International Series in
Operations Research & Management Science

Ali Emrouznejad
Emilyn Cabanda *Editors*

Managing Service Productivity

Using Frontier Efficiency Methodologies and Multicriteria Decision Making for Improving Service Performance

Springer

International Series in Operations Research & Management Science

Volume 215

For further volumes:
http://www.springer.com/series/6161

Ali Emrouznejad • Emilyn Cabanda
Editors

Managing Service Productivity

Using Frontier Efficiency Methodologies and Multicriteria Decision Making for Improving Service Performance

Springer

Editors
Ali Emrouznejad
Operations and Information
Management Group
Aston Business School
Aston University
Birmingham
United Kingdom

Emilyn Cabanda
School of Business and Leadership
Regent University
Virginia Beach, VA, USA

ISSN 0884-8289
ISSN 2214-7934 (electronic)
ISBN 978-3-662-43436-9
ISBN 978-3-662-43437-6 (eBook)
DOI 10.1007/978-3-662-43437-6
Springer Heidelberg New York Dordrecht London

Library of Congress Control Number: 2014945469

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

To our fathers

Preface

Frontier efficiency methodologies are classified into two types: *deterministic production frontier and stochastic production frontier*. *Frontier* refers to the maximum limit which represents the best-practice approaches to production. *Efficiency* is the use of maximum outputs produced from a given mix of inputs. *Stochastic production frontier* [i.e., Stochastic Frontier Analysis (*SFA*)] allows technical inefficiency effects, can account statistical noise in the measurement of efficiency, and also specifies a functional form for the production (e.g., cost function). *Deterministic production frontier*, such as *data envelopment analysis* (DEA), is a goal programming approach, which assumes that any deviations of decision-making units (DMUs) from the frontier are due to technical inefficiency. A key advantage of this approach over SFA is that it more easily accommodates both multiple inputs and multiple outputs and since it is a nonparametric approach prior aggregation of the inputs or outputs is not necessary. Further, a specific functional form for the production process does not need to be imposed on the model (as is required in the use of the SFA approach).

Since its introduction in 1978, DEA has become one of the preeminent nonparametric methods for measuring efficiency and productivity of DMUs. DEA is a linear programming technique which determines the best-practice frontier from a set of peers (DMUs) and measures efficiency between best-practice and observed units using multiple inputs and outputs. DEA models are now employed routinely in areas that range from assessment of public sectors such as hospitals and health care systems and schools and universities to private sectors such as banks and financial institutions. The advantage of DEA is to accommodate multiple inputs and multiple outputs for measuring the relative efficiencies of a set of homogeneous decision-making units without a priori assumption of profit maximization and cost minimization. DEA models are useful for performance evaluation and improvement of DMUs, including the multidimensional aspects of service efficiency issues and operations that can help managers improve service performance.

On the other hand, multi-criteria decision-making (MCDM) models require complex optimization problems using multiple objectives than using a single objective of either maximizing profit or minimizing cost. The multiple criteria

DEA and other goal programming models are examples wherein the decision maker can use multiple outputs and multiple inputs to examine service performance and improvement. This book also reveals how DEA is used in multi-criteria decision making and as a benchmarking tool.

Both DEA and MCDM have been frequently applied for measuring efficiency and productivity of service industries. Service sectors include financial services (banking, insurance, securities, fund management), professional services (accounting, legal, engineering, architecture), health services, education services, environmental services, energy services, logistics, tourism, information technology, telecommunications, transport, distribution, standards and conformance, audio-visual, media, entertainment, cultural, and other business services.

With the exception of some basic notions in DEA, this book is completely self-contained. Important concepts and applications in measuring efficiency of the service sector are carefully motivated and introduced. Specifically, we have excluded any technical material that does not contribute directly to the understanding of measuring efficiency with DEA. Many other excellent textbooks are available today that discuss DEA in much more technical detail than is provided here. This book is aimed at upper-level undergraduate as well as beginning graduate students who want to learn more about measuring and managing service productivity with DEA and other MCDM techniques, or who are pursuing research in DEA and its applications.

The main objective of this book is to provide the necessary background to work with existing DEA models. Once the material in this book has been mastered, the reader will be able to apply DEA models to his or her problems for measuring comparative efficiency of decision-making units in any service industry.

To facilitate this goal, the first chapter provides a literature review and summary of the current research in DEA with a focus on the service sector. In this introductory chapter we present a classification scheme with seven main primary categories in service industry, namely, education, hospital and healthcare, tourism, banking and finance, information technology and media services, transportations, and utilities. We discuss each classification scheme and group selected DEA papers published in the literature over the past three decades. Finally, we provide information on the use of Performance Improvement Management Software (PIM-DEA). A free limited version of this software and downloading procedure is also included in this book. This advanced DEA software enables you to make the best possible analysis of your data, using the latest theoretical developments in DEA. PIM-DEA software gives you the capacity to assess efficiency and productivity, set targets, identify benchmarks, and much more allowing you to truly manage the performance of any service industry. PIM-DEA is easy to use and powerful, and it has an extensive range of the most up-to-date DEA models, which can handle large sets of data.

This is followed by chapter "Development of Assessment Model for Research Efficiency of Universities," where **Jong-Woun Youn and Kwangtae Park** argue that the research in university is an essential part for national competitiveness and the foundation of knowledge and information of a society. This chapter assumes that the effective operation of limited resources by size of universities would be the plan for maximizing their effectiveness and suggests a grouping of similar universities by establishing a new classifying system. Based on the classifying system proposed in this chapter, four models including High Efficiency Expanding Model (HEEM), High Efficiency Stable Model (HESM), Low Efficiency Stable Model (LESM), and Low Efficiency Expanding Model (LEEM) are suggested through a practical analysis.

In the same content of education, **Dimitris Sotiros, Yannis G. Smirlis, and Dimitris K. Despotis** present an alternative method to assess the quality and extent of research in higher education in chapter "Incorporating Intra- and Inter-Input/Output Weight Restrictions in Piecewise Linear DEA: An Application to the Assessment of the Research Activity in Higher Education." They proposed an extension of Piecewise Linear DEA to value-based piecewise linear DEA that incorporates value judgments and allows common treatment for intra- and inter-input/output weight restrictions. Value-based piecewise Linear DEA further enables a better expression of individual preferences, enhances the model with the fully units invariance property, and also resolves the discontinuity issue that exists in the original Piecewise Linear DEA model.

The next two chapters are examples of use of DEA in health care efficiency. **Felix Masiye, Chrispin Mphukaa, and Ali Emrouznejad**, in chapter "Estimating the Efficiency of Healthcare Facilities Providing HIV/Aids Treatment in Zambia: A Data Envelopment Approach," discuss that many countries in Sub-Saharan Africa face a key challenge of sustaining high levels of coverage of AIDS treatment under prospects of dwindling global resources for *HIV/AIDS* treatment. Policy debate in *HIV/AIDS* is increasingly paying more focus on efficiency in the use of available resources. The aim of this chapter is to provide a framework to estimate short-term technical efficiency of HIV/AIDS treatment facilities using DEA. An application in Zambia shows the applicability of the proposed model.

In the same area of health efficiency, a benchmarking approach based on closest targets is given in chapter "Benchmarking in Healthcare: An Approach Based on Closest Targets" where **Juan Aparicio, Fernando Borras, Lidia Ortiz, and Jesus T. Pastor** examine the process of benchmarking in hospital performance. In particular, this chapter shows that the determination of closest targets as a benchmarking technique has significant advantages over traditional DEA methods for signaling keys for the inefficient hospitals to improve their performance. In doing so, this chapter uses a sample of hospitals, located in the eastern region of Spain. Further, some guidance in relation to determining potential improvement targets for each of the inefficient hospitals is given.

Services are becoming increasingly important to the developed and developing economies. However, evidence shows that as production moves from agriculture and manufacturing to service- and knowledge-based economies, productivity growth rates have declined. To date, there are no clear indicators for quantifying productivity for service and network based firms. This raises the question: *How can productivity be measured for service and network based firms*? **Moira Scerri and Renu Agarwal**, in chapter "Service Enterprise Productivity in Action (SEPIA)," present a systems view of productivity, which is organized into five sections: overview of productivity; current measures of productivity using KLEMS; existing service productivity models; service enterprise productivity in action (SEPIA) model; and new measures for service enterprise productivity. The key contribution of this chapter involves the operationalization of the SEPIA model and an illustration of the model through the use of an industry example.

This is followed by measuring good governance in chapter "Using Data Envelopment Analysis to Measure Good Governance" where **Rouselle Lavado, Emilyn Cabanda, Jessamyn Encarnacion, Severa de Costo**, **and Jose Ramon Albert** provide an estimate of good governance index using the DEA with evidence from Philippine provinces. This chapter illustrates how DEA can be used to provide insights on how provinces can improve on various indicators of governance. Aside from identifying peers, DEA is also able to estimate targets, which can serve as a guide for central governments in holding provinces accountable. This chapter shows that DEA is not used only for efficiency measurement but also applied to other applications in benchmarking and index generation, including nonprofit sectors such as public agencies.

A DEA-based methodology is developed in chapter "Measuring the Performance of Service Organizations and the Effects of Downsizing on Performance: Evidence from the Greek Citizen Service Centers" to measure the performance of not-for-profit and for-profit service organizations. **Panagiotis D. Zervopoulos** proposes a methodology that can incorporate endogenous and exogenous variables in the production process, which are directly or inversely related. This methodology always identifies reference units that are qualified in all of the dimensions of performance. In addition, it defines appropriate changes to the resources that are used by the low-performing units to enable them to become qualified in all facets of performance at the optimal condition. The methodology that is developed in this chapter is applied to public organizations, which are in charge of the provision of administrative services to citizens, in two instances: before and after the implementation of downsizing as part of the public management reform agenda. The results obtained from the assessment methodology are the basis for the analysis of the impact of structural reform, and particularly of downsizing, on the performance of public service organizations.

Luciana Yeung, in chapter "Measuring Efficiency of Courts: An Assessment of Brazilian Courts Productivity," develops a DEA framework for measuring efficiency in the Judiciary, specifically in State Courts with an illustration from Brazil. The chapter argues that both inefficient and unstable units could use DEA results to improve their management and to achieve better results in their efficiency, productivity, and effectiveness in the delivery of judicial services.

This is followed by an application of cost-efficiency and market power in chapter "Cost Efficiency and Market Power: A Test of Quiet Life and Related Hypotheses in Indonesian Banking Industry." **Viverita** investigates the relation between market power and cost-efficiency (the *quiet life hypothesis*), and the two competing hypotheses of the relationship between market power and efficiency as well as market concentration on profitability (*Structure Conduct Performance* and

Efficient Structure). This is illustrated with an application in the Indonesian banking industry from 2002 to 2011. Further to DEA and to capture the equilibrium dynamic of the Indonesian banking industry, the Lerner index method is used to measure the level of competition. Results of this chapter fail to reject both *Structure Conduct Performance* hypothesis and *Efficient Structure* hypothesis, but disapprove the existence of the *quiet life hypothesis* in the Indonesian banking market.

Internal structure of service organizations is important in service productivity. **Ming-Miin Yu and Li-Hsueh Chen**, in chapter "Internal Structure of Service Organization: From Multi-activity Financial Institutions to Network Structure Hotels," discuss that in recent years, based on characteristics that operational processes of financial institutions and hotels may jointly engage in multiple activities and multiple processes. This chapter is dedicated to describing internal structures of financial institutions and hotels as well as providing relative DEA models and applications. The chapter illustrates that in order to conform to real operational situations, the construction of DEA model should consider and match the internal operational characteristics of decision making units.

As another application **Michael L. Antonio** and **Ma. Socorro P. Calara,** in chapter "Application of DEA in the Electricity Sector: The Case of Meralco Distribution Sectors," present an application of DEA in the electricity sector with the Case of Meralco Distribution Sectors. The chapter seeks to (1) evaluate and compare the technical efficiency performance of Meralco Distribution Sectors using selected Performance-Based Regulation (PBR) indicators and other inputs, (2) determine which Meralco Distribution Sector achieved the highest technical efficiency performance, and (3) identify areas for improvement of each Meralco Distribution Sector. A linear monotone transformation was adapted to make use of an undesirable output in the DEA model. The chapter's findings imply that the management of Meralco or distribution sectors need to formulate strategies and policies that would further improve their performances.

Chapter "Improving Energy Efficiency Using Data Envelopment Analysis: A Case of Walnut Production" is an application of DEA for improving energy efficiency in farms with a case of Walnut Production. **Alireza Khoshroo and Richard Mulwa** discuss that Walnut is one of the most nutritive crops and modern production methods that can require large quantities of energy. Efficient use of these energies is a necessary step towards agricultural sustainability. Hence, this chapter focuses on optimizing energy consumption in walnut production by identifying and reducing excessive use of energy. DEA is used to model efficiency as an explicit function of human labor, machinery, fertilizers-chemicals, and irrigation energies. The result of DEA analysis shows a substantial inefficiency between the Walnut producers in the studied area, with the main difference between efficient and inefficient producers being in the use of chemicals, potash, machinery, and irrigation water.

Chapter "Service Productivity in IT: A Network Efficiency Measure with Application to Communication Systems" focuses on more advanced DEA models such as network efficiency measure with the application to communication systems. **Adeyemi Abel Ajibesin, Neco Ventura, H. Anthony Chan, and Alexandru**

Murgu introduce a network efficiency measure, which is a new kind of thinking for many evaluators in information technology and engineering. Efficiency measure involves going beyond knowledge (real or estimated) of program (nodes, algorithms, networks, etc.) impact and attempting to compare with other programs. In most cases, this knowledge leads to a decision as whether to replace the program with another more effective program. In this chapter, DEA is applied to extend the existing engineering method in computer networks and to evaluate the efficiency of communication networks.

In the same area of IT efficiency, **Geeta Sharma**, in chapter “Efficiency of Software Development Projects: A Case Study on an Information Technology Company in India,” applies DEA to evaluate the relative efficiency of software development projects of a leading software company in India. In this chapter, projects are categorized as per their efficiency scores into highly efficient, moderately efficient, and less efficient companies through a process called Tier Analysis. The chapter also includes an improvement path for the projects with low efficiencies. Furthermore, through the application of Kruskal Wallis test, the software development project efficiency is compared with team size to determine whether efficiency varies across various team size categories, i.e., small, medium, large and extra-large.

The rest of this book is on the transport efficiency. **Darold T Barnum, John M Gleason**, **and Matthew G Karlaftis**, in chapter “Protocol for Comprehensive Efficiency Analysis of Multi-Service Metropolitan Transit Agency Operators,” present a DEA protocol for analyzing the efficiency of metropolitan transit agencies that oversee multiple types of transportation services. The protocol is illustrated by applying it to United States transit agencies that can serve their cities with four types of subunits: self-operated motorbus, outsourced motorbus, self-operated demand-responsive, and outsourced demand-responsive. Using DEA models adapted for non-substitutable inputs and outputs, scores estimated for a focus agency include: (1) technical efficiency of the focus agency as a whole, (2) technical efficiency of each of the focus agency’s subunit types when each subunit is compared only to others of the same type, (3) allocation efficiency of the focus agency in apportioning resources among its subunits, and (4) the effect of each subunit’s technical efficiency on its parent agency’s technical efficiency. Finally, a mathematical programming algorithm is illustrated that allocates the focus agency’s resources to its subunits with the objective of decreasing the cost of transit in an urban area while holding total ridership constant. The protocol thereby is a comprehensive analysis and synthesis of a focus transit agency’s efficiency in providing services to its metropolitan area.

Measuring the sustainability of air navigation services is subject of the chapter “Measuring the Sustainability of Air Navigation Services.” **Vladimir Grigorov and Paula Rachel Mark** discuss that service productivity is synonymous with the organizational sustainability. It has applications that are broader than conserving the environment via agroindustrial innovation. The domain of Air Navigation Services is a classic example of a service industry, the sustainability of which can be determined using its organizational efficiency and effectiveness. It is a challenge to measure these organizational factors in this profession, because of insufficient data and the effect of random events such as inclement weather that cannot be quantified. A DEA caters for these restrictions and is thus an appropriate tool for determining the sustainability of Air Navigation Service Providers. The DEA results highlight the need for urgent attention to the organizational structure of Air Navigation Services and the reallocation of resources that will improve sustainability.

Sreekanth Mallikarjun, Herbert F. Lewis, and Thomas R. Sexton in chapter "Measuring and Managing the Productivity of U.S. Public Transit Systems: An Unoriented Network DEA" explain that the U.S. governments at all levels face budget shortfalls, and consequently public transit systems in the United States must compete with other public services for financial support. In order to depend less on public funding, it is critical that public transit systems focus on their operational performance and identify any sources of inefficiency. In this chapter, they present an unoriented network DEA methodology that measures a public transit system's performance relative to its peer systems, compares its performance to an appropriate efficient benchmark system, and identifies the sources of its inefficiency.

In the same area of public transport, **Thomas R. Sexton, Allan J. Jones, Andy Forsyth, and Herbert F. Lewis**, in chapter "Using DEA to Improve the Efficiency of Pupil Transportation," provide an example of use of DEA in Washington State that like many other states spends hundreds of millions of dollars annually to support the transportation of pupils to and from school. As with other state-funded activities, inefficiency increases costs and saps resources away from other critical state functions such as public and higher education, health care, transportation, and many others. In 2006, the state undertook a project to revise its pupil transportation funding formula and encourage its school districts to operate more efficiently. Together with Management Partnership Services, Inc., the state developed a DEA-based efficiency measurement system that it now uses to identify inefficient pupil transportation systems for management intervention. The system has identified potential first-year savings of roughly $33 million, with recurrent annual savings of at least $13 million.

Birmingham, UK — Ali Emrouznejad
Virginia Beach, VA — Emilyn Cabanda
July 2014

Acknowledgments

This book would not have been possible without the help of a number of people. First among these are the contributing authors—without them, it was not possible to put together such a valuable book, and we are deeply grateful to them for bearing with our repeated requests for materials and revisions while providing the high-quality contributions. We are also grateful to the many reviewers for their critical review of the chapters and the insightful comments and suggestions provided.

Thanks are also due to Professor Camille C. Price, the Editor of this Series, for supporting and encouraging us to complete this project. The editors would like to thank Dr Christian Rauscher (Springer Senior Editor, Business/Economics), Gabriele Fischer (Springer Project Coordinator, Production Heidelberg), Syakam Ramakrishna Reddy (Springer Production Editor, Total Services) and Ms Ramya Prakash G (Project Manager) for their editorial and Production assistance in producing this volume.

We hope the readers will share our excitement with this important scientific contribution to the body of knowledge in use of Data Envelopment Analysis to Managing Service Productivity.

Contents

Efficiency of Software Development Projects: A Case Study on an Information Technology Company in India

Geeta Sharma and Anshu Kataria

Abstract In this chapter, Data Envelopment Analysis (DEA) is applied to evaluate the relative efficiency of software development projects of a leading software company in India. Further, the projects are categorized as per their efficiency scores into highly efficient, moderately efficient and less efficient companies through a process called Tier Analysis. The chapter also includes an Improvement Path for the projects with low efficiencies. Furthermore, through the application of Kruskal Wallis test, the software development project efficiency is compared with team size to determine whether efficiency vary across various team size categories i.e. small, medium, large and extra large.

Keywords Efficiency of software development projects • Data Envelopment Analysis • Team size • Managing service productivity

1 Introduction

In the last few decades, there has been a remarkable evolution in the global Information Technology (IT) sector. The lucrative growth of the IT Industry has been a result of continuous innovations and improvements in software development. Also, the growth of IT companies can be attributed to the increased need of software development services. The booming phase has pulled a high degree of competition in this industry. To stay competitive, software companies need to focus on productivity of the software development process.

The efficiency of the software development process has been an important research area for several decades, evident from the large body of literature on the topic. This chapter aims to measure the efficiency/productivity of software

G. Sharma (✉) • A. Kataria
IIPS, Devi Ahilya University, MP, India
e-mail: geetasharmaiips@gmail.com

A. Emrouznejad and E. Cabanda (eds.), *Managing Service Productivity*,
International Series in Operations Research & Management Science 215,
DOI 10.1007/978-3-662-43437-6_15,

development projects, as efficiency and productivity have been used interchangeably in this context (Poel and Schach 1983; Gack 2011).

The massive growth accomplishment in the Indian IT industry in the last decade has made a sizeable contribution to India's GDP. The industry has played an imperative role in putting India on the global map. The industry helped India in transforming from a rural and agricultural based economy, to a knowledge based economy and becoming a global player in providing world class technology solutions and business services. Hence, assessing the productivity of this industry is of importance, given its significant contribution to the development of the country's economy.

In general, productivity is the amount of output (what is produced) per unit of input used. Productivity, in case of software development, is difficult to measure because outputs and inputs are typically quite diverse and are often themselves difficult to measure. One can really not expect to improve software productivity without measuring it, and hence, measuring software development efficiency is a must, before we can talk about improvements in productivity or say efficiency. Measurement of the efficiency of software development projects yields quantifiable and objective information. In successful software organizations, measurement-derived, quantifiable and objective information on efficiency of software development is treated as an important resource and is made extensively available to the decision makers throughout the organization. Measurement of software efficiency aids in defining a standard for improvement and can be re-evaluated once the improvements have been made. Also, efficiency measurements are important as benchmarks to determine whether there is a need for improvement to survive in a competitive environment.

In this chapter, Data Envelopment Analysis (DEA) is applied, to evaluate the relative efficiency of software development projects of a leading software company in India. The study models software development as a microeconomic production process, utilizing certain inputs to produce products.

DEA is a mathematical programming methodology that can be applied to assess the relative efficiency of an array of institutions using a variety of input and output data. It is a theoretical framework for performance analysis and has been applied in many applications (Emrouznejad and De Witte 2010; Emrouznejad et al. 2008). It analyzes the performance data of homogenous units, which are called Decision-Making Units (DMUs). Each DMU produces a certain number of outputs using certain number of inputs/resources. Consequently, the productivity of software development process can be measured just as that of a production process by simply dividing the outputs produced by the inputs consumed.

The study also shows an improvement path for the software development projects with low efficiencies, by first categorizing the software development projects into highly efficient, moderately efficient and less efficient units through a process called as Tier Analysis. Furthermore, the DEA efficiency scores, as a dependent variable, have been checked for any kind of association/relationship with an independent variable—Team Size—applying the Kruskal Wallis test.

2 Literature Review

Studies on performance evaluation of software development by way of efficiency or productivity assessment have long been debated. Many such research pieces provide a base to this study.

Banker et al. (1991) developed an estimable production frontier model for software maintenance, using a new methodology that allows the simultaneous estimation of both the production frontier and the effects of several productivity factors. The model is then estimated using an empirical dataset of 65 software maintenance projects from a large commercial bank. Estimates of the marginal impacts of all the included productivity factors were obtained to aid managers in improving productivity in software maintenance.

Hong et al. (1999) showed that DEA can be used to evaluate the efficiency of systems integration projects. The inputs were—labor hours, and material and equipment resources. The performance measures, i.e. the output comprised of customer satisfaction index, schedule performance, budget performance and rework after delivery.

Anselmo and Ledgard (2003) focused on the issues concerning productivity of software development and showed that productivity is inversely proportional to the costs incurred in providing the specified functionality and quality.

Gack (2011) measured the efficiency of the software development process and stated that efficiency and productivity are essentially synonyms. He used the method ‘Value Stream Mapping’ and categorized all the activities involved in a software development into value added and non-value added activities.

Jiang and Naudé (2007) studied the variations in software development productivity by analyzing 4,106 projects from International Software Benchmarking Standards Group (ISBSG) data repository, and examined the factors that significantly influence the software development productivity. The factors included average team size, development language, development type, development platform, development techniques, development methodology and the use of CASE tools; and the research investigated their impact on software productivity. With the use of multiple regression analysis they explored that the variations in productivity are most affected by the team size and the development language. They also concluded that development platform and development techniques had moderately significant impact on the productivity variations.

Pendharkar and Rodger (2009) studied the relationship between team size and software development cost. They held that since larger size projects are expected to include large team size and vice versa, they used software project size in function points as a contextual variable and made use of real-world data from 540 software projects obtained from the ISBSG. The results indicated that increasing team size does not linearly increase software development cost.

Chinubhai (2011) measures software development productivity by considering multiple outputs such as functionality (i.e. number of function points delivered), quality (i.e. number of defects per function point) and the lead time (i.e. elapsed time per function point) as a result of the software development process which takes in development effort (i.e. man-hours) as an input. The data on software development projects is obtained from the ISBSG repository. The methodology used in the study is the measurement of efficiency scores using a DEA model.

Rodriguez et al. (2012) empirically found out the impact of team size and other development factors on the productivity of software projects. They scrutinized the ISBSG repository of 4,106 projects and selected a sample of 951 projects. The results showed that projects with an average team size of 9 are less productive than those with a higher average team size.

Sudhakar et al. (2012) stated that the objective of any software business organization to reduce costs and to increase profitability is to achieve maximum team productivity. They concluded the organizational efficiency is dependent on individual and team productivity and one should try to increase the productivity of software development teams.

3 Objectives of the Study

Competition in software industry has increased significantly. As said earlier, one of the ways software companies can stay competitive is to improve software development productivity of their software products. Effective management of software development productivity leads to a better competitive position for an organization. Efficiency in the context of software development has traditionally been measured as the ratio of functionality, either lines of code or functions points, and the effort expended which ignores number of factors such as quality and elapsed time. Other measures such as case points, object points and feature points are also used in literature and some IT organizations to measure the productivity. This study utilizes Data Envelopment Analysis (DEA) to develop a multidimensional measure of efficiency in the context of software development. The objective of this analysis, therefore, is to measure efficiency using multiple measures of outputs and inputs and then to examine how efficiency varies with the factors such as team size. Thus, the objectives of the study are as follows:

1. To measure the efficiency of software development projects of a leading software company in India through the application of DEA.
2. To categorize the software development projects of a leading software company in India, on the basis of their efficiency and explore an improvement path for projects with less efficiency.
3. To explore the difference of efficiency of software development projects of a leading software company in India among various team size categories of those projects.

4 Methodology

4.1 Research Design and Sample

This research work is based on the case study design of research and is an exploratory research. It is also an analytical-cum-empirical research. According to Zainal (2007: 1): "Case studies, in their true essence, explore and investigate contemporary real-life phenomenon through detailed contextual analysis of a limited number of events or conditions, and their relationships". A thorough study of a particular situation is what a case study implies. It narrows down the extensive and broad field of research into an easily researchable topic. This research provides insights about the efficiency of software development projects of a leading software company in India. It is an exploratory research because this will provide new insights into a particular phenomenon that efficiency of software development projects. It is also an analytical research because existing, available facts and information are used for a critical evaluation. Empirical research is a data-based research, which is used when proof is sought that certain variables affect other variables in some way. One of the major objectives of the study is to test the association of an independent variable, i.e. team size and a dependent variable, i.e. operational efficiency of application software projects, thus giving it a shape of an empirical research as well.

Since the study revolves around efficiency of application software projects of a leading software company, the case unit is nothing but the particular leading software company. According to Benbasat et al. (1987), in case studies, the case and the units of analysis should be selected intentionally. This is in contrast to surveys and experiments, where subjects are sampled from a population to which the results are intended to be generalized. The purpose of the selection may be to study a case that is expected to be "typical", "critical", "revelatory" or "unique" in some respect. The case unit for this research is the leading software company in India. It is a part of the 58 companies recognized by The National Association of software and services companies (NASSCOM). These are the CMM (Capability Maturity Model) level 5 certified companies in India. The case unit selected is geographically-diversified and has a presence in major parts of the country including Vadodara, Indore, Noida, Mumbai, Hyderabad, Bangalore and Chennai.

Next, the sample projects, which would constitute the data set, had to be identified. For this purpose, the method of convenient sampling is adopted. A total of 173 software development projects completed during the year 2011–2012, of the identified company/case unit, are considered for the study. These 173 software development projects constitute the population. Of these 173 projects, the project closure forms of only 40 projects could be obtained. Hence, a final sample of 40 projects is considered for this study.

In the present study, a DEA model is proposed, to evaluate the relative efficiency of software development projects of the identified leading software company. The study models software development as a microeconomic production process, utilizing certain inputs to produce products. DEA is a theoretical framework for performance analysis. The choice of DEA is motivated by the need to simultaneously consider multiple inputs and products and to not impose an arbitrary parametric form for the underlying production correspondence. Furthermore, DEA estimates the minimum amount of inputs required, given the size and complexity of the project, rather than the average amount of inputs, which would be estimated using regression-based methods. The former is more meaningful for management control and efficiency evaluation purposes and is consistent with a microeconomic definition of a production function (Banker et al. 1987). It analyzes the performance data of homogenous units, which are called as the decision-making units (DMUs). Each DMU produces a certain number of outputs using certain number of inputs/ resources. Each software development project is a DMU in this process.

Hence, the present study measures the efficiency of software development projects using DEA. The study also shows an improvement path for the software development projects with low efficiencies, by first categorizing the software development projects into highly efficient, moderately efficient and less efficient units through a process called Tier Analysis. Furthermore, the DEA efficiency scores, as a dependent variable, have been checked for any kind of difference among an independent variable—Team Size categories—applying the Kruskal Wallis test.

4.2 Data Collection

Secondary data is used for the purpose of this study. The secondary source of data is the project closure form, which is the source of information for variables including the schedule, size, cost, effort, and other such details of the software project required for conducting the research successfully.

4.3 DEA Model for Measuring Efficiency of Software Development Projects of IT Company in India and Variable Description

The study circles around the productivity or what is synonymously called efficiency of software development projects, as the dependent variable for the study. Team size for each sample project is the independent variable for the research.

4.3.1 Efficiency of Software Development Projects

The efficiency of software development process can be measured just as that of a production process by dividing the outputs produced by the resources consumed. The present study measures the efficiency of software development projects using DEA. The following section defines the input and output variables used to model software development akin to a production process.

Input and Output Variables for DEA Model for Software Development Projects

Using the DEA technique requires an appropriate estimate and choice of inputs and outputs to be utilized. An observer will have many different choices with respect to scope and nature of both, the product/output produced and the resources utilized to produce these products.

Based on the literature and the constraints related to data availability, the input variables identified for this study are total project effort and actual project cost; while the outputs are project size, quality, and Schedule Performance Index (SPI). The model for this research can be explained in Fig. 1.

Developing new software quickly, successfully, and at low cost is critical in organizations (Akgun et al. 2007). In this chapter both project costs and project effort are included to ensure that the input variables have been holistically captured. Including the two variables help us cover cost efficiency, as well as, time for completion—the two critical project aspects for executives and managers. To survive, IT firms must accelerate the time to market for their software products. However reduced cycle times cannot be achieved at the expense of high development costs (Harter et al. 2000).

Moreover, it makes the analysis indifferent to variance in billing rates, typical of software projects handled by Indian IT firms, across projects and helps to benchmark projects against one another.

Actual Project Effort

The actual project effort is the total project effort expended in terms of hours from the date of project commencement to the date of project completion. When considering the effort for a software project, it is usually expressed in man-day/month/year, defining the total time taken by the project team to develop the software project. The effort put in is generally on the activities like project initiation, analysis, design, coding, testing, documentation, external rework, production support, etc. Also, to accomplish a project successfully, it is necessary as it makes all other project elements working together. For this purpose, activities like metrics analysis, review, consolidation, preparation and maintenance of project plans and status reports, project audits, project reviews and meetings, work

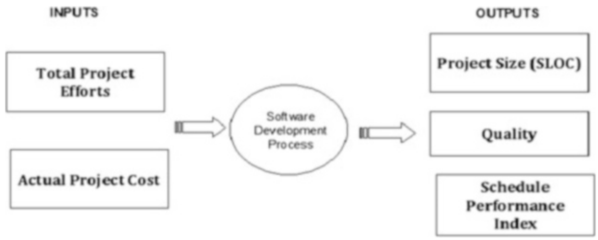

Fig. 1 DEA model for measuring efficiency of software development projects of an IT Company in India

allocation and issue resolution are conducted. It also involves activities that are conducted solely by the project manager, such as project planning and monitoring, administrative tasks, and steering group meetings. Since professional data processing staff time is the most expensive and scarce input resource in software development, work-hours has been the variable of interest in most previous studies. Chinubhai (2011) states that software development is a more of labor driven procedure, and therefore, this study used work effort measured in man hours as the sole input. And thus, the effort expended by the project team, in terms of hours, on the above mentioned and all the other activities required to complete a software project is a major variable to be entered into the model as an input. As the economic theory of productivity in context of minimization of inputs suggests, the lower the effort, the higher will be the operational efficiency of software projects.

Actual Project Cost

It is the actual and total cost of the project, i.e. the amount spent from the date of project initiation till the time it is completed. The input project cost manifests the development cost of the project including labor, overhead and other expenses (Paradi et al. 1997). Paradi et al. (1997) explained the costs as an input for software production process. Not many studies used the project cost as an input to software production process in the literature. But costs incurred, apart from the labor hour input, are a major determinant of the total resources consumed for producing software, as well as, a key factor affecting the efficiency of a software project. Hence, the study incorporates the use of 'project cost' as an input to the software production process, which will help determine the efficiency of the projects with the use of DEA technique. The cost is represented in dollars in this study.

Following Hong et al. (1999), this study uses the SPI as the output variable to be considered for measuring software efficiency, apart from project size. The relevance of choosing these variables can be seen in the following paragraphs.

Project Size

DEA helps identifying the minimum amount of input required, given the size of the project. Size is an important characteristic of any software. Hence, the significance of size of the project as an output cannot be ignored. Source Line of Code (SLOC) is the most traditional and extensively used 'sizing' measure for software projects as suggested by the literature. Albrecht (1979) and Albrecht and Gaffney (1983) developed a new measure of productivity by replacing the lines-of-code measurement with function point. Since Albrecht's work, researchers have followed using this dimension of software development as a major input to determine the efficiency. In a study conducted by Cusumano and Kemerer (1990), the lines-of-code metric was chosen to compare efficiency across a number of organizations in the USA and Japan.

Conventionally, in software development terms, productivity is defined as SLOC per man-month. Even though the LOC metric has been a much debatable issue, the fact is that many organizations still consider it as a more practical productivity metric than other options available (Boehm 1987). As the end product of any software development project is a coded program, the conventional and the most used measure of a software product has been the count of the number of written source lines of code (SLOC). SLOC are easily countable by automated measures, and it also represents the amount of work required to build a system (Banker et al. 1991). Banker also stated that SLOC is generally considered to be an appropriate proxy for all project activities. He used SLOC (for the testing and coding phase) as a measure of size of software development projects.

Consequently, this study makes use of NCSLOC (i.e. Non-Commented Source Lines of Code) as a measure of size of the software development project. Actual project size is the actual product size measured in terms of NCSLOC delivered at project completion, i.e. the delivery of the end product and its acceptance by the customer.

Project Quality

As a huge deal of effort is expended on assuring quality software, Paradi et al. (1997) suggested that quality should be included in efficiency and productivity measures. Hence, another major independent variable considered in this study is the quality of the software project delivered. Paradi et al. (1997) considered quality as a major output in the DEA model for determining efficiency of software production at two Canadian banks. Chinubhai (2011) stated quality per function point as the undesirable output that need to be minimized while maximizing the number of function points delivered. The author included the variable directly in the model as an output by transforming the undesirable outputs with the use of additive inverse approach used by Koopmans (1951) where an undesirable output Q is transformed using the additive inverse $f(Q) = -Q$. The quality measure considered in this study is similar to that used by Ramasubbu and Balan (2007). It incorporates the number of different issues/problems detailed by the customers during the acceptance tests and production trials. It is calculated as follows:

$$\text{Quality} = 1/(\text{No.of defects} - 1)$$

This formula represents that quality will increase as the number of defects decreases. Since it is a ratio, the study uses its log for calculating efficiency scores (Emrouznejad and Cabanda 2010; Emrouznejad et al. 2010).

In addition, it adheres to the production model assumption of output maximization, which was observed as a limitation in the previous studies—like that of Paradi et al. (1997)—which used the number of defects and the rework hours as the measure of quality. As a result, the assumption of 'output maximization' of a production model was violated. Hence, this variable could be directly linked with the operational efficiency of the software development projects.

Schedule Performance Index

It is a known fact that in order to be successful, a project should be completed on time and under budget. Projects that are behind schedule may go over budget and end being low on efficiency. Hong et al. (1999) states that the anticipated benefits of a completed project may be lost or delayed due to a project being behind the schedule or being over-budget. A significant concern is the level of practicality in the initial schedule and budget.

The SPI is a measure of finding out how well a project is accomplished in terms of following the schedule. It is a comparison of the part of the project that is planned to be accomplished and the part actually accomplished. The benefit of using this index as a measure is that it allows for comparison of projects of different sizes. SPI is dividing the estimated period of completing the project by the actual period of completing the project. It shows how the project is progressing compared to its schedule. It can be said that if the project is following its plan, the amount/value of work accomplished and the amount of money spent to accomplish it are the same, and the resulting value will be one. So an index of one means that the project is following its plan. Mathematically, SPI can be expressed as:

$$\text{SPI} = \text{Planned Period}/\text{Real Period}$$

Here, Planned Period is the period between the estimated start date and the estimated end date, and Real Period is the period between the actual start date and the actual end date.

Clearly, the formula indicates that the greater the SPI, the better will be the project performance as it will reflect the conformance of the project to the planned schedule. This study first calculates this measure with the available data of the planned and actual start date and end date, and then uses it for calculating efficiency scores.

Data Envelopment Analysis (DEA), a non-parametric method for measuring efficiency of DMUs, is usually undertaken with absolute numerical data, which reflects the size of the units. Thus, using ratio variables increases the likelihood of error. However, literature reveals that DEA has been applied with ratio variables in the past. To make the analysis error-proof, it is recommended that ratio variables (typically a combination of absolute input and output variables) be split into their components (Emrouznejad and Amin 2009). Given SPI is not a combination of input–output variables, breaking it down further, would possibly not yield the expected results. Hence, the study has used log data for calculating efficiency score (Emrouznejad and Cabanda 2010; Emrouznejad et al. 2010).

Hence, the study considers SPI to be the best suited measure to capture relative lag (or lead)—an important project aspect for assessing efficiency. It calculates SPI with the available data of the planned and actual start date and end date, and then uses it for calculating efficiency scores.

Team Size

Team size is the headcount of number of people working for a project. The influence of team size on the efficiency of software development projects has been a topic of research in software engineering for many years. Team size, as a factor that impacts software efficiency, has been used in several studies. The size of software project teams has been considered to be a driver of project productivity (Rodriguez et al. 2012). Rodriguez et al. (2012) mentioned that according to Putnam (1978) there is an optimum team size, defined as the one which allows the development team to achieve the maximum efficiency with shortest schedule and lowest cost without affecting the final outcome. Chatzoglou and Soteriou (1999) incorporates the number of members working for a particular project as a major determinant of software efficiency. In a global survey of different countries, Blackburn et al. (1996) argued that smaller teams may be more productive. However, the authors said that the affirmation about small team size and productivity are rarely supported by concrete facts. Large team size might hold back productivity due to the problems of coordination and communication between the members of the team. On the other hand, Banker and Kemerer (1989) argued that software projects might benefit from larger team size as specialized personnel with expertise in certain areas might improve overall productivity. Pendharkar and Rodger (2009) also held that the team size has a significant impact on the efficiency of software development projects. The ISBSG data also support that one of the main factors that affects the software development efficiency is the team size.

In view of the facts above, this study considers team size as a major determinant of the operational efficiency of software development projects. Hence, this factor is studied by linking it to the efficiency scores obtained via DEA for the projects.

The impact of team size is examined on the efficiency of software development projects by comparing it to the efficiency scores computed through the application of DEA. Efficient and non-efficient projects are observed in context of their team size.

5 Data Analysis and Results

5.1 *Measuring the Efficiency of Software Development Projects Through the Application of DEA*

The study uses the DEA (VRS) technique to evaluate the efficiency of application software development projects and models the software development as a

Table 1 Efficiency scores of software development projects of an IT Company in India

Project code	Efficiency scores	Project code	Efficiency scores	Project code	Efficiency scores	Project code	Efficiency scores
1	0.97	**11**	1.00	**21**	1.00	**31**	0.95
2	1.00	**12**	1.00	**22**	0.90	**32**	0.95
3	0.98	**13**	0.98	**23**	0.79	**33**	0.99
4	0.99	**14**	0.95	**24**	0.94	**34**	1.00
5	0.93	**15**	1.00	**25**	0.92	**35**	1.00
6	1.00	**16**	0.93	**26**	0.91	**36**	0.96
7	0.87	**17**	1.00	**27**	1.00	**37**	1.00
8	0.98	**18**	1.00	**28**	0.96	**38**	0.90
9	1.00	**19**	0.90	**29**	1.00	**39**	0.96
10	1.00	**20**	0.91	**30**	0.99	**40**	0.93

microeconomic production process. The DEA model applied, evaluates the relative efficiency of projects.

It analyses the performance data of homogenous units, described above as DMUs. Each DMU produces a certain number of outputs using certain number of inputs/resources. The technique helps in determining the group of most and least efficient DMUs (i.e. application software development projects). Using DEA, the software development projects are grouped into either an efficient or an inefficient group. Table 1 provides the efficiency scores obtained on the application of the output oriented DEA model corresponding to the respective software development projects.

5.2 Categorization of Software Development Projects and Development of an Improvement Path

Further, the Tier Analysis is used to classify the projects based on their efficiency scores. The approach, as proposed by Hong et al. (1999), is a kind of technique that can be used to cluster the projects/DMUs together based on their efficiency levels. Basically, the Tier Analysis is a simple iteration of the DEA technique. Through the application of DEA model, the target change for each of the projects is observed and the projected values of the parameters are used to define the improvement path.

Target change, for this study, means by how much the output can be proportionally expanded without altering the inputs consumed; or alternatively by how much can the resources/input being consumed be proportionally reduced without changing the output produced. The target changes indicate the improvement path for each of the tiers.

The first step of Tier Analysis involves obtaining efficiency scores of all the software development projects. The scores reveal the most efficient group of software development projects—these projects have a score equal to 1. This grouped is titled as ‘Tier I’. Further, DEA is applied again but only with the relatively inefficient projects which are not a part of Tier I. Projects with efficiency scores of 1. The same procedure is repeated until all the projects are categorized.

Accordingly, all 40 software development projects are grouped into three tiers. This facilitates proper discrimination between more and less efficient projects. The three tiers are categorized as ‘Highly Efficient’, ‘Moderately Efficient’ and ‘Less Efficient’. Also, this part of the analysis will help achieve two major objectives of the study—first, to categorize the software development projects on the basis of their efficiency and define an improvement path for software development projects with less efficiency scores; and second, to explore relevance between the dependent variable i.e. efficiency scores and the independent variable i.e. team size.

The improvement path is simply the path for less efficient and inefficient software development projects to become as efficient as those in Tier I, which constitutes the most efficient projects.

The first step involves identifying the projects within Tier I, which comprises of projects categorized as ‘Highly Efficient’. As part of the first iteration of the DEA application, 15 projects having efficiency score 1 are classified under the Tier I category. Those are project number 2, 9, 10, 11, 12, 15, 17, 18, 21, 27, 29, 34 and 35 as depicted in Table 1). They are the most efficient projects among all 40 projects. Hence, the marginal improvement or change required in these 15 projects is zero.

Further as a part of Tier Analysis, DEA is applied on projects excluding those that have been classified as Tier I. Hence, 25 software development projects are used for the second phase of Tier Analysis. The second iteration of the DEA technique generates a group of 11 projects that are classified as Tier II. These 11 projects are judged efficient in this category—implying that they are globally inefficient, but locally efficient and they are categorized as ‘Moderately Efficient’. An overall noticeable change is required in Tier II on an average, to become as efficient as Tier I projects.

The study substantiates on Tier II projects to become the most efficient, by proposing empirical dimensions on how much can input quantities be proportionally reduced without changing the output quantities being produced.

Table 2 shows the projected values for the parameters in comparison to the actual values, which facilitates the measurement of the target changes, and determination of the improvement path. These are the values that the parameters (Effort, Cost, Size, Quality, and SPI) must be equal to, for the software development projects to be the most efficient. The radial and slack movements for the input and output variables are used to find out the projected values of the parameters. The value of the radial movement indicates the amount by which the output must be increased and the slack movement reflects the amount by which the inputs must be reduced.

Table 3 provides details on the average target change that is required in each of the parameters of Tier II projects.

The aggregated results for Tier II show that operational efficiency of software development processes depends significantly on the capabilities of developers in improving project size, and schedule by 36 %, and 21 %, respectively, by using advanced tools and methods (Table 3).

Table 2 Actual and projected values for 'Moderately Efficient' software development projects representing Tier II (11 projects)

Project code	Actual size (SLOC)	Projected size (SLOC)	Actual quality	Projected quality	Actual SPI	Projected SPI	Actual effort	Projected effort	Actual cost	Projected cost
3	25,578	26,056	0.03	0.03	1.02	1.00	6,981	6,892	432,054	432,054
13	3,420	27,167	0.02	0.03	1.00	1.00	6,845	5,261	330,449	330,449
21	4,700	4,700	0.02	0.02	0.93	0.93	811	811	40,000	40,000
22	18,300	20,433	0.03	0.03	0.52	0.82	2,483	2,483	95,654	95,654
23	72,486	91,449	0.01	0.02	0.37	0.95	9,204	9,204	462,354	462,354
28	1,612	6,230	0.03	0.03	0.70	0.83	1,372	1,372	65,586	65,586
30	978	14,867	0.03	0.03	0.88	0.90	2,570	2,570	177,330	155,076
33	55,396	55,716	0.01	0.03	1.00	1.03	5,572	5,572	407,400	207,378
36	19,513	20,298	0.01	0.02	1.00	1.20	5,659	4,650	261,025	261,025
38	3,324	13,393	0.03	0.04	0.37	0.54	2,265	2,263	140,321	140,321
39	7,300	9,037	0.03	0.04	0.60	0.70	2,139	2,139	147,584	108,270

Table 3 Average target change required in Tier II 'Moderately Efficient' software development projects

Parameters of efficiency of software development project	Average target change required in moderately efficient projects (%)
Cost in Dollars	10
Effort in Hours	6
Size (SLOC)	36
Quality	13
SPI	21

All 14 remaining projects are a part of Tier III, Tier III constitutes the least efficient projects among the sample—categorized as 'Less Efficient projects'. Table 4 provides details on the target change that is required in each of the parameters of Tier III projects.

Table 5 shows the improvement path for Tier III. The improvement path indicates a reduction of 25 % and 30 % in development cost and labor hours, respectively, coupled with an improvement in schedule and size by 39 % and 30 % for Tier III project. Moreover, the distinguishing traits of less efficient projects are their inability to improve size and adherence on schedule.

Table 6 summarizes the improvement path for Tiers II and III, based on the Tier Analysis. As expected, the percentage improvement required in various metrics increases as we move from Tier II to Tier III with an exception that average SPI is lowest for Tier II.

Also evident from Table 6, the software development teams need to focus on increasing the size along with an improvement in adherence on schedule to produce projects with better efficiency.

The graphical representation (Fig. 2a,b, c, d, e) of input and output metrics of different tiers suggests that though highly efficient projects tend to have the highest average cost and efforts, but they also produce highest size and adhere to schedule and that makes the Tier I projects the 'Highly efficient project'. Graphical representation also shows that the higher input i.e. cost and man hours couple with declining output specially size and inability to adhere on schedule makes Tier 3 projects 'less efficient projects'. The result also indicates that in all categories of project i.e. Tier I, II and III, the quality is not a major issue.

Current economic conditions are forcing software companies to focus simultaneously on decreasing costs and effort while increasing software size and being schedule compliant along with increased quality. Improving software productivity is becoming critical because software costs of large in-house software companies have been increasing rapidly. However, for many organizations, measuring software productivity has been a difficult task. Using DEA, this research study investigates the productivity of 40 software development projects. The results

Table 4 Actual and projected values for "Less Efficient" software development projects representing Tier III (14 projects)

Project code	Actual size	Projected size	Actual quality	Projected quality	Actual SPI	Projected SPI	Actual effort	Projected effort	Actual cost	Projected cost
1	23,915	24,768	0.02	0.02	0.99	1.17	9,862	7,685	621,325	452,041
5	36,351	39,121	0.01	0.02	0.81	1.13	7,326	6,340	343,891	343,891
7	16,782	19,262	0.01	0.02	0.65	1.21	5,669	4,836	354,828	273,468
8	4,547	14,493	0.03	0.03	0.92	1.01	2,643	2,301	158,580	158,580
14	5,581	7,285	0.02	0.02	0.90	1.15	2,553	2,553	153,168	148,027
16	31,026	33,362	0.02	0.02	0.80	1.11	7,951	7,951	548,598	468,613
19	10,950	14,819	0.02	0.02	0.75	1.22	10,714	4,164	404,652	237,298
20	2,800	8,672	0.03	0.03	0.60	0.88	2,534	1,923	101,452	101,452
24	3,760	14,819	0.01	0.02	0.90	1.22	6,954	4,164	311,857	237,298
25	10,063	14,819	0.01	0.02	0.84	1.22	9,048	4,164	530,638	237,298
26	7,253	12,871	0.01	0.02	0.78	1.21	5,003	3,796	216,377	216,377
31	20,652	21,800	0.01	0.02	0.93	1.20	7,901	5,219	560,000	294,131
32	8,921	14,819	0.01	0.02	0.94	1.22	7,225	4,164	297,416	237,298
40	2,449	6,510	0.02	0.02	0.81	1.14	3,305	2,840	127,742	127,742

Table 5 Average target change required in Tier III 'Less Efficient' software development projects

Parameters of efficiency of software development project	Average target change required in less efficient project (%)
Cost in Dollars	25
Effort in Hours	30
Size (SLOC)	30
Quality	15
SPI	39

Table 6 Tier analysis summary for average target change required software development projects

	Average percentage improvement required	
Parameters of efficiency of software development project	Moderately efficient (%)	Less efficient (%)
Cost in Dollars	10	25
Effort in Hours	06	30
Size (SLOC)	36	30
Quality	13	15
SPI	21	39

of this study have practical implications for software project managers undertaking software development. The results showed that the DEA technology can be successfully used to identify efficient and inefficient software projects.

5.3 *Software Development Project Efficiency and Team Size*

Empirical findings in existing literature more or less support this notion that the size of software project teams has been considered to be a driver of project productivity (Rodriguez et al. 2012). Rodriguez et al. (2012) mentioned that according to Putnam (1978), there is an optimum team size, defined as the one, which allows the development team to achieve the maximum efficiency with shortest schedule and lowest cost without affecting the final outcome. Chatzoglou and Soteriou (1999) incorporates the number of members working for a particular project as a major determinant of software efficiency. Blackburn et al. (2006) argued that smaller teams may be more productive. Large team size might hold back productivity due to the problems of coordination and communication between the members of the team. On the other hand, Banker and Kemerer (1989) argued that software projects might benefit from larger team size as specialized personnel with expertise in certain areas might improve overall productivity.

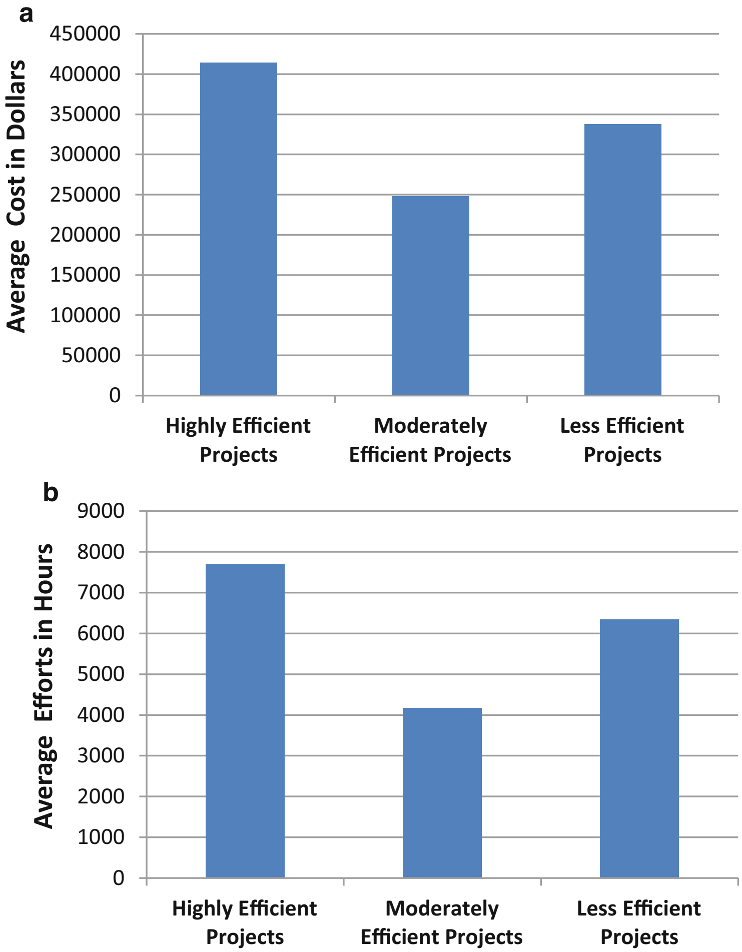

Fig. 2 (continued)

To explore, whether or not this stands true for the case unit of this research, an attempt is made to explore any difference in efficiency of software development projects among team sizes. For this purpose, the team sizes of the software development projects are grouped into four groups—Small, Medium, Large and Extra Large—through simple Quartile Division, which is shown in Table 7. Each quartile contains 25 % of the observations. The data are arranged in ascending order and to obtain the quartile positions. Thereafter, the Kruskal Wallis test is conducted with the help of SPSS 17.0. These results are presented in Table 8.

As evident from Table 9, a major part of the group of highly efficient projects seems to fall in the category of extra large team size, i.e. 33 % of the Tier I projects have a team size ranging from 46 to 90 persons.

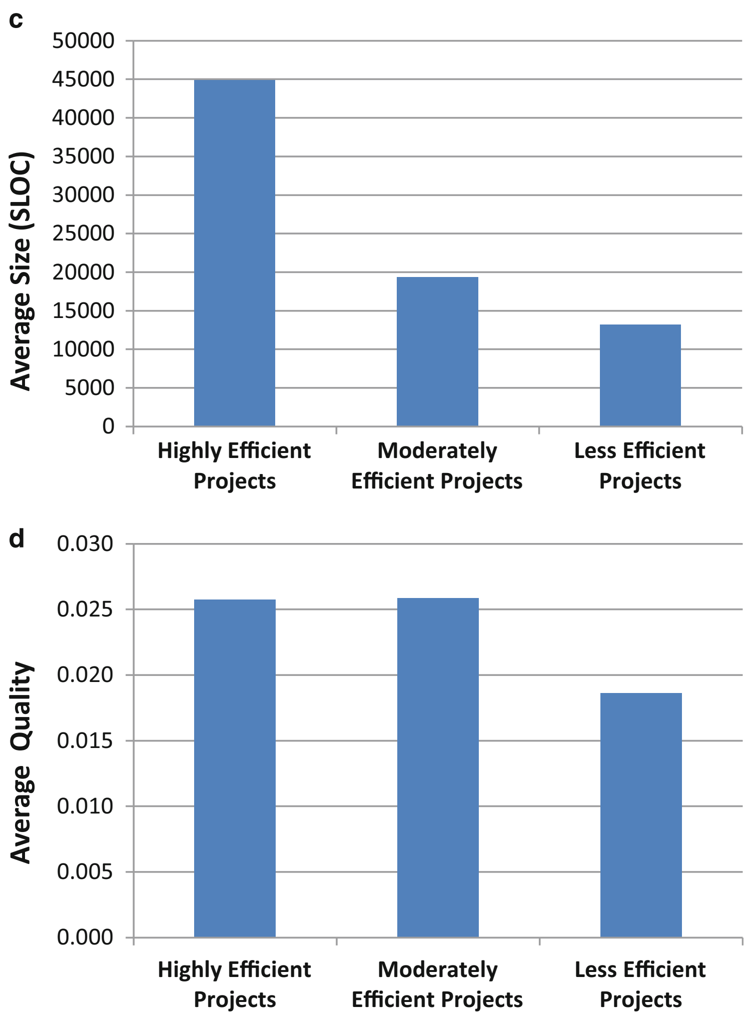

Fig. 2 (continued)

The test statistics of 2.724 and P value of .436 indicates that there is no significant difference in efficiency of software development projects among team size group categories i.e. small, medium, large and extra large. Team size appears to be less relevant, as long as a project delivers an optimum output by consuming minimum inputs.

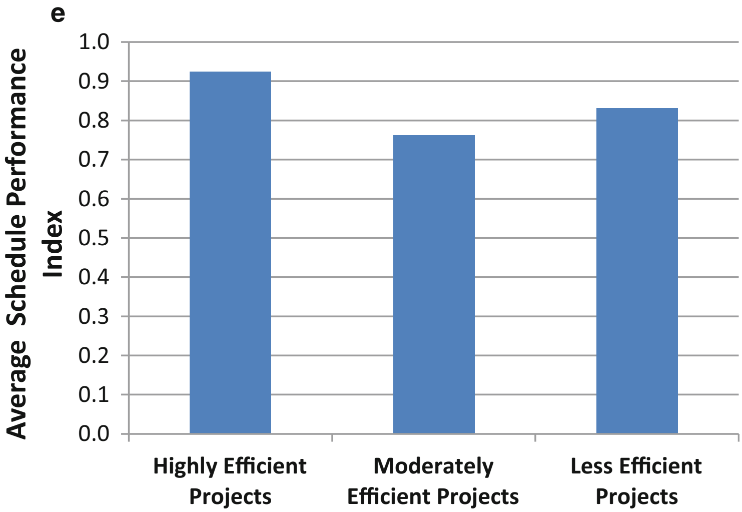

Fig. 2 (**a**) Efficiency and Cost of Software Development Projects of IT company in India. (**b**) Efficiency and Efforts of Software Development Projects of IT company in India. (**c**) Efficiency and Size (SLOC) of Software Development Projects of IT company in India. (**d**) Efficiency and Quality of Software Development Projects of IT company in India. (**e**) Efficiency and Schedule Performance Index of Software Development Projects

Table 7 Classification of team size of software development projects

Team Size	Quartile (%)	Classification
0–14	25	Small team
15–25	50	Medium team
26–45	75	Large team
46–90	100	Extra large team

Table 8 Kruskal Wallis statistics for team size and efficiency of software development projects of IT company in India

	Project efficiency
Chi-Square	2.724
df	3
Asymp. Sig.	.436
– Grouping variable: team size	

6 Conclusion and Limitations

The chapter measured the efficiency of software development projects at a leading software company using DEA. The DEA technique is applied on a sample of 40 out of a population of 173 projects, developed during 2011–2012 by the case unit, a software company in India.

Table 9 Descriptive statistics for association between team size and efficiency of software development projects of IT company in India

Project category↓		Team size				
		Small	Medium	Large	Extra large	Total
Highly efficient projects	Count (%)	2(13.3 %)	4(26.7 %)	4(26.7 %)	5(33.3 %)	15(100 %)
Moderately efficient projects	Count (%)	4(36.4 %)	2(18.20 %)	1(9.1 %)	4(36.4 %)	11(100 %)
Less efficient projects	Count (%)	4(28.6 %)	4(28.6 %)	5(35.7 %)	1(7.1 %)	14(100 %)
Total	Count (%)	10(25 %)	10(25 %)	10(25 %)	10(25 %)	40(100 %)

As an initial step, input and output variables are identified by reviewing the literature available. Once the inputs and outputs for the software development process are finalized, DEA is applied on the sample of projects and their efficiency scores are obtained.

Using the Tier Analysis, projects are categorized into 3 tiers—'Highly Efficient', 'Moderately Efficient' and 'Less Efficient'—based on their efficiency scores. Tier I comprised of 23 projects, Tier II consisted of 8 projects and Tier III had 9 projects.

Next, the study defined an improvement path for the projects with lower efficiency. It is found that the percentage improvement required in the various metrics increases as we move from Tier I projects (Highly Efficient) to Tier III projects (Less Efficient).

Qualitatively, there are three traits exhibited by Tier I projects (Highly Efficient)—'Fast', 'Large' and 'Better'- which represent time to deliver, size and quality, respectively.

The charts in Fig. 2 support the 'Fast-Large-Better' delivery of projects in the Tier I category. It is observed that on moving from Tier I to Tier III average size and quality and adherence to time to deliver decreases.

Interestingly, an aberration can be observed while plotting SPI—where a higher value indicates that the project was completed well within the planned duration. Tier III appears to have a higher average SPI than Tier II, indicating that Tier III projects are delivered well within the planned completion time. Also, it is observed that software development teams need to work on improving the software quality, along with an increase in size and functionality of the software, to improve project efficiency.

Furthermore, the software development project efficiency is compared with team size to determine whether efficiency scores vary across various team size categories i.e. small, medium, large and extra large through Kruskal Wallis Test. The result indicates that it does not matter if the team size is small, medium or large as long as the projects deliver an optimum output by consuming minimum inputs.

6.1 Limitations

Due to unavailability of data and to avoid complexity in the model, a few less vital factors affecting the software development process are overlooked and only 40 projects of the case unit are studied.

Different programming languages have varied effects on effort. This study has not accounted for these consequences because it is difficult and impractical to consider the effect of various types of programming languages.

Next, while there are number of development techniques (e.g. waterfall, prototyping) used in the sample, their interactions were not considered. The reason is that except for waterfall, others are not used in enough number of projects. This limitation provides a motivation to continue this research in future work.

The sample contains 40 software development projects; they are all gathered within one case (i.e. a software company in India). Therefore, the external validity of the results remains to be demonstrated.

References

Akgun, A. E., Keskin, H., Byrne, J., & Imamoglu, S. Z. (2007). Antecedents and consequences of team potency in software development projects. *Information and Management, 44*, 646–656.

Albrecht, A. J. (1979). Measuring development productivity. In *Proceedings of the Joint Share/Guide/IBM Application Development Symposium* (pp. 83–92)

Albrecht, A. J., & Gaffney, J. E. (1983). Software function, source lines of code and development effort prediction: A software science validation. *IEEE Transactions on Software Engineering, 9*(6), 639–648.

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

Banker, R. D., Charnes, A., Cooper, W. W., & Maindiratta, A. (1987). A comparison of dea and translog estimates of production functions using simulated data from a known technology. In R. Fare & A. Dogramaci (Eds.), *Applications of modern production theory: Efficiency and productivity*. Boston, MA: Kluwer Academic Publishers.

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), 1–18.

Banker, R. D., & Kemerer, C. (1989). Scale economies in new software development. *IEEE Transactions on Software Engineering, 15*(10), 416–429.

Benbasat, I., Goldstein, D. K., & Mead, M. (1987). The case research strategy in studies of information systems. *MIS Q, 11*(3), 369–386. doi:10.2307/248684.

Blackburn, J. D., Lapré, M. A., & Wassenhove, L. N. V. (2006). *Brooks' law revisited: Improving software productivity by managing complexity*. doi: 10.2139/ssrn.922768

Blackburn, D. J., Scudder, D. G., & 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.

Boehm, B. W. (1987). Improving software productivity. *IEEE Computer, 20*, 43–57.

Chatzoglou, P. D., & Soteriou, A. C. (1999). A DEA framework to assess the efficiency of the software requirements capture and analysis process. *Decision Sciences, 03*, 503–531.

Chinubhai, A. (2011). Efficiency in software development projects. *International Journal of Software Engineering and Its Applications, 5*(4), 171–179.

Cusumano, M., & Kemerer, C. F. (1990). A quantitative analysis of us and japanese practice and performance in software development. *Management Science, 36*(11), 1384–1406.

Emrouznejad, A., & Amin, G. R. (2009). DEA models for ratio data: Convexity consideration. *Applied Mathematical Modelling, 33*, 486–496.

Emrouznejad, A., & Cabanda, E. (2010). An aggregate measure of financial ratios using a multiplicative DEA model. *International Journal of Financial Services Management, 4*(2), 114–126.

Emrouznejad, A., Cabanda, E., & Gholami, R. (2010). An alternative measure of the ICT-opportunity index. *Information and Management, 47*(4), 246–254.

Emrouznejad, A. & De Witte, K. (2010). *COOPER-framework: A unified process for non-parametric projects*. Tier Working Paper Series 05(3), 1–32

Emrouznejad, A., Parker, B. R., & Tavares, G. (2008). Evaluation of research in efficiency and productivity: A survey and analysis of the first 30 years of scholarly literature in DEA. *Socio-Economic Planning Sciences, 42*(3), 151–157.

Gack, G. A. (2011). *Measuring software process efficiency*. Accessed March 25, 2013, from Process-fusion.net, http://www.compaid.com/caiinternet/ezine/Gack-Efficiency.pdf

Harter, D. E., Krishnan, M. S., & Slaughter, S. A. (2000). Effects of process maturity on quality, cycle time, and effort in software product development. *Management Science, 46*(4), 451–466.

Hong, H. K., Ha, S. H., Shin, C. K., Park, S. C., & Kim, S. H. (1999). Evaluating the efficiency of system integration projects using data envelopment analysis (DEA) and machine learning. *Expert Systems with Applications, 16*, 283–296.

Jiang, Z., & Naudé, P. (2007). An examination of the factors influencing software development effort. *International Journal of Computer and Information Engineering, 1*(3), 183–192.

Koopmans, T. C. (1951). Analysis of production as an efficient combination of activities. In T. C. Koopmans (Ed.), *Activity Analysis of Production and Allocation. Cowles Commission*. New York: Wiley.

Paradi, J. C., Reese, D. N., & Roden, D. (1997). Application of DEA to measure the efficiency of software production at two large Canadian banks. *Annals of Operation Research, 73*, 91–115.

Pendharkar, P. C., & Rodger, J. A. (2009). The relationship between software development team size and software development cost. *Communications of the ACM – Rural Engineering Development, 52*(1), 141–144.

Poel, K. J., & Schach, S. R. (1983). A software metric for cost estimation and efficiency measurement in data processing system development. *Journal of Systems and Software, 3*, 187–191.

Putnam, L. H. (1978). A general empirical solution to the macro software sizing and estimating problem. *IEEE Transactions on Software Engineering, 4*(4), 345–361.

Ramasubbu, N., & Balan, R. K. (2007). Globally distributed software development project performance: An empirical analysis. In *Proceedings of European Software Engineering Conference and the ACM SIGSOFT Symposium on the Foundations of Software Engineering* (pp. 125–134)

Rodriguez, D., Sicilia, M. A., Garciaa, E., & Harrisonb, R. (2012). Empirical findings on team size and productivity in software development. *The Journal of Systems and Software, 85*, 562–570.

Sudhakar, G. P., Farooq, A., & Patnaik, S. (2012). Measuring productivity of software development teams. *Serbian Journal of Management, 7*(1), 65–75.

Zainal, Z. (2007). Case study as a research method. *Jurnal Kemanusiaan, 9*, 1–6.