www.sjm06.com

Serbian Journal of Management 7 (1) (2012) 65 - 75

Serbian Journal of Management

MEASURING PRODUCTIVITY OF SOFTWARE DEVELOPMENT TEAMS

Goparaju Purna Sudhakar[a*], Ayesha Farooq[b] and Sanghamitra Patnaik[c]

[a]*Engineering Staff College of India, Gachibowli, Hyderabad, A.P., India, 500034*
[b]*Aligarh Muslim University, Aligarh, U.P., India*
[c]*Advanced Centre for American Studies (ACAS), Osmania University Centre for International Programmes (OUCIP), Osmania University, Hyderabad, A.P., India*

(Received 20 November 2010; accepted 26 September 2011)

Abstract

This paper gives an exhaustive literature review of the techniques and models available to measure the productivity of software development teams. Definition of productivity, measuring individual programmer's productivity, and measuring software development team productivity are discussed. Based on the literature review it was found that software productivity measurement can be done using SLOC (Source Lines of Code), function points, use case points, object points, and feature points. Secondary research findings indicate that the team size, response time, task complexity, team climate and team cohesion have an impact on software development team productivity. List of factors affecting the software development team productivity are studied and reviewed.

Keywords: Software Productivity, Team Productivity, Productivity Factors, Software Teams, Managing Information Systems Projects

1. INTRODUCTION

The objective of IT function 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,

* *Corresponding author: purna24@hotmail.com*

DOI: 10.5937/sjm1201065S

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 CMMI and PCMM, software services organizations in Asian countries are even thriving for continuous improvement. The definition of productivity follows.

2. DEFINITION OF PRODUCTIVITY

Traditionally Productivity can be defined as a ratio of output units produced to the input units of effort (Scacchi, 1995; Maxwell, 2001; Wagner & Ruhe, 2008a; Nwelih & Amadin, 2008). Output units can be the number of lines of source code and input units can be the person months of time. Traditionally the lines of code (LOC) or the Function Points are used for measurement of productivity in software development (Wagner & Ruhe, 2008a). According to Wagner and Ruhe (2008a), the number of lines of code written or the number of function points implemented per man hour by the developer is used as a measure of productivity. Previous software team productivity studies were conducted in organizations such as IBM, NASA, ITT, and HP. According to Nwelih and Amadin (2008), software productivity definition includes complexities of both software and people. According to them software productivity can be calculated by dividing software size with cost of development.

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.

3. 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 under utilized 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) (1)

4. MEASURING INDIVIDUAL PROGRAMMER'S 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.

Setting goals to the programmer, providing training, giving periodic feedback on his or her performance improves the individual productivity of programmer (Wagner & Ruhe, 2008). According to Chiang and Mookerjee (2004), improving software development productivity depends on people, technology and process.

The constraints which control the programmer productivity are the time constraints, financial constraints, software specifications, corporate environment and programming methodology (Vyhmeister, 1996). According to Vyhmeister (1996), one can use Lines of code (LOC), Function Points (FPs), and Object points (OPs) to measure the productivity of software programmer.

Individual programmer's productivity can vary between 1 to 10 times in the same experience level programmers and team productivity can have variations of a factor of 5 (White, 1999). People related issues are the critical factors of individual programmer's productivity.

According to M. Pinkowska, team productivity is dependent on individual team member's productivity and the team member's experience of success impacts his or her motivation, team cohesiveness and work atmosphere. Teams with high cohesiveness exhibit lower tension and anxiety, less variations of productivity, improved team member satisfaction, improved team communication, and commitment. Team members in cohesive teams enjoy team membership, experience low personnel turnover, and they are very productive (M. Pinkowska's Research).

Table 1. Techniques/Models for measuring Software Development Team Productivity

Sl. No:	Technique/Model	Formula/Description	High Lights	Reference
1.	Team Productivity (P)	P = Kilo Lines of Code/ Person months of effort	Further given the needed staff size as "person months of effort divided by project time duration in months"	Tausworthe (1982)
2.	Measurement model	Analysis/Design Activity Output measure = Function Points Coding/Testing Activity Output measure = Source Lines of Code Input Measure = Total Labor hours	This model considers Function Points, SLOC, environmental variables, and any deviations from the project.	Banker, Datar and Kemerer (1991)
3.	Productivity Model and Cost Model	Mathematical Models	This Model explains the impact of interaction of team members and team size on team productivity and project cost.	Tockey (1996)
4.	Model of Life Cycle Productivity and Quality	Quality = f_1(Personnel Capability, Usage of Tools, Product Size in LOC, PROCESS, Front End Resources) Life Cycle Productivity = f_2(Conformance Quality, Personnel Capability, Usage of Tools, PROCESS) Life Cycle Productivity = Product size in LOC / Total cost incurred in Product development and support.	This model considers variables such as personnel capability, quality, software process, product size in LOC, Front End Resources and Usage of tools.	Krishnan, Kriebel, Kekre and Mukhopadhyay (1999)
5.	Model of Correlated Team Behavior	Software Team productivity = KLOC per Calendar month.	Provides a simulation model which supports correlated team behavior.	Potok and Vouk (1999)
6.	Productive Ratio (α)	α = % of Direct Development time / % of Idle time	The model suggested considered productivity, requirements volatility and complexity.	Nogueira, Luqi, Berzins and Nada (2000)
7.	Productivity Model	Productivity = Number of Function Points / Effort in Man months	This model considers the factors such as Experience of Project Manager, size, requirements ambiguity, complexity, stable standards, user requirements, usage of tools, etc.	Blackburn, Lapre and Van Wassenhove (2002)

8.	Simple Model of Productivity	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)	This model considers the entities such as Product, Process or Sub-process, Requirements, Value, Cost, and Effort.	Card (2006)
9.	Normalized Productivity Delivery Rate (PDR)	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)	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.	Jiang and Comstock (2007)
10.	Software Productivity model which includes Software Reuse	Productivity = n $\sum (r_i + f_i + l_i + c_i) / \sum_l$ i=1 Where r_i = Reuse f_i = Functionality l_i = Length c_i = Complexity $\sum_l$ = Effort	This model considers factors like reuse, complexity, length, functionality and effort.	Nwelih and Amadin (2008)

(Source: Author Compiled, 2010)

According to Blackburn, Lapre and Wassenhove (2002), the measure of productivity using number of function points divided by man months of effort is applicable irrespective of the programming language in which the project is being implemented.

According to Card (2006), the productivity of large teams is lower than the productivity of small teams. According to him, while measuring productivity labor related to engineering, management, testing and support needs to be taken into consideration.

According to the research done by Banker, Datar and Femerer (1987), high project quality need not necessarily reduce the software maintenance team productivity. The teams that behave in correlated fashion perform better than the teams that behave randomly (Potok & Vouk, 1999). According to Potok and Vouk (1999), weak team's productivity is less than the productivity of teams that behave randomly and the better way to handle the weak team performance is to shuffle the team members randomly. Software team performance is dependent on the human characteristics of the team (Potok & Vouk, 1999).

Currently Function points and Lines of Code are measures of software productivity (Maxwell & Forselius, 2000).

5. FACTORS AFFECTING THE PRODUCTIVITY

Major changes to technologies of the project product lead to minimal productivity improvements (Scacchi, 1995). According to Scacchi (1995), some of the factors facilitate high software team productivity include substantial computing infrastructure, software engineering tools and techniques such as rapid prototyping tools, software testing tools, e-mail, document management systems, object oriented programming languages, and configuration management systems. The attributes such as well organized project teams, experienced programmers, and different team work structures facilitate the high software team productivity.

The product, process and production setting characteristics affect the software productivity of individual programmer as well as software development team (Scacchi, 1995). User participation, experience of the programmer with the programming language and program design constraints are the factors affecting the software productivity (Wagner & Ruhe, 2008a). According to Wagner and Ruhe (2008a), soft factors such as team culture, team identity, team cohesion, support for innovation, turnover, and team communication affect the software team productivity.

Average team size, programming language used (3GL or 4GL), development platform and development techniques have impact on software team productivity (Jiang & Comstock, 2007). Team size has impact on team productivity and project costs (Tockey, 1996).

According to Potok and Vouk (1999), strong teams have high productivity over all the assigned tasks where as weak teams have low productivity over all assigned tasks. It was also proven by Vijayashree and Jagdischchandra (2011) and by Kuye and Sulaimon (2011). Software reuse can be used to improve the software team productivity (Nwelih & Amadin, 2008). Large team size reduces the software development productivity (Blackburn, Lapre & Wassenhove, 2002). According to Blackburn, Lapre and Wassenhove (2002), project complexity increases the team size and team size decreases the team productivity significantly. They have studied 117 software projects provided by the Software Technology Transfer Finland. According to Banker and Kauffman (1991), experience of the programmer has impact on the productivity of software maintenance projects.

According to the research done by Blackburn, Lapre and Wassenhove (2002), experience of the project manager and project size increases the team productivity.

According to them, change in user requirements and usage of tools significantly decreases the software development team productivity. Banker, Datar and Femerer (1991) have examined the effects of project team member ability, application experience, system quality, hardware and methodological tools on software maintenance team productivity.

According to Banker, Datar and Femerer (1987), the environmental variables affecting the software team productivity are project management, personnel, user and technical environment. According to them personnel variables critically impact the productivity of software project teams. According to Banker and Kauffman (1991), reuse is the major factor affecting the software development productivity.

According to Agrell and Gustafson (1994), team climate consists of vision, participative safety, team orientation, and support for excellence have impact on team productivity. They have used Team Climate Inventory (TCI) developed by the Anderson and West for finding the team climate and productivity of Swedish work-groups. Wagner and Ruhe (2008) have derived some soft factors which affect the software development team's productivity. They are team identity, personnel turnover, team cohesion, team communication, clear goals and support for innovation. Support for innovation as a factor affecting the team productivity has been proved by the study of Agrell and Gustafson (1994) on Swedish work teams.

Premraj, Shepperd, Kitchenham and Forselius (2005) have identified the factors such as task difficulty, interaction with customer, skills of project team, and non functional requirements such as performance and dependability impact the software team productivity. According to Premraj, Shepperd, Kitchenham and Forselius (2005) research, productivity varies from company to company and business sector to business sector. Within the same company different business sectors can have different productivity levels and productivity is also dependent on year and hardware.

The factors affecting the software development team productivity are the system construction time and coordination efforts (Chiang & Mookerjee, 2004). This is because if the system construction time is less, team size increases and if team size increases productivity per programmer reduces. Organizational structure, internal politics, organization size, team morale, and physical facilities have impact on the software team productivity (Vyhmeister, 1996). According to Jiang, Naude and Comstock (2007), software team productivity variations are because of average team size and the unbalanced usage of programming languages (3GLs or 4GLs).

Product characteristics, people, process, and technology have impact in the software development time and product outcome (White, 1999). People factors affecting the software development team productivity are the staffing, motivation and work environment. According the M. Pinkowska's research, team cohesiveness has impact on software team productivity.

According to Comstock, Jiang and Naude (2007), fourth generation programming languages give more productivity than third generation programming languages. According to Teasley, Covi, Krishnan, and Olson (2000), teams in warrooms give double the productivity than the normal teams. According to their research team collocation increases the software team productivity.

6. 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.

7. 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 tabulated. The factors affecting the software development team productivity are explained. Other productivity measures such as Use Case points, object points, feature points are mentioned. 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

Abdel-Hamid, T. K., & Madnick, S. E. (1989) Lessons Learned from Modeling the Dynamics of Software Development, Communications of the ACM, 32(12), December 1989.

Agrell, A., & Gustafson, R. (1994) The Team Climate Inventory (TCI) and group innovation: a psychometric test on a Swedish sample of work groups. Journal of Occupational and Organizational Psychology, 67:143-151.

Albrecht, A.J. (1979) Measuring Development Productivity. Proceedings of the Joint SHARE/GUIDE/IBM Application Development Symposium, pp. 83-92.

Banker, R. D., Datar, S. M., & Kemerer, C. F. (1987) Factors Affecting Software Maintenance Productivity: An Exporatory Study. Proceedings of the International Conference on Information Systems, December 1987.

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.

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.

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.

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

Bouchaib, B., & Charboneau, R. (2005) An Evaluation of Productivity

МЕРЕЊЕ ПРОИЗВОДНОСТИ ТИМОВА ЗА РАЗВОЈ СОФТВЕРА

Goparaju Purna Sudhakar[a*], Ayesha Farooq[b], Sanghamitra Patnaik[c]

[a]Engineering Staff College of India, Gachibowli, Hyderabad, A.P., India, 500034
[b]Aligarh Muslim University, Aligarh, U.P., India
[c]Advanced Centre for American Studies (ACAS), Osmania University Centre for International Programmes (OUCIP), Osmania University, Hyderabad, A.P., India

Извод

Овај рад даје исцрпни литературни преглед техника и модела доступних за мерење продуктивности тимова за развој софтвера. Дискутовани су дефиниција продуктивности, мерење индивидуалне продуктивности програмера и мерење продуктивности тимова за развој софтвера. Засновано на литературном прегледу утврђено је да мерење продуктивности тимова за развој софтвера може да се учини употребом SLOC (Source Lines of Code), функционалних тачака, употребом кључних тачака, објектних тачака, и особинских тачака. Секундарни резултати истраживања индицирају да су величина тима, време одговора, комплексност задатака, клима у тиму као и кохезија тима од великог значаја на продуктивност. Проучена је листа фактора који утичу на продуктивност тимова за развој софтвера.

Кључне речи: Продуктивност софтвера, Продуктивност тимова, Фактори продуктивности, Тимови за развој софтвера, Пројекти за менаџмент информатичких тимова

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.

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

Chiang, I. R., & Mookerjee, V. S. (2004) Improving Software Team Productivity. Communications of the ACM, 47(5), May 2004.

Comstock, C., Jiang, Z., & Naude, P. (2007) Strategic Software Development: Productivity Comparisons of General Development Programs. World Academy of Science, Engineering and Technology, Volume 34, 2007.

Fredrick, P.B. (1995). The Mythical Man Month: Essays on Software Engineering Second Edition, Addison-Wesley. July 1995.

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

Jiang, Z., Naude, P., & Comstock, C. (2007). The Variation of Software Development Productivity 1995-2005. World Academy of Science, Engineering and Technology, Volume 27, 2007.

Jones, C. (1986). Programming Productivity, McGraw-Hill. January 1986.

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.

Kuye, O. L. & Sulaimon, A. H. A.(2011) Employee involvement in decision making and firms performance in the manufacturing sector in Nigeria. Serbian Journal of Management, 6 (1): 1-15.

Lakhanpal, B. (1993) Understanding the Factors influencing the Performance of Software development groups: An exploratory group level analysis. Information and Software Technology, 35(8): 168-173.

Little, T. (2004). Value Creation and Capture: A model of the Software Development Process. IEEE Software, May/June 2004.

Maxwell, K. D. (2001). Collecting Data for Comparability: Benchmarking software Development Productivity. IEEE Software, September/October 2001.

Maxwell, K. D., & Forselius, P. (2000). Benchmarking Software Development Productivity. IEEE Software, January/February 2000.

Nogueira, J. C. , Luqi, V. Berzins & Nada, N. (2000). A Formal Risk Assessment Model for Software Evolution. Paper Presented at Second International Workshop on Economics-Driven Software Engineering (EDSER-2), Limerick, Ireland, June 6, 2000.

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

Pinkowska, M. (Undated). IT Software Project Management: Impact of Team Cohesiveness on Productivity and Performance. Available online at http://www.lent.ch/pdf_diverse/Pinkowska-cohesiveness.pdf (Accessed on 02-Mar-2010).

Potok, T. E., & Vouk, M. A. (1999). A Model of Correlated Team Behavior in a Software Development Environment. Proceedings of IEEE Symposium on Application-specific Systems and Software Engineering and Technology, ASSET' 99, 1999, Richardson, TX.

Premraj, R., Kitchenham, B., Shepperd, M. & Forselius, P. (2005). An Empirical Analysis of Software Productivity over Time. 11th IEEE International Symposium on Software Metrics, 2005.

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.

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

Teasley, S., Covi, L., Krishnan, M. S., & Olson, J. S. (2000). How does Radical Collocation Help a Team Succeed?. Proceedings of the 2000 ACM Conference on Computer Supported Cooperative Work, Philadelphia, Pennsylvania, USA, pp. 339-346.

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.

Vijayashree, L., & Jagdischchandra, V. (2011). Locus of control and job satisfaction: PSU employees. Serbian Journal of Management, 6(2): 193-203.

Vyhmeister, R. (1996). Programmer Productivity. Available online at http://www.andrews.edu/~vyhmeisr/papers/progprod.html (Accessed on 02-Mar-2010).

Wagner, S. & Ruhe, M. (2008). A Structured Review of Productivity Factors in Software Development. Technical Report, Technische Universität München, 2008.

Wagner, S., & Ruhe, M. (2008a). A Systematic Review of Productivity Factors in Software Development. Proceedings of 2nd International Workshop on Software Productivity Analysis and Cost Estimation (SPACE 2008), State Key Laboratory of Computer Science, Institute of Software, 2008.

White, K. S. (1999). Software Engineering Management for Productivity and Quality. International Conference on Accelerator and Large Experimental Physics Control Systems, 1999, Trieste, Italy.