International Journal of Engineering and Advanced Technology (IJEAT)
ISSN: 2249-8958 (Online), Volume-2 Issue-4, April 2013

Identifying Performance Criterion of Software Projects That Leads To Increasing Project Productivity and Software Quality (A Pilot Study)

Mohamed El Zeweidy, Mohamed Mounir

***Abstract*— The identified major problems in software production centers are those related to nonstandard management system. There are much repeated works, rapid changes of requirements, lack of training, latency of software (S/W) delivery, and large number of defects detected after the delivery of software. This article aims to identify the performance criterion of software projects that leads to increasing project productivity, and software quality.**

The results shows that improving the performance of software productivity can be achieved by applying the Dynamic Forms, Queries, and Reports techniques through the criteria Application type, Functional size measurement approach, Project size, Language type, Team size, Development platform. While improving the performance of Software quality value (SQV) is achieved by applying other criteria such as the use of the six forms of testing collectively (Unit, New function, and Performance, Regression, System, and Acceptance tests) and full documentation.

***Index Terms*— software benchmark, software quality, software production centers, software productivity.**

I. INTRODUCTION

Software projects are a risky, complex, and costly process, the complexity of the task means that it is difficult to predict development effort, quality and schedules, where a fee is being charged for the development of software, the impact on the business of poor estimates of software development effort, quality and schedules.

II. OBJECTIVE

The research has been conducted to achieve the following objectives:

A. To clearly identify the customer complaints.
B. To select the case study which includes the software project specifications.
C. To study the factors that cause problems through implementing one project according to specifications and standards mentioned.

To achieve the previous goals, apilot studyhas been implemented to determine the software project specifications, then the empirical study has been applied on 37 software projects which come from 4 software production centers (including 2 organizational and 2 private centers)

III. METHODOLOGY

We started our research by specifying the appropriate criteria to evaluate and specifythe software projects suitable for our study. The following sections will clarify the selection criteria, and the type of projects/platforms used in our study.

3.1 Selection Criteria

To select the appropriate criteria to evaluate and specify software projects, the following steps were performed:

1) Study of the projects repository published in 2011 by International Software Benchmarking Standards Group (ISBSG).
2) Picked 5,052 Software Projects that have been carried out in the last ten years, over 24 foreign countries and these projects have the a broad range of project types from many industries and many business areas.
3) Study the software quality value collected from historical data from more than 600 corporations and more than 30 government organizations, more than 13,000 software projects have been reviewed by Software Productivity research (SPR).

From the studies mentioned above, we selected two groups selection criteria out of 8 for Software Projects Specifications. The first group objective (which includes 6 selection criteria) was to evaluate the project delivery rate (PDR, number of hours/total number of function points). The second group selection criteria objective(including 2 selectioncriteria) was to evaluate the software quality. We will discuss the different selection criteria in more details in the following sections.

3.1.1 First Group Criteria

The first group criteria includes the following 6 criteria(Peter R. Hill, 2011):

1) Application type,
2) Functional size measurement approach,
3) Project size,
4) Language type,
5) Team size,
6) Development platform.

Manuscript published on 30 April 2013.
***** Correspondence Author (s)
Dr. Mohamed El Zeweidy, El-Shourouk Academy Higher Institute of Computer & Information Technology
Eng. Mohamed Mounir , Arab Academy for Science and Technology

a) Application Type Criteria

Almost 64% of application types are transaction process, accounting, or management information system (MIS) applications as shown in figure (1-1).

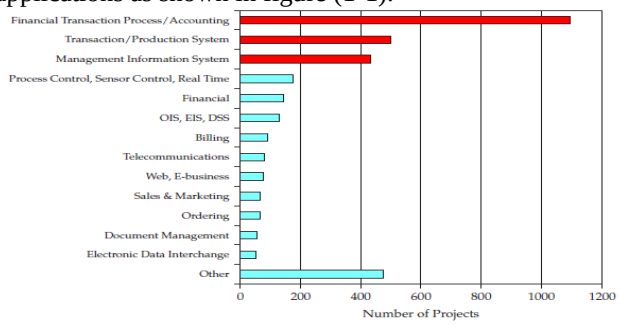

Figure (1-1) 64% of Application Types are Transaction Process /Accounting/MIS

b) Functional Size Measurement Criteria

Table (1-1) shows that among the different functional size measurement(FSM) approaches (Total metrics, 2007), ISBSG repository which elected 5,052 Software Projects includes 3379 projects that apply International Function Point Users Group (IFPUG) approach with 67%.

Table (1-1) IFPUG projects dominate the ISBSG repository (Peter R. Hill, 2011)

	FSM Approaches Type	Project	%
1	International Function Point Users Group (IFPUG)	3379	67
2	Functional Size Measurement Method (FiSMA)	478	9.46
3	Common Software Measurement International Consortium (COSMIC)	335	6.63
4	Netherlands Software Metrics Association (NESMA)	130	2.57
5	Others (ex: "LOC" lines of code or not used)	730	14.44

According to the application type criteria mentioned above in figure (1-1) and table (1-1) we selected the IFPUG because it is dominant and more suitable to these application types which support 'data rich' (Total metrics, 2009).

c) Project Size Criteria

Figure (1-2) shows the pareto chart of the IFPUG function point distribution of the projects.

From the figure, we can see that very small project Size (1108 projects) and Small size (740 projects) have the greatest percentage value (54.7%) of the 3379 projects.

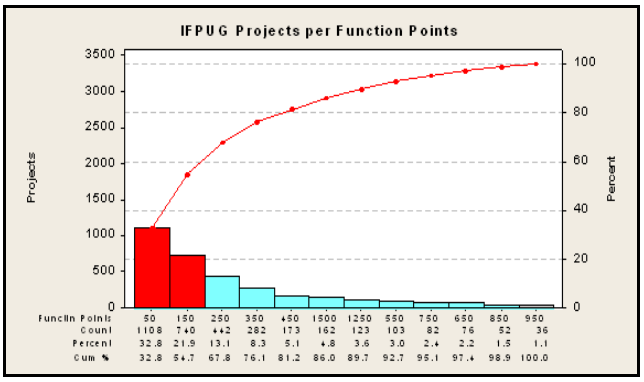

Figure (1-2) IFPUG Projects per Function Points

d) Programming Language Type Criteria

Table (1-2) shows that the 4th Generation Languages (4GLs) as a whole have significantly best Project Delivery Rate (PDR) (Peter R. Hill, 2011).The conclusion is that 4GLs is dominant.

Table (1-2) 4GLs have significantly better PDR than 3GLs

Generation Languages	No of projects	Mean (h/FP)	Standard Deviation (h/FP)
2nd	12	17.9	13.2
3	1105	6.7	15.1
4	359	11.3	8.8

e) Team Size Criteria

Maximum team size is known to be one of the most important factors that affects PDR. According to Table (1-3), once a team size exceeds five people, the productivity decreases which implies to high project delivery rate.

Table (1-3) Team size affects PDR (Peter R. Hill, 2011)

Team Size	No of projects	Mean (h/FP)	Standard Deviation (h/FP)
1 to 4	157	11.0	11.3
5 to 8	212	13.8	12.4
9 or more	215	20.0	15.2

f) Development Platform Criteria

Table (1-4) shows that the PCs have the best productivity, lowest project delivery rate values (mean and standard deviation).

Table (1-4) PCs have better productivity (Peter R. Hill, 2011)

Development Platform Type	No of projects	Mean(h/FP)	Standard Deviation (h/FP)
Mainframe	452	18.7	15.6
Midrange	128	15.6	13.3
PC	204	10.7	10.0

The result after using the 6 previous criteria had shown that there are variations in software project delivery rate data result withMean (10.7) and Standard deviation (10.0).

3.1.2 Second Group Criteria

The objectiveof the second group criteria was to evaluate the software quality. It includes 2 selection criteria as we will see.

Table (1-5) illustrate the SQV criteria according to the project size. The two measured criteria are project defect removal, and defects detected after delivery. The table shown illustrate the variations according to the project size. For the small project sizes (about 100 FP) the project defects removal efficiency results are varied from 94% to 99%, and the total number of project defects detected after delivery are varied from 2 to 21 defects.

Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)

Table (1-5) SQV Criteria According to Project Size (CAPERS JONES, 2010)

No	Criteria	100 FP	1000 FP	10,000 FP
1	Project defects removal efficiency (%)	94 to 99%	93 to 97%	84 to 96%
2	Project defects detected after delivery	2 to 21 defects	75 to 315 defects	1.400 to 9.600 defects
9	Project defects detected after delivery		21 defects at most	

IV. EXPERIMENTAL RESULTS

In this section we will compare our case study results against the standard international ones, and introduce the enhancement figures to improve our results grasp the standard international.

Our case study were performed on 37 S/W projects in 4 production centers in Egypt. We have identified the most important selectioncriteriafor the selected projects and it is summarized in table (1-6).

During the study we conducted meetings with the S/W houses directors and applied the first 6 mentioned selection criteria for the selected 37 projects. A summary of the characteristics of the chosen projects are as follows:

- Projects types are Transaction Process, Accounting or Management Information System
- On PCs Platform.
- The language type is 4th generation languages.

Meanwhile, we conducted a review with the internal customersto discover their complaints which were summarized in the following points:

- There are much repeated works.
- Rapid changes of external customer requirements.
- Lack of training.

Another review were conducted with the external customers for the same reason, and their complaints with the team work within 3 months after S/W delivery were as follows:

- Latency of software delivery.
- Large number of defects detected after the delivery of software

After investigating the project defects detected by the end of the software development life cycle phases, the results shows that the total number of defects detected after delivery were 28 defects, while Defects Removal Efficiency was (72%) (Defects detected before delivery by total number of defects detected).

Tables (1-7), and (1-8) demonstrate the difference in delivery rate and defects removal efficiency between the 37 software projects used in our case study and the values reported by (ISBSG, 2011) and by (CAPERS JONES, 2010).

Table (1-6) Proposal in the case study according to two groups of criteria

No.	Two groups of Criteria		Proposal in the case study
1	Group-1	Application Type	Transaction Process / Accounting / MIS
2		Functional Size Measurement Approach	IFPUG
3		Product Size Approach	up to 100 Function Points
4		Language Type	4th Generation Language.
5		Team Size	1-5
6		Development Platform	PC
7		Software Project Delivery Rate	Mean (10.7) h/FP.
8	Gro	Project defects removal efficiency	94% at least

Table (1-7) The difference of delivery rate between the available software projects(37 Software Projects) results and the values reported by (ISBSG, 2011)

Criteria	Results of the available S/W projects (37 projects)	The values reported by (ISBSG, 2011)
Project delivery Rate (hour per function point)	11.33 h/FP	10.7 h/FP (at most)

Table (1-8) The difference of defects removal efficiency between the results of 37 software projects and the values reported by (CAPERS JONES, 2010)

Criteria	Results of the available S/W projects (37 projects)	Value reported by (CAPERS JONES, 2010)
Defects Removal Efficiency	72 %	94 % at least
Defects detected after delivery	28 efects	21 defects at most

4.1 Improving Project Delivery Rate :

During our case study, we perceived that the S/W projects were using the standard static types of software projects forms, queries, and reports.Static forms, queries, and reports are defined as follows:

1) Static Forms:

Forms are created and its contents are known at design time.

2) Static Queries:

Query statements include parameters whose value are known at design time.

3) Static Reports:

Reports are created and its contents are known at design time.

There are three techniques used to improve the productivity delivery rate namely; dynamic forms, dynamic query, and dynamic reports that are defined as follows:

4) Dynamic Forms

Forms are created at runtime, each time a dynamic form is run, dynamic forms supports programmers in creating a single, dynamic, scrollable form using a form description language. The virtual form is structured into sections and subsections so that effective organization and navigation of the information are possible. Dynamic Forms is based on an understanding of the environment in which data-entry tasks are carried out, the manner in which users perform their tasks, and how the users utilize human and computer resources in solving these tasks. (Andreas Girgensohn, et al, 1995).

Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)

1) Dynamic Query

Dynamic Query statements include parameters whose values are not known until runtime. Which are very common for user-driven interactive applications where the user provides most of the information required to construct the exact SQL. (Neeraj Sharma, et al, 2010).

Characteristic	Stock Control Project		International values (mean)
	Using Static Forms	Using Dynamic Forms	
Project Delivery Rate (Hours per function points)	11.33	10.4	10.7 at most

2) Dynamic Reports

Reports are created at runtime. Each time a dynamic report is run, it gathers the most recent data. (Neil FitzGerald, et al, 2007).

To improve the productivity delivery rate in our case study, we have applied the dynamic form technique on one software project (Stock Control Project).

Using dynamic forms implied improvement in project delivery rate. Its value was reduced from 11.33 h/FP to 10.4 h/FP and reached the standard international values as shown in table (1-9).

Table (1-9) Improving Stock Control PDR after using Dynamic Forms (Own)

Additional improvement had been achievedUsing dynamic Queries in addition to dynamic Forms implied more improvement in project delivery rate. Its value was reduced from 10.7 h/FP to 9.7 h/FP. as shown in table (1-10).

Table (1-10) Improving Stock Control PDR after using Dynamic Forms and queries (Own)

Characteristic	Stock Control Project		International values (mean)
	Using Dynamic Forms	Using Dynamic Forms and Queries	
Project Delivery Rate (Hours per function points)	10.4	9.7	10.7 at most

4.2 Improving Software Quality Value :

The two measures that express the Software Quality Value**(SQV)** are the defect removal efficiency (during the test phase of the software development life cycle) and the defects detected after delivery.

The measuredvalues in our case study for defect removal efficiency were 72%and 28for the defects detected after deliverywhile the standard internationals values were 94% (at least) and between 2 to 21defects (at most).

Figure (1-3) use Dotplot(Stephan Lunau, et al, 2008) to illustrate the defects removal efficiency distribution of the 37 software projects used in our case study, we found that there are 10 best values software projects, these software projects already appliedproductivity techniques to improve PDR (dynamic forms and/or queries), while theother 27 software projects didn't apply these improvements techniques.

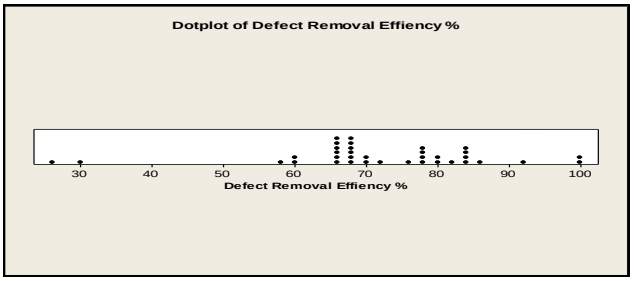

Figure (1-3) Shows the defects removal efficiency distribution (Own)

The results of the 10 best values, as shown in Table (1-11) reveals that there is an improvement in defect removal efficiencyas it became 84% (compared to 72% of all software projects available) and another improvement in defects detected after delivery as it became 6 defects (compared to 28 of all software projects available) .

Table (1-11) Analyzing Software Quality Value of 37 software projects (Own)

No. of software projects	Apply productivity techniques	Defects detected after delivery	Defect Removal Efficiency
27	No	36	67.49%
10 (Best)	yes	6	83.68%
37 (all)	all	28	71.78%
Target	yes	21 at most	94% at least

Although the 10 best software projects mean value for defects detected after delivery were 6 defects (reached the standard international value 'Target'), but its values related to defect removal efficiency became 84%still not reaching the standard international value(94-99%).

According to (CAPERS JONES, 2010), the six forms of testing collectively (unit test, new function test, regression test, system test and acceptance test) are needed to achieve high levels of reliability and customer satisfaction , these tests are defined(William E. Perry, 2006) as follows:

1) Unit test :

Is a method by which individual units of source code are tested to determine if they are fit for use.

2) New function test :

Tests that are based on the project specifications and related to new functions that may be uncovered in unit tests.

3) Performance test :

Tests that emphasis on the final performance characteristics.

4) Regression test :

is any type of software testing that seeks to uncover new errors, or regressions, in existing functionality after changes have been made to a system, such as functional enhancements, patches or configuration changes.

5) System test :

Is the test conducted on a complete, integrated system to evaluate the system's compliance with its specified requirements?

Published By:
Blue Eyes Intelligence Engineering and Sciences Publication (BEIESP)

6) Acceptance test :

Is conducted to determine if the requirements of a specification or contract are met. It should involve users.

After investigating the 10 best software project processes and documents, the results shows the percentage of six forms of testing processes and documents, while the target (CAPERS JONES, 2010) is to perform and document all test processes of these six forms of testing, table (1-12) demonstrate these gap, and shows the defects that should be covered to reach the standard.

Table (1-12) The difference of 6 phases tests and documentations between the results of 10 (best) software projects and the values reported by (CAPERS JONES, 2010)

V.CONCLUSION

In this article we introduced a case study for the Egyptian S/W production centers and investigated the major problems found during the S/W life cycle and after S/W deliveryto evaluate the project delivery rate and the software quality expressed in the defect removal efficiency and the defects detected after delivery.

We have seen that applying the dynamic form technique instead of the static one in one project (stock control) improvedtheproject delivery rate. This was sufficient for this project, while other projects that have large number of queries or reports that are similar in design should apply dynamic queries and reports to improve software productivity expressed as delivery rate, and software quality expressed asdefects removal efficiency and defects detected after delivery.

We have shown that using the six forms of testing collectively (unit, new function, performance, regression, system, and acceptance tests) and full documentation has improved the defects removal efficiency from 84% to 94%.

Application of one of the modern quality methodologiessuch as Six Sigma, Quality Assurance (QA), Lean or Capability Maturity Model Integration (CMMI) is important and will lead to much more improvements in software productivity and quality as well as sustaining these improvements and reach the international standards.

REFERENCES

[1] Andreas Girgensohn, et al, 1995. Dynamic Forms: An Enhanced Interaction Abstraction Based on Forms, Chapman & Hall, London. 362-367.
[2] CAPERS JONES, 2010 Software Engineering Best Practices, New York: McGraw Hill.
[3] Christine B. Tayntor, 2007. SIX SIGMA SOFTWARE DEVELOPMENT, 2nd ed. New York: CRC press Auerbach publication.
[4] Daniele Chenal and Paul Schwartz, 2010.Improve Your Software Development Lifecycle Process, - Practical Tips and Guidelines. QAVantage.
[5] David Garmus, 2011. Certified Function Point Specialist Examination Guide. New York: CRC press Auerbach publication.
[6] Mary Bradley, et al, 1999. Function Point Counting Practices Manual 4.1. USA: International Function Point Users Group.
[7] Mei He, et al, 2010. Understanding the Influential Factors to Development Effort in Chinese Software Industry. Institute of Software, Chinese Academy of Sciences.
[8] Neeraj Sharma, et al, 2010. Database Fundamentals 1st Ed.Canda: IBM Corporation, 168-178.
[9] Neil FitzGerald, et al, 2007. Crystal Reports XI Official Guide. USA:Sams Publishing, 133-153.
[10] Peter R. Hill, 2011, Practical Software Project Estimation. New York: McGraw Hill.
[11] Stephan Lunau, et al, 2008. Six Sigma+Lean Executing Improvement Projects Successfully Toolset , Germany: Springer.
[12] Total metrics, 2007. How to Decide which Method to Use,
[13] Total metrics, 2009. COSMIC and IFPUG Similarities and Differences.
[14] William E. Perry, 2006. Effective Methods for Software Testing 3rd Ed. Canada: Wiley Publishing.