Volume 5, Issue 3, March 2015 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Various Productivity Measures of Software Development Teams

Ajeta Nandal
Dept of CSE, CR Polytechnic,
Rohtak, Haryana, India

Abstract: Analysing and improving productivity has been one of the main goals of software engineering research since its beginnings. A plethora of studies has been conducted on various factors that resulted in several models for analysis and prediction of productivity. However, productivity is still an issue in current software development and not all factors and their relationships are known. This paper reviews the various productivity measuring techniques of software development teams. It was concluded through the literature survey that software productivity measurement can be done using SLOC (Source Lines of Code), function points, use case points, object points, and feature points. Various other features like the team size, response time, task complexity, team climate and team cohesion also have an impact on software development team productivity.. This paper reviews the large body of available literature in order to distill a list of the main factors influencing productivity investigated so far.

Keywords: Software Productivity, Team Productivity, Productivity Factors, Software Teams.

I. INTRODUCTION

The objective of IT deaprtment for any organization is to achieve operational efficiency, reduce costs on repetitive tasks, reduce response time of the customer, achieve consistency, reliability and accuracy in customer transactions so that customer satisfaction can be improved. In achieving this, the organization forms project teams, cross functional teams to meet the organizational objectives. More than 70% of the Fortune 500 organizations have teams in their organizations. Particularly software development is done by teams in organizations. Onsite, offshore teams, virtual teams, globally distributed teams, high performance teams, and self managed teams are some of the terminology we hear in software industry in current days. The objective of any software business organization is to achieve maximum team productivity to reduce costs and to increase profitability. With the advent of process maturity models such as Capability Maturity Model Integration and People Capability Maturity Model, software services organizations are even trying for continuous improvement. The definition of productivity can be given as.

II. SOFTWARE PRODUCTIVITY

Productivity can be defined as the rate of output per unit of input. In a classical manufacturing environment, this measure is taken straightforward: you can measure the effect of using labor, materials or equipment. The output can be measured as a number of products you deliver. In software engineering too, it could be useful to compare the output to the input. Input is the amount of effort we spend on the project to deliver the software. But for the amount of output, there is no straightforward measurement. The problem is that there is no definition for what exactly is being produced with a software project. One could see it as a physical amount of product, i.e. an amount of lines of code that are produced. However, one could also see the software product as a delivery of functionality or even express it in terms of the quality attributes it meets. All these measurements try to quantify the *size* of the delivered good, namely the software product. Therefore, we express the software productivity as one of size over effort.

Productivity = Size/Effort

According to Card (2006), Productivity is defined as the ratio of outputs produced to the resources consumed. Earlier researchers like Albrecht (1979) have developed Function Points at IBM and Jones (1986) has studied the productivity and quality of the software projects. Jones (1986) work on productivity is published in his popular book Programming Productivity. Researcher Lakhanpal (1993) has studied the characteristics of groups and their impact on productivity (Wagner & Ruhe, 2008a). Study of software development team productivity involves disciplines such as Software Engineering, Management and Organizational Psychology. Banker, Datar and Femerer (1991) have studied the variables impacting the productivity of software maintenance projects with the help of an empirical study of 65 software maintenance projects of a large commercial bank.

III. WHY MEASURE TEAM PRODUCTIVITY?

According to Scacchi (1995), Software team productivity is to be measured to reduce the software development costs, to improve the quality of deliverables, and to increase the rate at which software is to be developed. According to him, the software productivity is to be measured to recognize the top performers to reward and identify the bottom performers to provide the training.

The major productivity improvements can result into substantial amount of savings in development costs (Scacchi, 1995). Measuring productivity helps in identifying underutilized resources (Nwelih & Amadin,2008). The study of software productivity is important because higher productivity leads to lower costs (Bouchaib & Charboneau, 2005).

Bouchaib and Charboneau (2005) have studied the comparison of productivity of in-house developed projects and productivity of outsourced projects to third party with a sample of 1085 projects developed worldwide.

Krishnan, Kriebel, Kekre and Mukhopadhyay (1999) have studied the software life cycle productivity, which includes both development and maintenance costs and drivers of software team productivity and quality such as personnel capability, product size, usage of tools and software process factors.

According to Banker and Kauffman (1991), software productivity can be found from the following formula.

Productivity = (Size of Application Developed) / (Labor consumed during development)

IV. MEASURING TEAM PRODUCTIVITY

According to Wagner and Ruhe (2008), software productivity can be measured traditionally using the lines of code or function points and the productivity is the LOC or FP produced per hour by the programmer. The productivity and cost estimation model COCOMO developed by Boehm (1981) also considers the individual programmer's productivity as a decisive role. The factors identified by Barry Boehm and team which affect software productivity and cost include programmer capability, team cohesion, platform experience, programmer's programming languages and tools experience, software applications experience, and analyst capability.

According to Tausworthe(1982), Team Productivity (P)

P = Kilo Lines of Code/ Person months of effort

Needed Staff size = Person months of effort / Project time duration in months

According to Banker, Datar and Kemerer (1991), Measurement model considers Function Points, SLOC, environmental variables, and any deviations from the project.

Analysis/Design Activity Output measure = Function Points

Coding/Testing Activity Output measure = Source Lines of Code

Input Measure = Total Labor hours

According to Tockey (1996), Productivity Model and Cost Model explains the impact of interaction of team members and team size on team productivity and project cost.

According to Krishnan, Kriebel, Kekre and Mukhopadhyay (1999), Model of Life Cycle Productivity and Quality considers variables such as personnel capability, quality, software process, product size in LOC, Front End Resources and Usage of tools.

Quality = f1(Personnel Capability, Usage of Tools, Product Size in LOC, PROCESS, Front End Resources)

Life Cycle Productivity = f2(Conformance Quality, Personnel Capability, Usage of Tools, PROCESS)

Life Cycle Productivity = Product size in LOC / Total cost incurred in Product development and support.

According to Potok and Vouk (1999), Model of Correlated Team Behavior provides a simulation model which supports correlated team behavior.

Software Team productivity = KLOC per Calendar month.

According to Nogueira, Luqi, Berzins and Nada (2000), Productive Ratio (α) model suggested considered productivity, requirements volatility and complexity.

α = % of Direct Development time / % of Idle time

According to Blackburn, Lapre and Van Wassenhove (2002), Productivity Model considers the factors such as Experience of Project Manager, size, requirements ambiguity, complexity, stable standards, user requirements, usage of tools, etc.

Productivity = Number of Function Points / Effort in Man months

According to Card (2006), Simple Model of Productivity considers the entities such as Product, Process or Sub-process, Requirements, Value, Cost, and Effort.

Physical Productivity = Number of LOC / man hours or days or months

Functional Productivity = Number of Function Points / Man hours or days or months

Economic Productivity = Value / Cost

Where Value = *f* (Price, Time, Quality, Functionality)

According to Jiang and Comstock (2007), Normalized Productivity Delivery Rate (PDR) were defined. It uses two continuous variables Average Team Size and PDR and six categorical variables like Language Type, Development Type, Development Platform, Development Techniques, Case Tool Used, and How Methodology Acquired.

log (PDR) = 2.8400+0.3659 x log(TeamSize) - 0.6872 x I(3GL) - 1.2962 x I(4GL) – 1.3225 x I(ApG) – 0.1627 x I(MR) – 0.4189 x I(Multi) – 0.3201 x I (PC) – 0.4280 x I(OO) – 0.2812 x I(Event) + 0.7513 x I(OO:Event) – 0.2588 x I(Business) – 0.0805 x I(Regression) + 1.0506 x I(Business:Regression)

Normalized PDR = (Normalized Work Effort) / (Adjusted Function Points)

According to Nwelih and Amadin (2008) , Software Productivity model includes Software Reuse complexity, length, functionality and effort.

Productivity = $\Sigma (r_i + f_i + l_i + c_i) / \Sigma_I$ Where r_i = Reuse , f_i = Functionality, l_i = Length, c_i = Complexity, Σ_I = Effort

V. OTHER PRODUCTIVITY MEASUREMENTS

Traditionally SLOC and Function points are used as units of software team productivity. Other measures such as use case points, object points and feature points are also used in some of the IT organizations. These new measures kept in mind the object orientation, extendibility and reusability in finding the software team productivity.

VI. CONCLUSION

The definition of productivity and why to measure software productivity have been explained. The productivity measures such as SLOC, KLOC and Function points are discussed. The models and techniques related to software development team productivity are also concluded. Further research can be done on finding the soft factors affecting the software development team productivity. The productivity measured can be improved. An organization working on the factors affecting the software development team's productivity can improve the overall organizational productivity. Organizational productivity is dependent on individual and team productivity. Thus improving software development team's productivity results into improved organizational productivity. The productivity improvement is part of team development. Hence, one should try to increase the productivity of software development teams resulting into the better organizational productivity and performance.

REFERENCES

[1] Banker, R. D., Datar, S. M., & Kemerer, C. F. (1991) A Model to Evaluate Variables Impacting the Productivity of Software Maintenance Projects. Management Science, 37(1), January 1991.

[2] Banker, R. D., & Kauffman, R. J. (1991) Reuse and Productivity in Integrated Computer-Aided Software Engineering: An Empirical Study. MIS Quarterly, September 1991, 14(3):374-401.

[3] Blackburn, J. D., Lapre, M. A., & Van Wassenhove, L. N. (2002) Brooks' Law Revisited: Improving Software Productivity by Managing Complexity. Vanderbilt University Working paper 2002-85.

[4] Boehm, B. (1981). Software Engineering Economics. Englewood Cliffs, New Jersey, USA: Prentice Hall.

[5] Bouchaib, B., & Charboneau, R. (2005) An Evaluation of Productivity Measurements of Outsourced Software Development Projects: An Empirical Study. Proceedings of the 2nd Software Measurement European Forum, SMEF 2005, March 16-18, 2005, Rome, Italy.

[6] Card, D. N. (2006). The Challenge of Productivity Measurement. Proceeding of Pacific Northwest Software Quality Conference, Portland, OR, 2006.

[7] Jiang, Z., & Comstock, C. (2007). The Factors Significant to Software Development Productivity. World Academy of Science, Engineering and Technology, Volume 25, January 2007.

[8] Krishnan, M.S., Kriebel, C.H., Kekre, S., & Mukhopadhyay, T. (2000) An Empirical Analysis of Productivity and Quality in Software Products. Management Science, 46(6):745-759.

[9] Nwelih, E., & Amadin, I.F. (2008) Modeling Software Reuse in Traditional Productivity Model. Asian Journal of Information Technology, 7(11):484-488.

[10] Scacchi, W. (1995). Understanding Software Productivity. Article in software Engineering and Knowledge Engineering: Trends for the Next Decade edited by D.Hurley, Vol. 4, World Scientific Press,1995.

[11] Tausworthe, R. C. (1982). Staffing Implications of Software Productivity Models. TDA Progress Report 42-72, October-December 1982.

[12] Tockey, S. (1996). The Effect of Team Size on Team Productivity and Project Cost. Lecture notes of Software Project Management, CSSE-515, Seattle University, Winter 2000.

[13] Wagner, S. & Ruhe, M. (2008). A Structured Review of Productivity Factors in Software Development. Technical Report, Technische Universitat Munchen, 2008.

