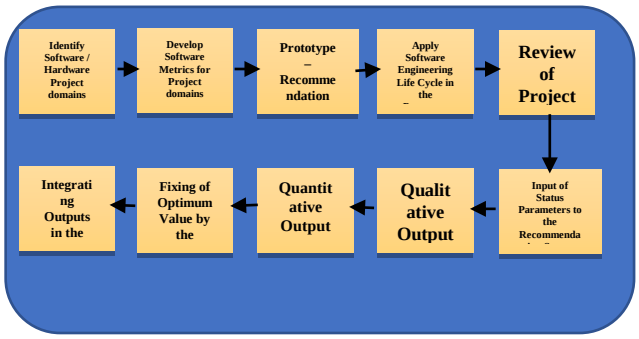

Fig. 1 Research Design

4. Results and Discussion

The following stages in the software engineering are used in the study. Each stages are named with group numbers for the initial analysis.
Group 1: Software Requirements Management
Group 2: Software Project Planning
Group 3: Software Product Engineering
Group 4: Software Design
Group 5: Software Coding Standards and Code Reviews
Group 6: Software Programming Practices
Group 7: Software Testing
Group 8: Software Configuration Management
Group 9: Software Quality Assurance
Group 10: Technology Change Management

Five metrics are identified for each group and a total of 50 metrics are identified for the project. These metrics are tabulated and surveyed with the identified Universities among the experts. There were 96 responses.

The results are tabulated and analyzed using SPSS. "SPSS is a comprehensive system for analyzing data. SPSS can take data from almost any type of file and use them to generate tabulated reports, charts, and plots of distributions and trends, descriptive statistics, and complex statistical analysis. SPSS is the acronym of Statistical Package for the Social Science. SPSS is one of the most popular statistical packages which can perform highly complex data manipulation and analysis with simple instructions. SPSS has scores of statistical and mathematical functions, scores statistical procedures, and a very flexible data handling capability.

The developed system is novel and unique. There are no tools/models available to evaluate the student projects in computer science/information technology/information sciences/computer engineering. It is noted from the review that, software metrics is not also used for student project evaluation. It is very clear from the reviewed literature that,

the approaches used and tested are not available previously. The following session presents the detailed analysis of the developed model.

4-1: Factor Analysis

Factor analysis denotes a class of procedures primarily used for data reduction and summarization. The factor analysis was run with the 44 metrics, so as to reduce the metrics into sizable factors so that it will be easier for further analysis. The 44 metrics were extracted by the principal component method using varimax rotation method in which the rotation minimizes the number of metrics with high loading factor and thereby it enhances the interpretability of the factor. The indicators refers to the developed metrics. The 44 metrics were grouped into five factors and each factor with Eigen value more than one. The five factors extracted were named as testing configuration, product and quality engineering, Requirement and change management, coding standards and planning and design standards.

4-2: Descriptive Statistics

Table 1: Requirement and Change Management

Requirement and Change Management	Mean	Std. Deviation
The students selects their own interested specified project as a part of the course requirement.	3.68	1.091
The students select their project with innovative theme as part of the course requirement.	3.60	1.110
The students select their project to improved existing version in software development.	3.36	1.097
The student is able to do their project as a software application.	3.71	.939
The student is able to do their project as a hardware product.	2.76	1.103
The student presented the correct data model with the help of a diagram. (Entity-Relationship diagram or Unified Modelling Language diagram etc.,).	3.67	1.102
Student follow a plan for managing technology changes.	3.27	.900
Student able to update his/her knowledge as demand of technology and advances.	3.49	.951
Student is able to evaluate and incorporate if any new technology arrives or exists to previous.	3.41	.980

The table (table 1) shows the mean and standard deviation of the respondent's opinion about the various reasons of Requirement and Change Management. Mean and standard deviation is calculated in order to measure the central tendency. The highest mean score is 3.71, which denotes that the respondents strongly agree the main reason of Requirement and Change Management that the students is able to do their project as a software application.

Table 2: Planning and Design Standards

Planning and Design Standards	Mean	Std. Deviation
Adequate resources are provided for planning the software project (e.g., project management tools such as MS-Project etc.,).	3.65	.995
Sufficient resources are provided for performing the software engineering tasks (Resources: Computer Assisted Software Engineering Tools).	3.74	.954
Normalization is used to design the databases (First Normal Form / Second Normal Form / Third Normal Form).	3.60	.864
When naming functions, student includes a description of the value being returned, such as *GetCurrentWindowName()*.	3.36	.975

The table (table 2) shows the mean and standard deviation of the respondent's opinion about the various reasons of Planning and Design Standards. Mean and standard deviation is calculated in order to measure the central tendency. The highest mean score is 3.74, which denote the respondents strongly agree that the main reason of Planning and Design Standards is that Sufficient resources are provided for performing the software engineering tasks (Resources: Computer Assisted Software Engineering Tools).

Table 3: Coding Standards

Coding Standards	Mean	Std. Deviation
The student uses customary opposite pairs in variable names, such as *min/max*, *begin/end*, and *open/close*.	3.29	.857
Students uses a standard size for an indent, such as four spaces, and use it consistently.	3.20	.913
When naming tables, student express the name in the singular form. For example, use `Employee` instead of `Employees`.	3.37	.886
The student keep the lifetime of variables as short as possible.	3.23	.888
The scope of variables are as small as possible to avoid confusion and to ensure maintainability.	3.32	.864
Forced data conversion, sometimes referred to as variable coercion or casting, which may yield unanticipated results is avoided.	3.29	.807

The table (table 3) shows the mean and standard deviation of the respondent's opinion about the various reasons of Coding Standards. Mean and standard deviation is calculated in order to measure the central tendency. The highest mean score is 3.37, which denote the respondents agree that the main reason of Coding Standards is that when naming tables, student expresses the name in the singular form. For example, use Employee instead of Employees.

Table 4: Testing and Configuration Management

Testing and Configuration Management	Mean	Std. Deviation
The estimation (e.g., size, cost, and schedule) are documented for use in planning and tracking the software project (Size: Number of Modules, Cost: Duration / Money etc., Schedule: Road map)	3.13	1.098
The students uses standard measurement models (ISO/IEC TS 14143-2: 2007 / 2012) to determine the functionality and quality of the developed project.	2.97	1.090
The student test the software using any testing software (e.g test Environment Toolkit (TETWare)	3.10	1.081
The student chose the appropriate testing technique (black box / white box).	3.19	1.127
The student uses live test data (part of on-going process) for testing.	3.09	1.047
Test cases are generated for all the developed modules.	3.17	1.063
Test reports are prepared and documented in the project report.	3.27	1.138
The students uses software configuration management activities like configuration identification, configuration control, configuration status accounting, configuration auditing etc.,	2.98	1.046
The developed project follows the standard of the configuration management (e.g IEEE 828-2012 standard).	2.84	1.029
The students identifies the configuration items / units.	2.95	1.060
The student follow a standard guidelines to control changes to configuration items/units.	2.86	1.082
The students uses the standard version control guidelines (Major version/Minor Version/First release etc.)	2.83	1.130
The student planned the activities for managing software quality (e.g Formulating a quality management plan, Controlling change, Audits, Conducting formal technical reviews).	3.08	.867
The students uses a standard method for quality management (e.g Auditing, Document Analysis etc.).	3.10	.957
The students uses measurable and prioritized goals for managing the quality. (e.g., functionality, reliability, maintainability and usability).	3.16	.921

The table (table 4) shows the mean and standard deviation of the respondent's opinion about the various reasons of Testing and Configuration Management. Mean and standard deviation is calculated in order to measure the central tendency. The highest mean score is 3.27, which denote, the respondents agree that the main reason of Testing and Configuration Management is that test reports are prepared and documented in the project report.

Table 5: Product and Quality Engineering

Product and Quality Engineering	Mean	Std. Deviation
Project supervisor regularly review and record the activities of the project.	4.06	1.074
Students develop the software according to the project's defined objectives of the project.	3.81	.987
Students maintain consistency across the developed modules.	3.60	.957
The developed project follows a written college policy for performing the software engineering activities.	3.66	.927
The student identified all the modules for the project by considering the scope and objective.	3.50	.962
At the beginning of every routine, the student provides standard comments, indicating the routine's purpose, assumptions, and limitations.	3.38	1.008
The supervisor helps the student in identifying a standard method for quality management.	3.63	.976
The supervisor reviews the quality management tasks regularly.	3.64	1.058
Student consults the supervisor before incorporating any new technology in enhance the level of the project.	3.90	.888
The supervisor reviews the technology management tasks regularly.	3.73	.968

The table (table 5) shows the mean and standard deviation of the respondent's opinion about the various reasons of Product and Quality Engineering. Mean and standard deviation is calculated in order to measure the central tendency. The highest mean score is 4.06, which denote, the respondents strongly agree that the main reason of Product and Quality Engineering is that Project supervisor regularly review and record the activities of the project.

4-3: One-Way Anova between Universities and the Factors Used in the Study

Table 6: ONE-WAY ANOVA between Universities and usage of software development factors

Factors	University	Sum of Squares	df	Mean Square	F	Sig.
Requirement and Change Management	Between Groups	6.067	5	1.213	2.188	0.062
	Within Groups	49.918	90	.555		
	Total	55.984	95			
Coding Standards	Between Groups	6.606	5	1.321	2.972	0.016*
	Within Groups	40.001	90	.444		
	Total	46.606	95			
Planning and Design standards	Between Groups	6.638	5	1.328	2.529	0.034*
	Within Groups	47.235	90	.525		
	Total	53.872	95			
Testing & Configuration management	Between Groups	3.215	5	.643	0.950	0.453
	Within Groups	60.896	90	.677		
	Total	64.111	95			
Product and Quality engineering	Between Groups	4.677	5	.935	1.591	0.171
	Within Groups	52.913	90	.588		
	Total	57.590	95			

* Significant at 5 percent level.

The table (table 6) shows the one way ANOVA result between the Universities and usage of software development standards. One way ANOVA test was used to determine whether the respondent's opinion on testing and configuration management, product and quality engineering, requirement and change management, coding standards and planning and design standards differed among the universities and usage of software development standards. It could be inferred from the table that coding standards and planning and design standards is significant and thus it implies there exist a difference based on universities. Testing configuration, Product and quality engineering and Requirement and change Management are not significant and thus it implies that there is no difference based on universities and usage of software development standards.

4-4: Correlation Analysis

Table 7: Karl- Pearson Correlation for usage of software development factors

	RC	CS	PD	TC	PQ
RC	**1**				
CS	**0.560****	**1**			
PD	**0.638****	**0.626****	**1**		
TC	**0.608****	**0.623****	**0.626****	**1**	
PQ	**0.719****	**0.617****	**0.616****	**0.616****	**1**

From the above table (table 7) it is inferred that the factors testing configuration management, product and quality engineering, Requirement and change management, coding standards and planning and design standards are significantly and positively correlated to each other.

4-5: Multiple Regression Analysis

Table 8 Influence of dependent variable and independent variables

Model Summary					
R	**R Square**	**Adjusted R Square**	**Std. Error of the Estimate**	**F**	**Sig.**
0.742[a]	0.550	0.531	0.52587	27.861	**0.000[b]***
a. Predictors: Planning and Design Standards, Coding Standards, Product and Quality Engineering, Testing and Configuration Management					

*Significant at 5% level

From the above table (table 8) it refers that the R2 value is 0.550, which implies that 55% changes in dependent variable due to independent variables. R value as 0.742 shows a moderate and significant relationship (F= 27.861) between the factors.

Table 9 Multiple Regression results between dependent and independent variables

	Unstandardized Coefficients		Standardized Coefficients	t	Sig.
	B	**Std. Error**	**Beta**		
(Constant)	.462	.301		1.533	.129
Planning Design	.195	.103	.191	1.892	.062
Coding Standards	.031	.108	.028	.290	.773
Testing Configuration	.193	.089	.207	2.179	**.032***
Product Quality	.430	.097	.436	4.424	**.000***
Dependent Variable: Requirement Change Predictors: Planning and Design Standards, Coding Standards, Product and Quality Engineering, Testing and Configuration Management					

*Significant at 5% level

The table (table 9) shows that the multiple regression results and the relationship between the factors. From the table it could be inferred that requirement and change management is influenced by all the factors (planning and design standards, coding standards, product and quality engineering and testing and configuration management).

Table 10 Demographic Data Distribution

University	Frequency	Percent	Cumulative Percent
Majmaah University, Al Majmaah	8	8.3	8.3
King Abdul Aziz University, Jeddah	40	41.7	50.0
King Faisal University, Al Asha	14	14.6	64.6
King Khalid University, Abha	10	10.4	75.0
Jazan University, Jazan	3	3.1	78.1
Any other Please specify	21	21.9	100.0
Total	**96**	**100.0**	

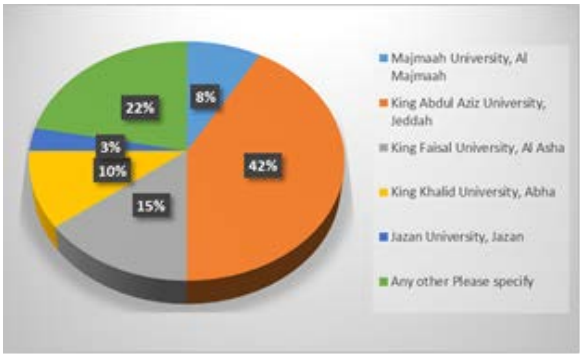

Fig. 2 Demographic Data Distribution

4-6: Confirmatory Factor Analysis

Table 11: Loadings of indicators for Requirement and Change Management, Planning and Design Standards, Coding Standards, Testing and Configuration Management and Product and Quality Engineering

Construct	Indicators	Mean	Standard Deviation	Loadings
Requirement Change Management	RC1	3.68	1.091	0.608
	RC2	3.60	1.110	0.828
	RC3	3.36	1.097	0.785
	RC4	3.71	.939	0.706
	RC6	3.67	1.102	0.803
	RC7	3.27	.900	0.861
	RC8	3.49	.951	0.829
	RC9	3.41	.980	0.808
Planning and Design Standards	PD1	3.65	.995	0.818
	PD2	3.74	.954	0.740
	PD3	3.60	.864	0.794
	PD4	3.36	.975	0.822
Coding Standards	CS1	3.29	.857	0.766
	CS2	3.20	.913	0.867
	CS3	3.37	.886	0.681
	CS4	3.23	.888	0.874
	CS5	3.32	.864	0.876
	CS6	3.29	.807	0.752
Testing and Configuration Management	TC1	3.13	1.098	0.758
	TC2	2.97	1.090	0.684
	TC3	3.10	1.081	0.714
	TC4	3.19	1.127	0.786
	TC5	3.09	1.047	0.760
	TC6	3.17	1.063	0.809
	TC7	3.27	1.138	0.773
	TC8	2.98	1.046	0.851
	TC9	2.84	1.029	0.805
	TC10	2.95	1.060	0.827
	TC11	2.86	1.082	0.821
	TC12	2.83	1.130	0.857
	TC13	3.08	.867	0.687
	TC14	3.10	.957	0.824
	TC15	3.16	.921	0.753
	PQ1	4.06	1.074	0.754
	PQ2	3.81	.987	0.772
	PQ3	3.60	.957	0.753
	PQ4	3.66	.927	0.769
	PQ5	3.50	.962	0.780
	PQ6	3.38	1.008	0.784
	PQ7	3.63	.976	0.841
	PQ8	3.64	1.058	0.850
	PQ9	3.90	.888	0.807
	PQ10	3.73	.968	0.803

From the above table (table 11), it can be seen that confirmatory factor analysis and the estimation of the item-loading of the constructs for testing and configuration management, product and quality engineering, Requirement and change management, coding standards and planning and design standards. The items with loading less than 0.6 should be eliminated. The items with loadings lower than 0.6 were removed in order not to endanger the explanatory power of the model. The construction for testing and configuration management, product and quality engineering, Requirement and change management, coding standards and planning and design standards has the entire item with loading above 0.6.

Table 12: AVE, Composite Reliability and Cronbach Alpha

Constructs	Composite Reliability	AVE	Cronbach Alpha
Requirement Change Management	0.926	0.612	0.955
Planning and Design Standards	0.872	0.631	0.806
Coding Standards	0.917	0.650	0.891
Testing and Configuration Management	0.959	0.612	0.955
Product and Quality Engineering	0.944	0.627	0.935

The table (table 12) shows the relationship between Composite reliability, AVE and Cronbach alpha. The

reliability is assessed by examining the loading (or correlations). According to general rule of thumb, the statements with loading less than 0.6 should be eliminated. The reliability measures the shared variance between and individual item and its construct is higher than the error variance. Discriminant validity is assessed based on Average Variance Extracted (AVE), which shows the average variance shared between a construct and its measure. This measure should be higher than the variance shared between the construct and other constructs in the model. From the above table (table 13) it can be seen that path Coefficient, t – statistics and P values. Since t-values is >1.96, it indicates that there exists significant relationship between the factors, and also P values. Therefore, it can be concluded that the path is significant, and the model is good fit.

Table 13: Model estimates

Constructs	Original Sample	Sample Mean	Standard Deviation	SMPLS path Coefficient	t-statistics	P Values
RC -> PD	0.638	0.645	0.056	0.638	11.335	0.000
PD -> CS	0.626	0.637	0.086	0.626	7.306	0.000
CS -> TC	0.623	0.627	0.068	0.623	9.098	0.000
TC -> PQ	0.616	0.628	0.057	0.616	10.892	0.000

Table 14: Cross Loadings

	Coding Standards	Planning and design standards	Product and Quality engineering	Requirement Change Management	Testingand Configuration management
RC1	0.430	0.268	0.351	**0.608**	0.251
RC2	0.382	0.494	0.504	**0.828**	0.379
RC3	0.438	0.502	0.456	**0.785**	0.527
RC4	0.356	0.414	0.416	**0.706**	0.237
RC6	0.422	0.547	0.689	**0.803**	0.428
RC7	0.436	0.582	0.725	**0.861**	0.683
RC8	0.620	0.605	0.676	**0.829**	0.610
RC9	0.418	0.474	0.558	**0.808**	0.539
PD1	0.411	**0.818**	0.599	0.543	0.447
PD2	0.453	**0.740**	0.429	0.414	0.286
PD3	0.425	**0.794**	0.472	0.566	0.492
PD4	0.666	**0.822**	0.463	0.500	0.705
CS1	**0.766**	0.494	0.543	0.384	0.517
CS2	**0.867**	0.665	0.482	0.541	0.618
CS3	**0.681**	0.456	0.341	0.261	0.224
CS4	**0.874**	0.520	0.453	0.507	0.536
CS5	**0.876**	0.510	0.619	0.516	0.560
CS6	**0.752**	0.309	0.542	0.437	0.456
TC1	0.594	0.545	0.550	0.551	**0.758**
TC2	0.368	0.514	0.482	0.482	**0.684**
TC3	0.560	0.336	0.455	0.424	**0.714**
TC4	0.636	0.629	0.525	0.595	**0.786**
TC5	0.563	0.466	0.415	0.422	**0.760**
TC6	0.643	0.608	0.612	0.669	**0.809**
TC7	0.552	0.555	0.534	0.697	**0.773**
TC8	0.484	0.518	0.382	0.382	**0.851**
TC9	0.294	0.476	0.401	0.416	**0.805**
TC10	0.320	0.418	0.413	0.350	**0.827**
TC11	0.299	0.418	0.426	0.344	**0.821**
TC12	0.413	0.498	0.405	0.430	**0.857**
TC13	0.319	0.288	0.393	0.259	**0.687**
TC14	0.394	0.407	0.502	0.352	**0.824**
TC15	0.515	0.480	0.543	0.453	**0.753**
PQ1	0.370	0.480	**0.754**	0.582	0.229
PQ2	0.509	0.563	**0.772**	0.480	0.432
PQ3	0.507	0.432	**0.753**	0.595	0.385
PQ4	0.510	0.576	**0.769**	0.677	0.350
PQ5	0.459	0.547	**0.780**	0.756	0.520
PQ6	0.611	0.521	**0.784**	0.494	0.667
PQ7	0.434	0.438	**0.841**	0.474	0.589
PQ8	0.484	0.474	**0.850**	0.477	0.537
PQ9	0.533	0.488	**0.807**	0.619	0.407
PQ10	0.418	0.393	**0.803**	0.654	0.476

From the above table (table 14) it can be seen that confirmatory factor analysis and the estimation of the item-Cross loading of the constructs for testing and configuration management, product and quality engineering, Requirement and change management, coding standards and planning and design standards. The items with loading less than 0.6 should be eliminated in order not to endanger the explanatory power of the model. The construct for testing and configuration management, product and quality engineering, Requirement and change

management, coding standards and planning and design standards has the entire item with loading above 0.6.

4. Conclusion and Recommendations

Developed Metrics

The following developed metrics are confirmed with the detailed analysis

Software Requirements Management
1. Selection of own interested specified project as a part of the course requirement.
2. Project with innovative theme as part of the course requirement.
3. Selection of project to improved existing version in software development.
4. Project as a software application.
5. Project as a hardware product.
Software Project Planning
6. The estimation (e.g., size, cost, and schedule) are documented for use in planning and tracking the software project (Size: Number of Modules, Cost: Duration / Money etc., Schedule: Road map)
7. Adequate resources are provided for planning the software project (e.g., project management tools such as MS-Project etc.,).
8. Regular review and record the activities of the project by the project supervisor.
Software Product Engineering
9. Develops the software according to the project's defined objectives of the project.
10. Maintain consistency across the developed modules.
11. Follow a written college policy for performing the software engineering activities.
12. Sufficient resources are provided for performing the software engineering tasks (Resources: Computer Assisted Software Engineering Tools).
13. Use of standard measurement models (International Organization for Standardization (ISO)/ International Electrotechnical Commission (IEC) TS 14143-2: 2007 / 2012) to determine the functionality and quality of the developed project.
Software Design
14. Identification all the modules for the project by considering the scope and objective.
15. Presenting the correct data model with the help of an a diagram. (Entity-Relationship diagram or Unified Modeling Language diagram etc.,).
16. Use of normalization to design the database (First Normal Form / Second Normal Form / Third Normal Form).
Software Coding Standards and Code Reviews
17. Provide standard comments, indicating the routine's purpose, assumptions, and limitations.
18. Use of customary opposite pairs in variable names, such as *min/max*, *begin/end*, and *open/close*.
19. Use of a standard size for an indent, such as four spaces, and use it consistently.
20. When naming tables, express the name in the singular form. For example, use *Employee* instead of *Employees.*
21. When naming functions, includes a description of the value being returned, such as *GetCurrentWindowName().*
Software Programming Practices
22. Keep the lifetime of variables as short as possible.
23. Scope of variables are as small as possible to avoid confusion and to ensure maintainability.
24. Forced data conversion, sometimes referred to as variable coercion or casting, which may yield unanticipated results is avoided.
Software Testing
25. Software is tested using any testing software (e.g test Environment Toolkit (TETWare), Mozilla Testopia, test Link, qaManager, Litmus etc.,).
26. Choose the appropriate testing technique (black box / white box).
27. Use live test data (part of on-going process) for testing.
28. Generate the test cases for all the developed modules.
29. Prepare and document the test reports in the project report.
Software Configuration Management
30. Uses software configuration management activities like configuration identification, configuration control, configuration status accounting, configuration auditing etc.,
31. Project follows the standard of the configuration management (e.g IEEE 828-2012 standard).
32. Identify the configuration items / units.
33. Follow a standard guidelines to control changes to configuration items/units.
34. Uses the standard version control guidelines (Major version/Minor Version/First release etc.,).
Software Quality Assurance
35. Plan the activities for managing software quality (e.g Formulating a quality management plan, Controlling change, Audits, Conducting formal technical reviews).
36. Uses a standard method for quality management (e.g Auditing, Document Analysis etc.,).
37. Uses measurable and prioritized goals for managing the quality. (e.g., functionality, reliability, maintainability and usability).
38. Identify a standard method for quality management.
39. Review the quality management tasks regularly.
Technology Change Management
40. Follow a plan for managing technology changes.
41. Able to update his/her knowledge as demand of technology and advances.
42. Consults before incorporating any new technology in enhance the level of the project.
43. Evaluate and incorporate if any new technology arrives or exists to previous.
44. Reviews the technology management tasks regularly.

The above 44 metrics were considered in developing a prototype and it is mapped for each student while assessing the project.

The following metrics are not considered due to its non-effectiveness based on the results presented in the chapter 7.

Software Project Planning	
1.	Identify the number of modules schedule for the project.
2.	In case of group projects, all the individuals are agree to their work schedules
Software Design	
3.	Industry accepted standards are used in designing the web pages (like W3C standards).
4.	The design practice (such as simple designs, using design patterns, using abstractions) adopted by the student is easy to change.
Software Programming Practices	
5.	Selective in the choice of data type to ensure the size of a variable is not excessively large.
6.	Variables and routines are used for one and only one purpose.

Acknowledgments

The Investigators would like to express their gratitude and appreciation to King Abdul Aziz City for Science and Technology (KACST) for providing the research grant for the project number 35-226.

References

[1] Abdullah Saleh Al Sadaawi (2010), Saudi National Assessment of Educational Progress (SNAEP), International Journal of Education Policy and Leadership, December 13, 2010. Volume 5, Number 11.

[2] Bavota, G., De Lucia, A., Marcus, A., & Recommending Oliveto, R. (2014). Refactoring Operations in Large Software Recommendation Systems in Systems Software Engineering, Springer, Chapter 15.

[3] Calvin Selig, Sallie Henry. (1998). A design tool used to quantitatively evaluate student projects, Proceedings of the nineteenth SIGCSE technical symposium on Computer Science Education, Pages 124-128, USA.

[4] Fahad Al Faadel, Mohammed Alawairdhi, Mahran Al Zyoud (2012), Success and Failure of IT Projects: A study in Saudi Arabia, Proceedings of the 11th WSEAS international conference on Applied Computer and Applied Computational Science, Pages 77-82.

[5] Ghaleb Hamad Alnahdi (2014), Educational Change In Saudi Arabia, Salman bin Abdulaziz University, Saudi Arabia, Journal of International Education Research – First Quarter, Volume 10, Number 1.

[6] Gunnar Schröder, Maik Thiele, Wolfgang Lehner. (2011). Setting Goals and Choosing Metrics for Recommender System Evaluations, Joint proceedings of the RecSys 2011 Workshop on Human Decision Making in Recommender Systems (Decisions@RecSys'11) and User-Centric Evaluation of Recommender Systems and Their Interfaces-2 (UCERSTI 2) affiliated with the 5th ACM Conference on Recommender Systems, 2011.

[7] Iman Avazpour, Teerat Pitakrat, Lars Grunske and John Grundy. (2014). Dimensions and Metrics for Evaluating Recommendation Systems, Recommendation Systems in Software Engineering, Springer, 2014.

[8] Infosys White Paper. (2013). "Realizing efficiency and effectiveness in software testing through a comprehensive metrics model".

[9] Jayabalaraja, T Edwin Prabakaran. (2012). "Study on software process metrics using data mining tool – A rough set theory approach", International Journal of Computer Applications, Volume 47, No 18.

[10] Marko Gasparic. (2007). Metrics based recommendation system for software engineering, Masters Thesis, University of Maribor.

Saravanan obtained his PhD in Computer Science in the Department of Computer Science and Engineering, Bharathiar University during 2004. He specialized on automated and unified data mining using intelligent agents. His research area includes data warehousing, data mining and software agents. In his credit he has 40+ publications in international journals indexed with ISI, Scopus etc., He has presented many research papers in National, International conferences and also guiding many researchers leading to their PhD degree. He has totally 18+ years' experience in teaching including 3 years as researcher in Bharathiar University. As the credit toward funded projects, he is working with Majmaah University and King Abdulaziz City for Science and Technology, Kingdom of Saudi Arabia with the total project cost of USD $41500. He is a Senior Member of IEEE, life member of Computer Society of India, Indian Society for Technical Education, and Indian Association of Research in Computing Sciences. He is working as Associate Professor in the Dept. of Computer Science, College of Computer and Information Science, Majmaah University, Kingdom of Saudi Arabia.

Abdullah Al Hussein obtained his PhD in Software Engineering from De Montfort University, United Kingdom in 2011. He specialized in software quality. He has 10+ years of experience in software project management at both corporate and project level, including such successes as, centralizing and managing all payroll operations, improving of IT department in Riyadh Public Transport Center (RPTC). He is instrumental in framing academic policies and procedures for the establishment of college of computer and information sciences of Majmaah University, KSA. AT present, he is working as Assistant Professor in the Dept. of Computer Science, College of Computer and Information Science, Majmaah University, Kingdom of Saudi Arabia.

Manjunatha Siddappa Lakshmana obtained his PhD in Nuclear Physics from University of Mysore, India during 2000. He is specialized in environmental radiation physics, nuclear electronics and data analysis. He has 17 years of experience in research and project management. He is advising research students for their PhD. He is working as Assistant Professor in the Dept. of Computer Science, College of Computer and Information Science, Majmaah University, Kingdom of Saudi Arabia.