Software Engineering Productivity: Concepts, Issues and Challenges

Adrián Hernández-López, Ricardo Colomo-Palacios,
Ángel García-Crespo, Fernando Cabezas Isla

{adrian.hernandez, ricardo.colomo, angel.garcia, fernando.cabezas}@uc3m.es

Universidad Carlos III, Madrid, Spain

Adrián Hernández-López
Computer Science Department,
Universidad Carlos III de Madrid
Av. Universidad, 30.
28911 Leganés (Madrid). Spain
Phone: +34 91 624 9417
Fax: +34 91 624 9129
adrian.hernandez@uc3m.es

Adrián Hernández López is a PhD student at the Computer Science Department of the Universidad Carlos III de Madrid in Spain. His research interests include applied research in People in IT, Human Aspects in IT and Software Process Improvement. He finished his Bachelor's degree in Computer Science on 2007 and his Master's degree in IT Science specialized on Software Engineering on 2009 at the Universidad Carlos III de Madrid. He has been working as software engineer in several companies including Telefónica and INDRA.

Ricardo Colomo-Palacios
Universidad Carlos III de Madrid
Computer Science Department,
Av. Universidad, 30.
28911 Leganés (Madrid). Spain
Tel: +34 91 624 5958
Fax: +34 91 624 9129
ricardo.colomo@uc3m.es

Ricardo Colomo-Palacios is an Associate Professor at the Computer Science Department of the Universidad Carlos III de Madrid. His research interests include applied research in Information Systems, Software Project Management, People in Software Projects and Social and Semantic Web. He received his PhD in Computer Science from

the Universidad Politécnica of Madrid (2005). He also holds a MBA from the Instituto de Empresa (2002). He has been working as software engineer, project manager and software engineering consultant in several companies including Spanish IT leader INDRA. He is also an Editorial Board Member and Associate Editor for several international journals and conferences and Editor in Chief of International Journal of Human Capital and Information Technology Professionals.

Ángel García-Crespo
Universidad Carlos III de Madrid
Computer Science Department,
Av. Universidad, 30
28911 Leganés (Madrid). Spain
Tel: +34 91 624 9417
Fax: +34 91 624 9129
angel.garcia@uc3m.es

Angel García-Crespo is the Head of the SofLab Group at the Computer Science Department in the Universidad Carlos III de Madrid and the Head of the Institute for promotion of Innovation Pedro Juan de Lastanosa. He holds a PhD in Industrial Engineering from the Universidad Politécnica de Madrid (Award from the Instituto J.A. Artigas to the best thesis) and received an Executive MBA from the Instituto de Empresa. Professor García-Crespo has led and actively contributed to large European Projects of the FP V and VI, and also in many business cooperations. He is the author of more than a hundred publications in conferences, journals and books, both Spanish and international.

Fernando Cabezas-Isla
Computer Science Department,
Universidad Carlos III de Madrid
Av. Universidad, 30.
28911 Leganés (Madrid). Spain
Phone: +34 91 624 9417
Fax: +34 91 624 9129
fernando.cabezas@uc3m.es

Fernando Cabezas-Isla is a PhD student at the Computer Science Department of the Universidad Carlos III de Madrid in Spain. His research interests include software engineering, human factors in technology and semantic technologies. He finished his Bachelor’s degree in Computer Science on 2009 at the Universidad Carlos III de Madrid.

Software Engineering Productivity: Concepts, Issues and Challenges

Adrián Hernández-López, Universidad Carlos III, Madrid, Spain

Ricardo Colomo-Palacios*, Universidad Carlos III, Madrid, Spain

Ángel García-Crespo, Universidad Carlos III, Madrid, Spain

Fernando Cabezas Isla, Universidad Carlos III, Madrid, Spain

ABSTRACT

Software engineering productivity has been widely treated in past research literature, but there are many issues that remain unsolved. Interesting works related to new metrics and more replications of past productivity analysis have emerged; however, in order to fulfill these unsolved issues, a consensus about influencing factors and well recognized and useful sets of inputs and outputs for using in measurements may be reached. Moreover, a clear state of the art may shed light on further research in software engineering productivity, which remains a promising research area. In this paper, general concepts of software engineering productivity along with general issues and recent challenges that need further attention from the research community are presented.

KEYWORDS: Productivity; Software Engineering; Management; Process metrics; Product metrics

INTRODUCTION

Productivity management continues to prove a challenge for IT projects. While the manufacturing industries have designed and tested methods for determining productivity, IT industry lags behind in terms of methods for evaluating the outcomes and predicting the effort required to complete projects (Dalcher, 2006). The importance of productivity management in this sector is due to be a key factor for knowledge activities (Ramirez & Nembhard, 2004) and a crucial aspect for decision making in the global market (Ross & Ernstberger, 2006). There are two main areas of production in the IT industry that can coexist in an IT project: software and hardware. Focusing on IT software project, primary types of projects are considered: new development and maintenance, in which productivity measurement seems similar but need to be analyzed separately (Premraj, Shepperd, Kitchenham, & Forselius, 2005).

On the one hand, productivity metrics in IT software projects are mainly based on ratios between the size of delivered software and the effort performed to obtain it (Arnold & Pedross, 1998; MacCormack, Kemerer, Cusumano, & Crandall, 2003). Following this direction, Function Points (FP) and Source Lines Of Code (SLOC) are misleading measurements because the overall development may be unproductive even

though programming productivity has increased (Lee & Schmidt, 1997). Furthermore, there are some recent proposals such as measuring productivity based on postmortem Function Points (Asmild, Paradi, & Kulkarni, 2006), using multiple size measures (Kitchenham & Mendes, 2004), and specific methods for methodologies such as Object Oriented (Pendharkar, 2006). On the other hand, the factors that influence IT Productivity are widely accepted, i.e. increasing store constraints, timing constraints, reliability requirements, high level languages, team size, requirements volatility, staff tools skills, staff availability, customer participation, and project duration (see for example (Foulds, Quaddus, & West, 2007; Maxwell & Forselius, 2000)). Despite of its acceptance, the influence, positive or negative, of some of these factors in IT Productivity is not clear and may vary due to external factors such as the business sector (Maxwell & Forselius, 2000; Premraj, Twala, Mair, & Forselius, 2004) and outsourcing ratio (Tsunoda, Monden, Yadohisa, Kikuchi, & Matsumoto, 2009).

Moreover, metrics based on code only represent the productivity of coding activities, while other activities such as management, analysis, and design stay out of those metrics scope. Therefore, the relation between productivity and non coding activities in software development projects has not been properly addressed. Finally, the use of code measures such as SLOC is not sufficient by itself; multidimensional inputs must be included in measuring productivity (Mahmood, Pettingell, & Shaskevich, 1996).

In this scenario, the definition of new models, measures and assessment methods may elucidate productivity in IT projects, especially in software development projects as it has been done for software maintenance projects (Banker, Datar, & Kemerer, 1991). In turn, companies need to establish their own productivity metrics in addition to benchmarking their data (Maxwell & Forselius, 2000). Furthermore, the construction of a state of the art about IT productivity regarding software development may result in a solid contribution to be considered by future researches in this area.

In order to shed some light on this area, we present the state of the art of IT projects performance from different standpoints. Firstly, we introduce the general concepts regarding productivity and software engineering productivity. Secondly, we analyze issues that influence productivity measurement in this industry sector. Thirdly, we present the challenges that have been proposed in the research literature and some new ones that we propose. Finally, we finish this article with some conclusions.

CONCEPTS

First of all, it is important to introduce the main concept of productivity. The origins of the term “productivity” traced back to the eighteen century, and was introduced by Quesnay (1766); however, until the middle of the past century, the definitions were blurred. Traditionally, productivity has been defined as the ratio of outputs produced per unit of input (Jefferys, Hausberger, & Lindblad, 1954). This definition fits well in manufacturing paradigms because is based on quantities of standardized and clearly identified units of measurement, but it does not fit in new environments such as service industries, or the software industry, where there are also intangible assets along with the tan-

gible ones. Moreover, as Grönroos and Ojasalo (2004) concluded, the notion of productivity in services as the combined effect of a service provider effectiveness manages the profitability of its resources and production processes (internal efficiency), and the perceived quality of its services (external efficiency) makes productivity a very different concept compared to the traditional concept of manufacturing productivity.

Productivity should be viewed as a component of performance, not a synonym for it (Sink, Tuttle, & DeVries, 1984). This claim is argued from the concept of comparative productivity performance and not as a result unit, namely productivity measures should be useful for comparison over time, while performance represents a timely measure. In this direction, the value of productivity measurement lies in the capability to manage and monitor, in order to reach a more efficient resources use (Fitzgerald & Moon, 1996). In addition, as Nachum (1999) argued, the main objective of productivity measurement is productivity enhancement. Moreover, productivity improvements must be reflected in ROI improvements. Therefore, productivity is inversely proportional to the costs incurred (Anselmo & Ledgard, 2003).

As Anselmo and Ledgard (2003) pointed following Lord Kelvin's affirmation[1], software productivity enhancement cannot be expected without measuring productivity. An appropriate productivity measure provides a prognostic tool as to how to achieve productivity amelioration (Nachum, 1999). Furthermore, following Gummesson (1992) recommendation, before any attempt to measure productivity in the service industry, identification of what is to be captured is required. Thus, considering these contributions, in order to create a software engineering productivity measurement, distinguishment of factors, inputs and outputs susceptible to be measured is required. The roadmap for this process may follow the flow chart of Figure 1 (Sahay, 1997). In this roadmap, there are some steps that could be carried out with qualitative research methodologies such as interviews and group techniques like Nominal Group Technique (NGT) or Delphi. Also, qualitative approaches may be applied for all the steps.

[1] "When you can measure what you are speaking about, … you know something about it; but when you cannot measure it, … your knowledge is of a meager and unsatisfactory kind… —Lord Kelvin

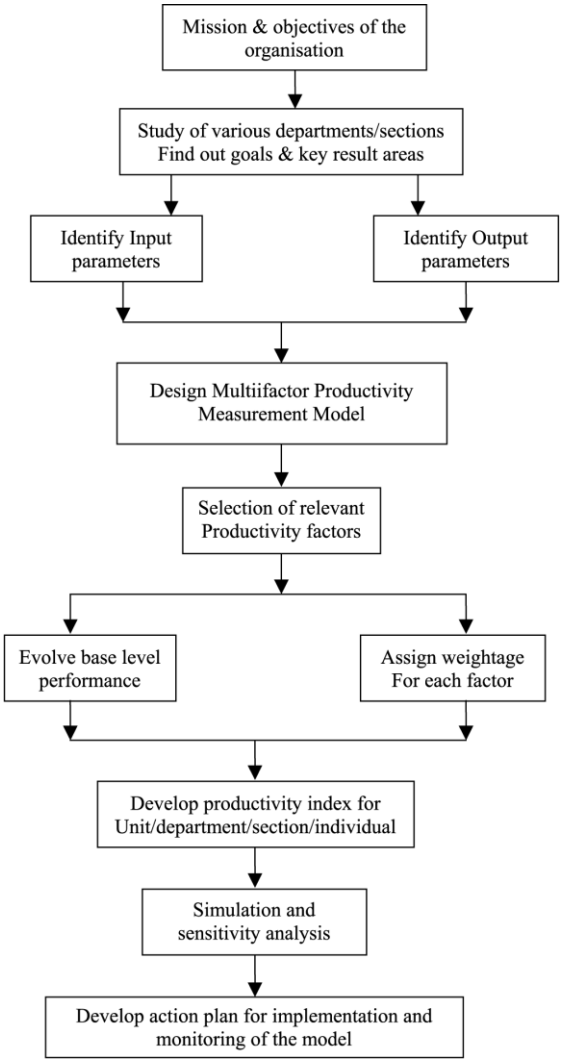

Figure 1. Flow chart of methodology adopted for productivity measurement in a service organization (Sahay, 1997)

In addition, the level of analysis should be taken in consideration when measuring software engineering productivity: sector, organization, department, project, unit and individual. Not every measure will be useful for all the levels. There should be a clear specification of the measurement purpose and the intended audiences for measurement data, which may be defined prior to any other step (Sink et al., 1984). At a sector and organization level, the involved factors, inputs and outputs granularity is not enough to for a precise measurement that could lead to improvement; the measurements at that levels constitute just an item for inter period comparison. Lower level measurements require a wide analysis of all the items that may be measured, which requires an established process such as the one created by Sink et al. (1984). At this point, we question whether current practices of productivity measurement in software engineering follow these types of processes; and, whether the input and output parameters are indentified for all measurement levels. Moreover, are quality and other well accepted factors such as reuse considered in software engineering productivity measurement?

ISSUES

Before considering new challenges in software engineering productivity measurement, it is necessary to set some general requirements that guide the formulation of a productivity concept. Following the six general requirements established by Adam, Johanson, and Gravesen (1995), software engineering productivity measurement demands of the presence of the following facts:

- Software engineering output has to be seen as a value for the customer and from the customer's perspective.
- Software engineering outputs must be defined by their quality level.
- The customer must become a part of the productivity concept.
- Productivity measures must be more customer-related.
- Dynamic indicators of productivity must be used instead of static output/input measures.
- Specific situation measures have to be available to allow for the complexity and diversity of software engineering operations.

In addition, identification of the factors that influence productivity measurement is required. Anselmo and Ledgard (2003) summed up some of these factors: independence of the modules, understandability of the code and architecture, flexibility of software development process, visibility of architecture, and abstraction of the software production process in order to be examined experimentally. Also, the importance of architecture and design in software development productivity is well adressed (Anselmo & Ledgard, 2003; Tan et al., 2009). Therefore, the understandability and independence of modules produced influence software productivity in a huge manner. These are inherent properties of a software development environment, and may be increased or decreased by design. Other factors that influence productivity are resource constraints, requirement volatility, use of software tools, programs complexity, programming languages, customer and user involvement, personnel experience and capabilities, staffing stability, team size, outsourcing ratio, etc (Maxwell & Forselius, 2000; Premraj et al., 2005; Tsunoda et al., 2009). Scacchi (1994) divided these factors into three attribute lists: software development environment, software system product, and project staff attributes.

From our viewpoint, personnel factors require a special attention (i.e. Colomo-Palacios, Tovar-Caro, García-Crespo, & Gómez-Berbís, 2010). Software engineering activities are capital intense, so the human factor has to be analyzed in any management practice order to obtain a more adequate result. In the context of productivity measurement, it is well accepted that factors related to personnel such as (technical, non-technical) capabilities and skills, and (programming language, project, process…) experience influence directly on productivity results. In addition to these factors, and considering the lack of literature related to this area, we propose that other factors such as motivation, performance management practices, compensation and rewards systems, organizational climate, and happiness could influence productivity results; but it is not

clear how they influence and how to introduce them in productivity measurement. Thus, a wide range of research possibilities presents through the combination of knowledge of human resources management and productivity management, which could lead to a transfer of cognition for a common research purpose (Koskinen, 2008).

In software engineering, the quality of resulting outputs and inputs may affect productivity; therefore, quality and productivity should be measured as interrelated concepts (Grönroos & Ojasalo, 2004). As Gummesson (1995) observed, quality, productivity and profitability form a triplet, all parts of which are related to the same phenomenon: the economic result of the organization. The quality of a product may be significantly and positively affected by personnel capability, software development process factors, and the deployment of resources in initial stages of product development (especially design) (Krishnan, Kriebel, Kekre, & Mukhopadhyay, 2000).

The software engineering industry to a large extent is an open system where stakeholders, clients and the end users influence inputs and outputs, which produces a contribution to both the internal and external efficiency, and therefore the productivity of the production process (see for example Hsu, Chen, Jiang, & Klein, 2010). Hence, a totally different approach to productivity has to be undertaken in order to obtain a global measure that establishes how well a software engineering organization uses resources to create outputs with acceptable perceived quality and customer value (Grönroos & Ojasalo, 2004). Thus, inputs and outputs measurement should consider both quantity and quality. This importance is reflected in the premises that Grönroos and Ojasalo established: "The better the perceived quality that is produced using a given amount of inputs (service provider's inputs and customers' inputs), the better the external efficiency is, resulting in improved service productivity" and "The more efficiently the service organization uses its own resources as input into the processes and the better the organization can educate and guide customers to give process-supporting inputs to produce a given amount of output, the better the internal" (Grönroos & Ojasalo, 2004).

The frontiers between inputs and outputs are fuzzy (Gupta, 1995); for instance, knowledge used in software production processes may be considered both an input and an output, and suffer a transformation during these processes. For example, in maintenance phases, all products from the software production process are used as inputs and are transformed to adapt to client needs, or to solve encountered bugs. A deep analysis of inputs should be taken into consideration because the use of input proxies introduces biases for productivity measurement. This source of error is one of the most difficult to overcome, and the possible methods of dealing with it are not at all obvious (Rees, 1980). Therefore, this issue is important because more time and money is spent supporting product enhancements and error corrections than in development (Anselmo & Ledgard, 2003).

Moreover, it is difficult to define "a unit of software engineering". So there are some universal measures for input and output measurement (Boehm, Abts, & Chulani, 2000). From the input standpoint, some productivity measures uses man hours as an effort

placeholder, staff capabilities as a correction factor for effort, technical and non-technical issues. From the output viewpoint, some productivity measures use source SLOC as a quantity output, error ratio as a correction factor, function points completed, and products finished. These parameters are physical measures, but they could be transformed in a monetary unit in order to obtain a financial productivity measure. However, one should keep in mind that there are problems with financial measures that have to be observed. For instance, revenues are not always a good output measure, since price does not always reflect the perceived service quality (Grönroos & Ojasalo, 2004).

Once the inputs, outputs, and factors influencing productivity measurement are defined, a formulation of the measure can be established. Hence, specific metric can be defined and each organization may use one or several of them for measuring its productivity. In order to sum up the state of the art about inputs, outputs and metrics, the most used in each category are presented in Table 1. They are ordered according to the measurement difficulty, from easier to harder. The degree of representation scale of the production process itself is also represented: lower difficulty measures are less representative of the production process than harder measures. From our standpoint, it is surprising that despite of the controversial validity of some inputs and outputs for software productivity measurement such as the derivate from Man Month and SLOC, they remain in use in nowadays organizations and new research publications, due to its practical utilization as Boehm argued a while ago (1987). Moreover, two of the most widely used inputs, SLOC and FP, tend to be highly correlated (Banker et al., 1991; Laranjeira, 1990).

Inputs	**Outputs**	**Metric (P=Productivity)**
Wages	Sales	P = Sales / Wages
Effort = Men Hours	TLOC = SLOC + DLOC	P = TLOC / Effort
Effort = Men Hours	Function or Feature Points (includes all the variations of the original ideal)	P = FP / Effort
Effort = Man Month	Delivered Source Instructions (DSI)	P = DSI / Effort (i.e. Gaffney, 1989)
Any	Any	Data Envelopment Analysis (DEA)(i.e. Mahmood et al., 1996)
Any	Any	Multifactor metrics (i.e. Kitchenham & Mendes, 2004)
Any	Any	General Linear Model

		metrics

Table 1. Existing inputs and outputs used to calculate productivity

Finally, productivity measurement is related with management practices, and more specifically with organizational culture and structure. In organizations with a culture of reporting and benchmarking, productivity measurement could be considered as a useful enhancement indicator by which, productivity could be assessed at all organization levels. Also, in organizations with a directive management style where the orders come from top to bottom, productivity measurement will be accepted as a way to communicate to upper level managers how well projects are evolving. On the other hand, in organizations with a culture of innovation and creativeness, productivity measurement could be viewed as a monitoring tool that does not lead to goals achievement. Moreover, the link between organization's goals and productivity could be unclear in such organizations, and this may lead to a misunderstanding of the what, how and why to assess productivity measurement.

CHALLENGES

As pointed out by Boehm (2006), software engineering projects are growing larger in size and become more and more complicated in order to fulfill our requirements in every possible way, whilst delivery requires faster turnaround time. Hence, efficient and efficiency indicators are needed. In this line, productivity measurement represents an indicator of how efficient is the production process carried at a specific level of analysis and in an organization. Existing metrics are useful in specific situations but researchers appear to have no idea when to stop the empirical validation, even in the case of well-investigated metrics, and there also is no reflect on whether the methodology is appropriate (Kitchenham, 2010).

One specific challenge that should be covered is the influence of reuse in software engineering productivity metrics. It is unclear how it could be linked to productivity measurements due to reasons such as the difficulty that makes abstraction and generalization when pulling a module from one system and place it in another (Anselmo & Ledgard, 2003). Also, the unsolved link between reuse and some tasks of software engineering such as requirement engineering does not enable the selection of commercial off-the-shelf (COTS) software (Maiden & Ncube, 1998), which influences the production process productivity.

Another challenge that remains unsolved is the creation of a metric that could be applied in both new development and maintenance projects, considering the differences. However, from a more general standpoint, metrics are required to quantify productivity along each phase of software engineering production process. These issues are important because not all of the outputs in software engineering activities are SLOC or FP or convertible to them, and also, not all of the inputs are wages and effort. Thus, metrics to measure other non-traditional productivity factors are demanded. In addition, an

analysis of the dynamic productivity rate represents a future research aim (Heričko & Živkovič, 2008).

Furthermore, the research about productivity measurement in software engineering is mainly focused on productivity at project level or higher levels (organization, sector, industry…), and there is a gap in the literature about productivity measurement in lower levels such as team and individual. For example, there are some studies that reflect the importance of design in overall software production processes (i.e. Anselmo & Ledgard, 2003) but it remains unclear how to measure the productivity of software designers and even the software design task. This challenge could be extrapolated to other software engineering areas such as management or quality assurance where the outputs remain unmeasured and unvalued in the productivity context.

Finally, we consider that software productivity measurement is a learning activity and therefore, history information related to factors and measures is required. Thence, in order to learn and keep improving software engineering processes, organizations may continuously record and accumulate diverse metrics of their projects (Tsunoda et al., 2009). However, establishing this record process is not enough; organizations should achieve a balance in the investment of recording the required data and its future in order to accomplish improved goals. In this direction, there have been some national and international research organizations responsible of the creation of specific projects for establishing data banks of productivity measures along with many measurement factors, but generally these projects have ended fading away. Therefore, the creation and promotion of new data banks, mainly international, will enable a solid start point to further research in this important area.

CONCLUSIONS

Along this paper the concept of productivity in the software engineering area have been presented. One of the general conclusions extracted after reviewing the related literature is that despite the productivity analyses performed in many different countries (Blackburn, Scudder, & Wassenhove, 1996; Maxwell & Forselius, 2000; Maxwell, Wassenhove, & Dutta, 1996; Premraj et al., 2005) there is a lack of studies in other countries and regions, for instance South America, or Mediterranean countries of the European Union. This gap has to be covered because software development environment and culture are different in each country, so different results from a myriad of datasets might be obtained (Tsunoda et al., 2009). In addition, replication studies with different data sets are considered extremely important (Andersson & Runeson, 2007; Cusumano, MacCormack, Kemerer, & Crandall, 2003; Mendes & Lokan, 2008), due to the existence of so many unclear factors in software development. These studies increase the reliability of the results because of its prominent reflection in the real world.

Moreover, as presented in previous sections there are many issues and problems that remain unsolved and require further attention from research community. From our standpoint, the next generation of software engineering productivity measures will need to consider the human capital factor capital due to the presence of highly intellectual

capital intense activities in production processes. Furthermore, new measures will need to consider quality because it's strong connection with productivity in service industries. Therefore, this factor cannot be omitted from productivity measurements; and as Krishnan et al. (2000) concluded, further research is required to define objective metrics for other software quality dimensions and study their effects on productivity.

REFERENCES

Adam, K., Johanson, M., & Gravesen, I. (1995). *Service productivity: a vision or a search for a new outlook.* Paper presented at the 9th World Productivity Congress, Istanbul.

Andersson, C., & Runeson, P. (2007). A Replicated Quantitative Analysis of Fault Distributions in Complex Software Systems. *IEEE Transactions on Software Engineering, 33*(5), 273-286.

Anselmo, D., & Ledgard, H. (2003). Measuring productivity in the software industry. *Communications of the ACM, 46*(11), 121-125.

Arnold, M., & Pedross, P. (1998). *Software size measurement and productivity rating in a large-scale software development department*. Paper presented at the Proceedings of the 20th international conference on Software engineering.

Asmild, M., Paradi, J. C., & Kulkarni, A. (2006). Using data envelopment analysis in software development productivity measurement. *Software Process Improvement and Practice, 11*(6), 561-572.

Banker, R. D., Datar, S. M., & Kemerer, C. F. (1991). A model for evaluate variable impacting the productivity of software maintenance projects. *Management Science, 37*(1), 1-18.

Blackburn, J. D., Scudder, G. D., & Wassenhove, L. N. V. (1996). Improving Speed and Productivity of Software Development: A Global Survey of Software Developers. *IEEE Transactions on Software Engineering, 22*(12), 875-885.

Boehm, B. W. (1987). Improving Software Productivity. *Computer, 20*(9), 43-57.

Boehm, B. W. (2006). *A view of 20th and 21st century software engineering*. Paper presented at the Proceedings of the 28th international conference on Software engineering.

Boehm, B. W., Abts, C., & Chulani, S. (2000). Software development cost estimation approaches - A survey. *Annals of Software Engineering, 10*(1), 177-205.

Colomo-Palacios, R., Tovar-Caro, E., García-Crespo, Á., & Gómez-Berbís, J. (2010). Identifying Technical Competences of IT Professionals: The Case of Software Engineers. *International Journal of Human Capital and Information Technology Professionals 1*(1), 31-43.

Cusumano, M., MacCormack, A., Kemerer, C. F., & Crandall, B. (2003). Software Development Worldwide: The State of the Practice. *IEEE Softw., 20*(6), 28-34.

Dalcher, D. (2006). Supporting software development: enhancing productivity, management and control. *Software Process: Improvement and Practice, 11*(6), 557-559.

Fitzgerald, L., & Moon, P. (1996). *Performance measurement in service industries: making it work*. London: Cima.

Foulds, L. R., Quaddus, M., & West, M. (2007). *Structural Equation Modelling of Large-scale Information System Application Development Productivity: the Hong Kong Experience*. Paper presented at the 6th IEEE/ACIS International Conference on Computer and Information Science, 2007. ICIS 2007.

Gaffney, J. (1989). Software reuse--key to enhanced productivity: some quantitative models. *Information and Software Technology, 31*(5), 258-267.

Grönroos, C., & Ojasalo, K. (2004). Service productivity: Towards a conceptualization of the transformation of inputs into economic results in services. *Journal of Business Research, 57*(4), 414-423.

Gummesson, E. (1992). Quality dimensions: what to measure in service organizations. In T. A. Swartz, D. E. Bowen & S. W. Brown (Eds.), *Advances in services marketing and management* (pp. 64-78). Greenwich, CT: JAI Press.

Gummesson, E. (1995). Service productivity: a blasphemous approach. In E. Gummesson (Ed.), *Quality, productivity and profitability in service operations (conference papers from the QP&P Research Program 1992–1994)* (pp. 8-22). Stockholm: University of Stockholm, Department of Business Administration.

Gupta, A. (1995). Productivity measurement in service operations: a case study from the health-care environment. *Managing Service Quality, 5*(5), 31-31.

Heričko, M., & Živkovič, A. (2008). The size and effort estimates in iterative development. *Information and Software Technology, 50*(7-8), 772-781.

Hsu, J. S.-C., Chen, H.-G., Jiang, J., & Klein, G. (2010). The Role of User Review on Information System Project Outcomes: A Control Theory Perspective *International Journal of Information Technology Project Management, 1*(1), 1-14.

Jefferys, J., Hausberger, S., & Lindblad, G. (1954). *Productivity in the distributive trade in Europe: wholesale and retail aspects*: Organisation for European Economic Co-operation.

Kitchenham, B. (2010). What's up with software metrics? - A preliminary mapping study. [10.1016/j.jss.2009.06.041]. *Journal of Systems and Software, 83*(1), 37-51.

Kitchenham, B., & Mendes, E. (2004). Software Productivity Measurement Using Multiple Size Measures. *IEEE Transactions on Software Engineering, 30*(12), 1023-1035.

Koskinen, K. U. (2008). Boundary brokering as a promoting factor in competence sharing in a project work context. *International Journal of Project Organisation and Management, 1*(1), 119-132.

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.

Laranjeira, L. A. (1990). Software Size Estimation of Object-Oriented Systems. *IEEE Transactions on Software Engineering, 16*(5), 510-522.

Lee, S., & Schmidt, R. C. (1997). Improving application development productivity in Hong Kong. In B. J.M. & M. B.M. (Eds.), *Information technology and challenge for Hong Kong*. Hong Kong: Hong Kong University Press.

MacCormack, A., Kemerer, C. F., Cusumano, M., & Crandall, B. (2003). Trade-offs between Productivity and Quality in Selecting Software Development Practices. *IEEE Software, 20*(5), 78-85.

Mahmood, M. A., Pettingell, K. J., & Shaskevich, A. I. (1996). Measuring Productivity of Software Projects: A Data Envelopment Analysis Approach. *Decision Sciences, 27*(1), 57-80.

Maiden, N. A., & Ncube, C. (1998). Acquiring COTS Software Selection Requirements. *IEEE Software, 15*(2), 46-56.

Maxwell, K. D., & Forselius, P. (2000). Benchmarking Software-Development Productivity. *IEEE Software, 17*(1), 80-88.

Maxwell, K. D., Wassenhove, L. V., & Dutta, S. (1996). Software Development Productivity of European Space, Military, and Industrial Applications. *IEEE Transactions on Software Engineering, 22*(10), 706-718.

Mendes, E., & Lokan, C. (2008). Replicating studies on cross- vs single-company effort models using the ISBSG Database. *Empirical Software Engineering, 13*(1), 3-37.

Nachum, L. (1999). Measurement of productivity of professional services. *International Journal of Operations and Production Management, 9*(9/10), 922-950.

Pendharkar, P. C. (2006). Scale economies and production function estimation for object-oriented software component and source code documentation size. *European Journal of Operational Research, 172*(3), 1040-1050.

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

Premraj, R., Twala, B., Mair, C., & Forselius, P. (2004). *Productivity of Software Projects by Business Sector: An Empirical Analysis of Trends*. Paper presented at the 10th IEEE International Software Metrics Symposium (Late Break-in Papers).

Quesnay, F. (1766). Analyse de la formule arithmétique du tableau économique de la distribution des dépenses annuelles d'une nation agricole. *Journal de l'Agriculture, du Commerce & des Finances*, 11-41.

Ramirez, Y. W., & Nembhard, D. A. (2004). Measuring knowledge worker productivity: A taxonomy. *Journal of Intellectual Capital, 5*(4), 602-628.

Rees, A. (1980). Improving Productivity Measurement. *The American Economic Review, 70*(2), 340-342.

Ross, A., & Ernstberger, K. (2006). Benchmarking the IT productivity paradox: Recent evidence from the manufacturing sector. *Mathematical and Computer Modelling, 44*(1-2), 30-42.

Sahay, B. S. (1997). *Productivity linked incentive scheme in a large engineering industry in India: a case study*. Paper presented at the Xth World Productivity Congress.

Scacchi, W. (1994). Understanding Software Productivity. In W. D. Hurley (Ed.), *Software Engineering and Knowledge Engineering: Trends for the Next Decade* (Vol. 3, pp. 293-321). Pittsburgh, PA.

Sink, D. S., Tuttle, T. C., & DeVries, S. J. (1984). Productivity measurement and evaluation: what is available? *National Productivity Review, 3*(3), 265-287.

Tan, T., Li, Q., Boehm, B. W., Yang, Y., He, M., & Moazeni, R. (2009). *Productivity trends in incremental and iterative software development*. Paper presented at the Proceedings of the 2009 3rd International Symposium on Empirical Software Engineering and Measurement.

Tsunoda, M., Monden, A., Yadohisa, H., Kikuchi, N., & Matsumoto, K. (2009). Software development productivity of Japanese enterprise applications. *Information Technology and Management, 10*(4), 193-205.