Master Thesis
Computer Science
Thesis no: MCS-2008:45
January 2009

Investigating the Nature of Relationship between Software Size and Development Effort

Sohaib Shahid Bajwa

Department of
Interaction and System Design
School of Engineering
Blekinge Institute of Technology
Box 520
SE – 372 25 Ronneby
Sweden

This thesis is submitted to the Department of Interaction and System Design, School of Engineering at Blekinge Institute of Technology in partial fulfillment of the requirements for the degree of Master of Science in Computer Science. The thesis is equivalent to 20 weeks of full time studies.

Contact Information:
Author:
Sohaib Shahid Bajwa
E-mail: sohaib.shahid.bajwa@gmail.com

University advisor:
Dr. Cigdem Gencel
Assistant Professor, Dept. of Systems and Software Engineering
SoftCenter, Ronneby
Tel: 0 457 385736

Department of
Interaction and System Design
Blekinge Institute of Technology
Box 520
SE – 372 25 Ronneby
Sweden

Internet : www.bth.se/tek
Phone : +46 457 38 50 00
Fax : + 46 457 102 45

ABSTRACT

Software effort estimation still remains a challenging and debatable research area. Most of the software effort estimation models take software size as the base input.

Among the others, Constructive Cost Model (COCOMO II) is a widely known effort estimation model. It uses Source Lines of Code (SLOC) as the software size to estimate effort. However, many problems arise while using SLOC as a size measure due to its late availability in the software life cycle. Therefore, a lot of research has been going on to identify the nature of relationship between software functional size and effort since functional size can be measured very early when the functional user requirements are available.

There are many other project related factors that were found to be affecting the effort estimation based on software size. Application Type, Programming Language, Development Type are some of them.

This thesis aims to investigate the nature of relationship between software size and development effort. It explains known effort estimation models and gives an understanding about the Function Point and Functional Size Measurement (FSM) method. Factors, affecting relationship between software size and development effort, are also identified. In the end, an effort estimation model is developed after statistical analyses.

We present the results of an empirical study which we conducted to investigate the significance of different project related factors on the relationship between functional size and effort. We used the projects data in the International Software Benchmarking Standards Group (ISBSG) dataset. We selected the projects which were measured by utilizing the Common Software Measurement International Consortium (COSMIC) Function Points. For statistical analyses, we performed step wise Analysis of Variance (ANOVA) and Analysis of Co-Variance (ANCOVA) techniques to build the multi variable models. We also performed Multiple Regression Analysis to formalize the relation.

Keywords: Software Effort Estimation, Functional Size Measurement, COSMIC Functional Size, ISBSG Data Set

ACKNOWLEDGEMENTS

First of all, I would like to thank almighty Allah who has given me an opportunity to contribute in human knowledge.

I would like to express my heartily gratitude to my supervisor Dr. Cigdem Gencel, who has always guided me and encouraged me in pursuing my thesis. As a supervisor, her support and guidance has always been a valuable asset for my thesis. Her keen interest and discussions over the work always provided me a ray of hope especially in difficult times. I believe that most of the things in my thesis would not have been possible without her professional support. Without her support, guidance and insightful comments, it would be impossible for me to perform a high quality research work. I am really grateful to her valuable contributions in my thesis.

In the end, I would like to thank my family for giving me love and support throughout this time. I am very thankful to my sister for her encouragement and motivations during my study. I also would like to express my deepest gratitude to my fiancée, for her care and support during my thesis work.

ABBREVIATIONS

SLOC	Source Lines of Codes
FP	Function Points
ISO	International Organization for Standardization
FPA	Function Point Analysis
IFPUG	International Function Point Users Groups
COSMIC	Common Software Measurement International Consortium
FFP	Full Function Points
NESMA	The Netherlands Software Metrics Users Association
FSM	Functional Size Measurement
FiSMA	The Finnish Software Metrics Association
ISBSG	International Software Benchmarking Standards Group
ANOVA	Analysis of Variance
ANCOVA	Analysis of Co-Variance
BFC	Base Functional Component
COCOMO	Constructive Cost Model
SEI	Software Engineering Institute
FUR	Functional User Requirements
CFP	COSMIC Function Points
SPSS	Statistical Packages for Social Sciences
NKLOC	Non Commented Thousands Lines of Codes
LLOC	Logical Lines of Codes
E	Entry
X	Exit
R	Read
W	Write

Table of Contents

CHAPTER 1: INTRODUCTION

Software development effort is found to be one of the worst estimated attribute over the past few years [7, 69]. It is also found that less than one quarter of the projects are estimated accurately [7]. There is a need to reliably estimate effort in order to complete the projects on time and within budget.

Software cost estimation means to predict the effort required to develop a software [5]. In software cost estimation, we predict that how much effort and time is needed to complete the software. Software cost estimation helps us to determine what resources we should use to complete the projects on time [5]. It also helps us in better software development planning [19]. It is important to note that we must know some background information about the estimate before using it [6].

Many methods have been developed for cost estimation. They are broadly categorized into algorithmic models and non-algorithmic models [5]. Algorithmic models include linear models, power function models while non algorithmic models include expert judgment etc [5].

Software size is one of the base measures which is used for software effort estimation. There are different aspects of software size measurement. Each measure has its own purpose. Software size can better be described as a set of following attributes [1]:

1. Length
2. Functionality
3. Complexity

The length of the source code of the program is measured in Source Lines of Codes (SLOC) [1]. Functionality of the software is often measured as Function Points [1]. There are many advantages of using Function Points over SLOC to estimate effort [8]. It is better to use functional size as an input for the estimation purposes instead of using SLOC.

Software size measurement remains a very hot and challenging area in the field of software engineering for many years [4]. Many problems are associated with software size measurements. Some of the main reasons for reliably measuring software size are [2, 4]:

1. Software size is a key measure that is used as an input in estimating cost and effort.
2. Planned size and actual size are compared to monitor the project achievements.
3. Contracts are also formed by using software size measurements. They are based on the cost of unit amount of software size.
4. Software size is also used in normalization of other measures.

SLOC often comes late in the software development life cycles and it is also dependent upon the technology [9]. Due to this reason, Function Points measure and the related method were developed by Albrecht in 1979 [7]. Today, five methods are certified by International Organization for Standardization (ISO) as an international standard to measure the software functional size [7]. These methods are MK II Function Point Analysis (FPA) [12], International Function Point Users Groups (IFPUG) FPA [13], Common Software Measurement International Consortium (COSMIC) Full Function Points (FFP) [14], The Netherlands Software Metrics Users Association (NESMA) Functional Size Measurement (FSM) [15] and the Finnish Software Metrics Association (FiSMA) FSM [16].

Effort estimation is always been a debatable issues between researchers in the field of software engineering [9]. It is highly desirable to make a relationship between functional size and effort to correctly estimate the effort [7]. Apart from size, there are many other factors that influence the effort to be utilized. Application Type, Architecture Type, Programming Language, Team Size, Software Quality and Platform were found to be significant [1, 9, 11, 20, 26, 30].

Function Points have some fundamental flaws in their construction that prevent them to use as a valid measures [10]. It means that they may behave in a manner that is not expected. It is highly desirable to correctly measure the Function Points to avoid any un-expected behavior. Barbara Kitchenham [10] presented a very novel idea to represent functional size. We can measure person clothing size by measuring chest size, hip size, waist size. All these three measures represent the size of a person for clothing purpose. Based on this notion, she presented an idea of representing functional size as a vector of measures [10]. Instead of a using a single value that represents functional size, it is better to use vector of measures to represent functional size.

There is a need to use accurately measured functional size for effort estimation. It is also required that we must fully understand the relationship between functional size and effort. There are many factors that affect this relationship. These factors must be discovered and their impact on the relationship between functional size and effort must be analyzed. It is also important to find out which representation of software size can better be used in effort estimation purposes [10].

The primary aim of this study is to explore the nature of the relationship between software functional size and development effort. Accordingly, the following research questions are addressed by conducting an empirical study on the International Software Benchmarking Standards Group (ISBSG) dataset r10 [41].

- What factors affect the relationship between development effort and functional size?
- Which representation of size is better for effort estimation?
- Does representing software functional size in a different way and considering the other factors improve the effort estimation reliability?

The rest of the thesis is organized as follows: Chapter 2 gives a detail definition about the problems being addressed. Research questions, and the expected contribution and goals are also given in this chapter. Chapter 3 describes the research methodologies that have been adopted to find answers of the research questions. A comprehensive literature review is provided in chapter 4. It includes software cost estimation models, software size measurement, FSM method and the relationship between functional size and effort. Chapter 5 presents a detailed comparison of different studies that has been carried out to explore the factors affecting effort-size relationship. We have discussed the empirical studies we performed in chapter 6. Finally, conclusions of this study are given in the last chapter.

CHAPTER 2: PROBLEM DEFINITION/GOALS

This chapter discusses the problems that have been addressed in this study. The research questions are presented as well. It also discusses the expected contributions and goals.

2.1 Problem Definition

The relationship between effort and functional size is not yet fully analyzed [9]. We must understand the relationship between effort and functional size so that we could be able to make reliable estimation and thus better project planning and management.

There are many projects related factors that affect software size and effort relationship at different levels. 'Application Type', 'Language Type', 'Development Type', 'Organization Type' and 'Experience of the Team Members' were found to be among the most significant ones [9, 11, 20, 26, 30]. These studies were made on the publicly available large datasets which involved detailed information on different projects.

Since these datasets, such as ISBSG dataset, have been made publicly available for benchmarking purposes and further research fairly recently, only a few studies were conducted on the nature of the relationship between functional size and effort. Most of these studies considered IFPUG FPA method as the FSM method due to the fact that most of the projects were measured by this method. Pekka Forselius [25], L. Angelis [26], Martin Shepperd [20], analyzed the factors that affect the nature of relationship between functional size and effort.

Pekka Forselius used Experience Database in Finland and considered Experience 2.0 Function Points method to find out the significant factors that affect this effort – size relationship [25]. A study by L. Angelis considered IFPUG Function Point as a base measure and ISBSG datasets release 6 to find out the significant factors for effort- size relationship [26]. Martin Shepperd also considered Experience Point 2.0 as a base for size measure to explore the relationship between functional size and effort [20]. He performed his study on Experience datasets 2004 release [20].

In [9], Gencel and Buglione, also explored the relationship between size and effort using COSMIC FFP as a base for size measure and using ISBSG datasets release 10 [9]. In [11], Gencel also suggested how we can better use COSMIC functional size for effort estimation purposes [11]. After reviewing all these studies, we do not find a single model that is accepted by the whole community. There is a need to build a model and formalize the relationship between functional size and effort for the effort estimation purposes.

Another relevant issue is which software size representation to use for effort estimation. The functional size can be represented in a number of ways. Traditionally, total functional size figures are used in effort estimation models. Representing functional size as a vector of measures is another possibility [10]. In civil engineering, we use different representations of size such as floor area (in m^2) and height (in m) or volume (in m^3) according to our needs. If we are going to build a road then we use kilometer (km) to estimate effort. We do not use this size measure in estimating effort when we are constructing a building, but we use square meter floor area and height in meters. We might also represent size in different measures for different purposes in software engineering field as well.

In [9], the impact of representing functional size as BFC Types was investigated. The results shows that we would better estimate effort if we use BFC Types instead of using total functional size in effort estimation [9].

2.2 Research Questions

The major goal of this study is to find out the different factors that play a key role in effort estimation. Following are the research questions that are derived from the discussion in the previous section:

1. What factors affect the relationship between effort and functional size?
2. What kind of representation of size can better be used for effort estimation?
3. Does representing software functional size in a different way and considering the other factors improve the effort estimation reliability?

2.3 Goals

The aim of this thesis is to better understand and improve the relationship between effort and functional size. It will also take into account the different factors that may have an effect on the effort and functional size relationship.

The expected outcome/contribution will be:

1. An understanding of the Function Points and FSM method
2. An understanding of the functional size - effort relationship
3. An understanding of the effort estimation model
4. A recommendation for the improvement of effort and functional size relationship
5. An identification of the factors that have impact on relationship between functional size and effort

CHAPTER 3: RESEARCH METHODOLOGY

This chapter presents the research methodology in detail that we have applied. It also discusses the reasons behind using these research methodologies.

In this thesis study, we have used mixed method methodologies. A mixed method methodology is a research methodology that includes both [35]:

1. Qualitative Research Methodology
2. Quantitative Research Methodology

The next sections discuss that how we have applied these methodologies in our research study.

3.1 Qualitative Research Methodology

Qualitative research methodology is used when one wants to explore any activity [35]. We have used qualitative research methodologies to explore the relationship between effort and functional size. It is used to answer the following research question.

Q. What factors affect the relationship between effort and functional size?

In the qualitative research methodologies, a through literature survey is carried out to understand the relationship between effort and functional size. The aim of the literature survey is to find out what has been given in the literature about the effort estimation. The literature also helps us to find the different studies that have been conducted to find the factors related to effort estimation. This literature survey also helps us to understand the different representation of functional size that has already been presented.

In the literature survey section, we have studied the different research papers, articles related to the identification of different factors that affect effort estimation. The results of the literature survey depict the factors that may have an effect on effort estimation. These factors were identified utilizing different statistical analysis techniques on the projects datasets available. In this thesis study, we further validate these findings with different techniques and on the large benchmarking datasets.

3.2 Quantitative Research Methodology

Quantitative research methodology is used when one wants to quantify a relationship [36]. We have the following research questions:

1. What factors affect the relationship between effort and functional size?
2. What kind of representation of size can better be used for effort estimation?
3. Does representing software functional size in a different way and considering the other factors improve the effort estimation reliability?

Our first research question is to identify the relationship between effort and functional size. Quantitative research methodology is applied when one wants to explore a cause-effect relationship [36]. Due to these characteristics of quantitative research methodology, we have used quantitative methodology to find out the significant factors. In the quantitative section, we performed empirical studies to identify the significant factors which have an influence on functional size – effort relationship. We also investigated the effect of BFC Types and total

functional size in these empirical studies. The result of the empirical study also helps us to answer research question 2 and 3.

We performed the empirical studies on the ISBSG data set release 10. We made sub sets of this data set according to our need. The sub data sets were based on the different filtrating criteria. We used Analysis of Variance (ANOVA) to find the significance of categorical variables. We further wanted to find out the significance of all the factors both categorical and numerical at ones. We used Analysis of Co-variance (ANCOVA) to find the significance of both categorical and numerical variable at ones. It is important to note that no other study consider all these factors both numerical and categorical at ones.

The results of these tests were analyzed. The result of the empirical studies is the effort estimation model the comprised of those factors that have significant affect on effort estimation.

CHAPTER 4: BACKGROUND

This chapter presents a literature review. It discusses software effort/cost estimation as well as the problems that are associated with it. A discussion about software size measurement is also presented. A comparative study about widely known software size measures, i.e. variations of Function Points; SLOC are discussed. In the end, a general over view about FSM methods is presented.

4.1 Software Effort and Cost Estimation

Software cost estimation plays a significant role in the completion of any project. Correct estimations normally lead towards the completion of the project on the scheduled time. Accurate estimation is essential so that we can properly manage the resources etc. Software cost estimation involves the estimation of the followings [5]:

1. Effort (usually in Person-Months)
2. Project Duration (in Calendar Time)
3. Cost (in Dollars)

Effort is usually measured in Person-Month. This effort estimate is then converted into dollar cost by using the following equation [5]:

Cost = (Average Salary per Unit Time of the Involved Staff) * Estimated Effort Required

Software cost estimation has been a major problem. It is difficult to reliably estimate the effort due to the following reasons [5]:

1. Lack of historical database for cost measurement
2. Factors that affect effort and productivity are not understood well. Their relation is also need to analyze.
3. Unqualified and untrained staff for estimation
4. Little penalty for the poor estimate

4.1.1 Estimation Process

Cost estimation is often done in the early phase (Planning) of the project. Following are the steps that are followed for the estimation purposes [5]:

1. Set the objectives for the cost estimation
2. Make a project plan keeping in mind the available data and resources
3. Write down the software requirements
4. Work out the complete details of the software systems
5. Use multiple cost estimation techniques
6. Compare different estimation based on different estimation techniques and iterate the process
7. Monitor the project actual cost, and feedback the result to the project management

4.1.2 Methods for Effort and Cost Estimation

Software estimation methods are classified as algorithmic models and non-algorithmic models [5]. A short overview of some of the algorithmic models and non-algorithmic models are presented next.

A) Non-Algorithmic Models

Analogy Costing: This method is based on already completed projects that are similar to the current project and estimation is done through the use of reasoning by analogy [5]. The actual costs of the previous projects are already known.

Expert Judgment: Expert judgment involves; consulting with any expert person [5]. Expert person uses his own experience and tools to estimates the project cost. Expert judgment has many problems. Some of the problems by using expert judgment for estimation purposes are [5]:

1. This approach is not repeatable and means of deriving an estimate are not implicit.
2. It is very difficult to find a highly experienced person for every new project.
3. The expert judgment is better to use when the current and past project have the same size.

Parkinson: Using this method the cost is determined by considering the available resources instead of objectives [5]. Estimation is based on the available resources.

Price-to-Win: Software cost is estimated in a way so that company would easily win the project [5]. Customer budget is keeping in mind as compared to functionality of the software.

Bottom Up: Each component is estimated separately and the sum of each of the estimation of the component is the total estimation required for the project [5].

Top Down: Top down approach is the opposite of the bottom up approach. An overall cost of the project is estimated and then this cost is divided into different components of the software [5].

B) Algorithmic Models

Algorithmic methods are those models that are based on some mathematical models and produce cost estimation as a function of a number of variables [5]. Different variables that affect the estimation are considered. These variables are often called as 'Cost Drivers' [5].

Many quantitative software cost estimation models have been developed [5]. They are mainly categorized into the following categories:

- **Analytical Models:** An analytical model uses data from the previous project in order to evaluate the current project [5].

- **Empirical Models:** An empirical model is based on some global assumption [5]. It uses global assumptions to estimate the effort required in the completion of the project. Some of the global assumptions are [5]:

    i. Developer rate to solve the problems
    ii. Numbers of problem available

Following are the details of some of the widely-known algorithmic models.

Constructive Cost Model (COCOMO): The COCOMO model is proposed by Boehm [5]. The general form of the COCOMO model is [5]:

$$\text{Effort} = a * \text{Size}^{b}$$ where a, b are the cost factors/function of other cost factors.

An evolution history of COCOMO models is presented below:

1) Basic COCOMO
The value of a, b are measured according to the complexity level of the project [5].

Table 4.1. Application Types with a, b Values (adapted from [5])

Application Type	Value of a	Value of b
Simple Application	2.4	1.05
Complex System	3.0	1.15
Embedded System	3.6	1.20

The basic COCOMO model is not difficult. It is relatively simple and easy to use. It is better to use this method as rough estimate because many cost factors are not considered [5].

II) Intermediate and Detailed COCOMO
The value of a, b are slightly different than that of basic COCOMO model.

Table 4.2. Application Types with a, b Values (adapted from [5])

Application Type	Value of a	Value of b
Simple Application	3.2	1.05
Complex System	3.0	1.15
Embedded System	2.8	1.20

There are additional cost factors whose impact is calculated to calculate the estimated effort. Following are the cost factors [5]:

1. Product
    a. Required Software Reliability
    b. Database Size
    c. Product Complexity
2. Computer
    a. Execution Time Constraint
    b. Main Storage Constraint
    c. Virtual Machine Volatility
    d. Computer Turnaround Time
3. Personnel
    a. Analyst Capability
    b. Programmer Capability
    c. Application Experience
    d. Virtual Machine Experience
    e. Language Experience
4. Project
    a. Software Tools
    b. Modern Programming Practices
    c. Development Schedule

III) COCOMO II
The difference between early cost models and COCOMO II based on the change in factor b, that changes according to the following cost factors [5]:

1- Development Flexibility
2- Architecture or Risk Resolution
3- Team Cohesion
4- Process Maturity

The main aim of the development of the COCOMO II model is to provide support for different activities e.g. project planning, cost estimation [17].

Putnam's Model: This model is based on the Norden/Rayleigh manpower distribution and his findings after investigating different completed projects [5]. SLIM is a tool based on the Putnam's model [5]. This tool is used for cost estimation and man power scheduling [5].

Other Models: SoftCost and Price-S are the two other models that are used for effort estimation [5]. SoftCost model relates size, effort and duration to address risk and also contains heuristics to help estimators in dealing with the new technology [5]. Price-S is developed by RCA New Jersey [5]. It also computes effort, project cost and schedule [5].

4.1.3 Comparison of Estimation Models

The strengths and weaknesses of different models are given below:

Table 4.3. Comparison of Estimation Models (adapted from [5])

Method	Strengths	Weaknesses	Algorithmic/Non-Algorithmic
Expert Judgment	Fast estimation, Expert having experience provide good estimation.	Dependent upon expert, Biasness	Non-Algorithmic
Analogy	Bases on actual project data and past experience	Inaccuracy in historical data, similar project may not be available	Non-Algorithmic
Parkinson, Price to Win	Good to win the contract	Poor practice, Large overruns	Non-Algorithmic
Top Down	Minimal requirement	Less accurate than other models	Non-Algorithmic
Bottom Up	Based on detailed analysis, Better project tracking as compared to other models	Need more estimation effort as compared to top down, not suitable for early estimation	Non-Algorithmic
Algorithmic	Objective Repeatable results, Better	Subjective Inputs may not reflect the current	Algorithmic

	understanding of estimated methods	environment, Company specific algorithmic model, may not be generalize	

4.1.4 Why Accurate Estimation is Difficult?

Currently, none of these models are accepted by the whole community since none of them performs reliable-enough in different environments. [5]. We do not have a generalized model that we can apply in different environments of various organizations. Many factors and issues that affect the estimation are not discovered. Some of the reasons of not creating any best model for estimation are [5]:

1- Development environment does not remain the same and is evolving continuously.
2- We are unable to get the measures that reflect the complexity of the system.
3- There are many interrelated factors that affect the project development process. These factors must be discovered and their impact on the effort estimation must be analyzed.

4.2 Productivity

Another related attribute to effort estimation is 'Productivity'. Productivity can be defined as [1]:

Productivity = Size/Effort

Where

Size; measured in SLOC (or any other size measure)
Effort; measured in Person-Months

Many people are involved in a software projects. These people have different backgrounds and different set of skills. They have different experience. All these diversifications make it difficult to compare different projects and its effects on the effort to be utilized for developing a specific product. However, it is possible to find average productivity figures for a specific organization or for a specific project. If average productivity is known, it is also possible to estimate the effort when the software size figure is available.

However, there are many problems related to the above equation. One of the problems is related to effort data collection mechanisms. It has been observed that many projects, that are completed, are unable to exhibit effort [1]. Usually, we do not have the correct calculation for the total effort utilized even for the completed projects. There are many people involved in a project, their effort is ignored while calculating the total effort. Contribution from corporate staff (researchers) and support staff are usually not calculated while measuring the total effort for different projects [1].

The size in the productivity equation also creates the problems.. In this equation, the input is the effort whereas the output is the size of the product. However, there is also at least one attribute of the product, which is the quality of the product [1]. It is important to take into account the quality of output while calculating productivity. Because, most of the time, productivity increases if quality is not considered.

Productivity equation which uses SLOC as a size measure, leads to the following issues [1, 20]:

1. Different programming languages have different expressiveness style. A certain piece of code written in C# cannot be equal to COBOL, although both codes implement the same functionality.
2. Every programmer has its own coding style.
3. Most software engineering activities do not directly involve code.
4. We must count the comments and reused code separately.

Functional size (in Function Points) has been increasingly used as the size figure to calculate the productivity. [1]:

Productivity = Function Points /Effort

Productivity is usually calculated when we have some sort of output. There are many people involved in a project whom output is not clear e.g. project managers, testing teams, quality assurance staff etc [1].

Following are the factors that may affect the productivity [1]:

A) Team Structure
It is generally understood that more complex team structure leads to low staff productivity [1]. It's not true that the most people in a team, the more productive the work is. There are many reasons for this low staff team productivity in a complex team structure size. Communication problems are one of the reasons.

B) Personal Experience
Person experience also plays very important role in the productivity. We must take into account the personal experience while measuring productivity.

C) Tools
The development environment and tools also play significant role in measuring productivity. The productivity of a team will be different from that team who has already familiar with the tools and development environments.

4.3 Software Size Measurement

Size is one of the key attributes of software. It is very important to accurately measure the size of the software. It helps us to understand how big the software is? There are many project related questions that demand that size of software should be measured. Size of software is important for the following reasons [22, 23]:

1. It shows that how big is the software.
2. We can measure effort by using size. We can estimate that how much effort is more required to complete the project.
3. What is the quality of our project as compared to other projects?
4. Software size is helpful in benchmarking the productivity. What is our productivity as compared to others?
5. The information about the size of the software is very important in the contract management. Based on the software size, contracts are established with the customers.

Software size measurement plays a significant role to estimate effort and productivity. Accurately measured software size helps to make more accurate effort estimation. There are different ways to measure software size. Following are the two views to measure software size [22]:

1. Physical measurement of software
2. Measuring the functionality of software

4.3.1 Physical Measurement of Software

Different measures are defined to measure the length of different software products such as code, specification documents or design.

A) Measuring the Length of Code

It is very simple to measure the SLOC of software. Measuring SLOC is also reliable [22]. There are many tools available that we can use to measure the SLOC. The main issue with the counting of the SLOC is the rules that you are following [22]. Different companies provide different counting techniques to count the SLOC. These counting rules are different from each other. Software Engineering Institute (SEI) has published a framework that can be use to count the SLOC [22]. This framework contains checklist and other rules to count the SLOC. An organization may have its own counting rules to count the SLOC. The most popular rules to count the SLOC that many organizations using are [22]:

1. NKLOC (Non Commented Thousands Lines of Codes)
2. LLOC (Logical Lines of Codes)

Some of the issues that are related with SLOC are [22]:

1. Measuring software size as a SLOC does not represent the functionality of the software.
2. It does not tell us that how complex the software is.
3. It also does not tell us about the technology that we are using.

Above problems that are associated with SLOC do not mean that SLOC is not a valid measure. It is important to use the appropriate measure for the appropriate purpose. Although SLOC is not a valid measure to estimate effort, but it's a good measure to estimate how much the file size will be [22]. We must not reject this measure but use this measure appropriately where it is needed.

B) Measuring the Length of Specification and Design

We can measure the length of specification or design. This measurement can be used to predict the SLOC or effort [22]. Length of specification or length of design can be a good measure to estimate SLOC if you measure them accurately. Following are the two measures that are used to measure design and specification [22]:

1. Number of Pages
2. Number of "Shalls"

We estimate SLOC by using either number of pages or by calculating the number of shall statement in a specification. We use a co-relation between designs to code or specification to code to estimate SLOC [22]. It is better to use these techniques in a well established and disciplined organization that has consistent style [22]. If any organization does not record information from previous project and does not have the previous design-code ratio then these techniques may not work.

4.3.2 Measuring Functionality (Function Points)

Functionality of software can be represented by calculating Function Points. Function Points reflect the functionality that the software must exhibits [1]. Function Points measure is used to represent the amount of functionality in software. It is important to note that functional size is the only standardized way to measure the software size [23]. This method is independent of the development tools. It is also independent of the technical requirements of the software.

A functional size must exhibits the following characteristics [24]:

1. It is not derived from the effort that is needed to develop or support particular software.
2. It is not dependent upon the methods that are used to develop the software.
3. It is not dependent upon the methods that are used to support the software.
4. It is not dependent upon the physical components of the software.
5. It is also not dependent upon the technical component of the software.

Measuring size by calculating the Function Points have the following advantages [1, 22, 23]:

1. Function Points based size measurement more correctly and accurately represents the value of the output.
2. Function Points is used to measure the productivity of the software development staff at any stage of life cycles.
3. We can also find the progress of our project. We can compare the completed and implemented Function Points with the incomplete ones. It helps to analyze where we stand and how much time and effort is more required to complete the project on scheduled time.
4. Function Points are not dependent upon any technology.
5. It is very effective measure for effort estimation as we measure Function Points in early phases.
6. Function Points are normally well documented and specified.
7. International standard for Function Points are available.
8. Function Points are helpful in building contract and negations between managers and users.
9. Function Points provides a ways to understand the effort.
10. It helps us in better management of projects.

FSM methods play a significant role in performing different tasks related to the software development and project management. We can use these methods to find out many answers about different project related questions. These methods are helpful in a number of ways e.g. [24]:

1. Finding the current progress of the project
2. Scope management
3. Performance management
4. Forecasting
5. Measuring and managing productivity
6. Managing quality requirements of the software
7. Resource required for project
8. Budgeting for support and maintenance
9. Managing contracts

Following are the issues that are associated with the Function Points [22, 23]:

1. Function Points are difficult to understand especially the terms and components that are related with Function Points.
2. Calculating Function Points is a lengthy process. It has many steps.
3. Automatic calculation of Function Points with the help of tools is not available.
4. Measuring functional size also requires training.

4.3.3 Function Points vs. SLOC

Following are the advantages of using Function Points over SLOC in effort estimation [8]:

1. Software cost estimation often takes place early in the project life cycles. It is very difficult to estimate the code size of the software so early in the project.
2. It is easy for non technical members of the project to estimate the Functions Points as compared to SLOC.
3. Functions Points are language independent and are independent of limitation differences as well.

Due to the above mentioned advantages of Function Points over SLOC, it is better to use functional size measured in Function Points as an input for the estimation purposes instead of using SLOC.

4.4 Functional Size Measurement (FSM) Methods

There are many significant attempts taken to measure the software size. Various methods and approaches have been developed to measure the software size. These methods are based on different metrics. These methods use the SLOC and other technical characteristics to measure the software size [24]. The methods that are based on the technical characteristics and SLOC have the following limitations [24]:

1. These methods cannot always be applied in the early phases of software development life cycles.
2. These methods cannot always be understood by the user of the software.

To overcome above mentioned limitations, methods that are not based on SLOC have been proposed. Most of the methods that are used today to measure the size of the software are based upon the “Functionality” of the software [7]. These methods measure the size of the software by measuring the functionality that it provides to the customer. These methods have shifted the focus from measuring the technical characteristics of the software towards measuring the functionality of the software that is required by the intended users of the software.

4.4.1 Evolution of FSM Methods

The first method that calculates the functionality of the software is designed in 1979 by Albrecht [42]. This method is based upon the functionality of the software. This is the first FPA method that is based upon the Function Points [7]. A detailed evolution of FSM methods is presented next:

Table 4.4. Evolution of FSM Methods (adapted from [7])

Year	Method Name	Developer
1979	Albrecht FPA/IFPUG FPA	[42, 43, 44]/ International Function Point Users Group (IFPUG) [45, 46]
1982	DeMarco's Bang Metrics	DeMarco [47]
1986	Feature Points	Jones [48]
1988	MK II FPA	Symons [49], The United Kingdom Software Metrics Association (UKSMA) [12, 50]
1990	NESMA FPA	The Netherlands Software Metrics Users Association (NESMA) [51, 52]
1990	Asset-R	Reifer [53]
1992	3-D FP	Whitmire [54]
1994	Object Points	Banker et al [55].Kauffman and Kumar [56]
1994	FP by Matson, Barret and Mellichamp	Matson et al. [57]
1997	Full Function Points (FFP)	University of Quebec in cooperation with the Software Engineering Lab. in Applied Metrics [58]
1997	Early FPA (EFPA)	Meli [59, 60], Conte et al. [61]
1998	Object Oriented FP	Caldiera et al. [62]
1999	Predictive OP	Teologlou [63]
1999	COSMIC FFP	The Common Software Measurement Consortium (COSMIC) [64,65]
2000	Early and Quick COSMIC FFP	Meli et al [66], Conte et al. [61]
2000	Kammelar's Component OP	Kammelar [67]
2001	Object Oriented Method FP	Pastor et al. [68]
2004	FiSMA FSM	The Finnish Software Metrics Association (FiSMA) [3]

Many methods to measure functionality have been developed since 1979. Evolution of these methods results inconsistencies among different methods [7]. It is important to make standard for these methods to end the inconsistencies between them. Many attempts have been taken to resolve these inconsistencies. In 1998, ISO defined the standard for the development of these methods [7]. This ISO standard defines components that must be in FSM methods. These components are [7]:

1. Functional User Requirements (FURs)
2. Functional Size
3. Base Functional Components (BFC)
4. BFC Types

5. FSM Requirements

Table 4.5. Concepts, FSM Methods and Description (adapted from [7, 24])

Concepts	Description	Example
FURs	It represents the user requirements that the software must fulfill.	Maintain Customer
Software/Functional Size	Size of the software driven by quantifying the FUR	20 Function Points
BFC	An elementary unit of FUR	Add a new customer
BFC Type	It is a defined category of BFC	External Input, External Output

After defining the standard, following are the methods that are ISO certified [21]:

1. Mk II Function Point Analysis (FPA) [12]
2. International Function Point Users Groups (IFPUG) FPA [13]
3. Common Software Measurement International Consortium (COSMIC) Full Function Points (FFP) [14]
4. The Netherlands Software Metrics Users Association (NESMA) Functional Size Measurement (FSM) [15]
5. The Finnish Software Metrics Association (FiSMA) FSM [16]

CHAPTER 5: THEORETICAL WORK

This chapter discusses the relationship between functional size and effort. It also discusses the studies that have been carried out over the past few years to see the significance of different factors affecting size-effort relationship. In the end, a detailed comparative analysis of these studies is presented.

5.1 The Relationship between Functional Size and Effort

Software cost and effort estimation plays a significant role in the successful completion of any software. Resources are assigned according to the effort required to complete the software. Accurate effort estimation leads the completion of software project on the scheduled time. Many models and approaches have been developed over the time to estimate the effort. Most of the models take software size as a basic input to estimate the effort [7]. We have already discussed that it is better to use functional size instead of SLOC to estimate effort. Effort is most of the time calculated by using functional size of the software. There is a strong relationship between functional size and effort. Correctly measured functional size is helpful in estimating effort. It is the need of hour to correctly establish a relationship between functional size and effort so that we could be able to estimate effort accurately.

There are many other project related factors that affect positively or negatively this relationship. Application Type, Language Type, Development Type, Organization Type, Experience of the Team Members are some of them [11]. It is also important to note that BFC Types also affect the relationship between functional size and effort [9].

Many significant attempts have been taken to explore the relationship between Functional Points and Effort. COCOMO II model is one of the widely known models in the software industry. Although it uses Function Points to estimate effort, but there are many problems that are associated with it. COCOMO II model converts the Function Points into SLOC which is not a reliable conversion [18]. A wrong conversion that is used in the estimation purposes in COCOMO II will not give sufficient results. An improper conversion of SLOC/FP can be considered an error while estimating the effort [18]. It is not a good idea to estimate effort based upon the wrong conversion of SLOC/FP.

Different representations of functional size are also considered to estimate effort. A better representation of functional size also improves the effort estimation. In [11], Gencel suggested a better use of COSMIC Function Point that will be helpful in better effort estimation. An estimation models based upon the correct representation of functional size may be a good model for the estimation purposes.

5.1.1 The Factors Affecting Size-Effort Relationship

Different studies have been performed over the past few years to investigate the relations between software size and effort. These studies are present below:

Study by Pekka Forselius: A study in the Finland to explore the factors that affect productivity and effort estimation shows the following results:

Table 5.1. Factors Affecting Productivity by Pekka Forselius (adapted from [25])

Data Set	Factors	Significant Factors	Base of Size Measurement
Experience Database	Application Programming Language, Application Type (MIS etc), Hardware Platform, User Interface, Development Model, DBMS Architecture, DB Centralization, Software Centralization, DBMS Tools, Case Cools, Operating System, Company where project was developed, Business Sector (Banking, Insurance etc), Customer Participation, Staff Availability, Standard Use, Method Use, Tool Use, Software Logical Complexity, Requirement Volatility, Quality Requirement, Efficiency Requirement, Installation Requirement, Staff's Analysis Skills, Staff's Tools Skills, Staff's Team Skills, Staff's Application Knowledge	Customer Participation, Staff Availability, Standard Use, Tools Use, Logical Complexity, Requirement Volatility, Efficiency Requirements, Staff's Tools Skills,	Experience 2.0 Function Point Method

Risk is also another important cost factor. According to Forselius, although risk is not included in the process of effort estimation, but it must be taken into account into the project contracts [23].

Study by L. Angelis: L. Angelis has also made important contribution towards finding the different factors that affect size and effort relationship. The result of his study is given below [26]:

Table 5.2. Factors Affecting Productivity by L. Angelis

Data Set	Factors	Significant Factors	Base of Size Measurement
ISBSG release 6	1. Development Type 2. Development Platform 3. Language Type 4. Used Methodology 5. Organization Type 6. Business Area Type 7. Application Type	1. Development Type 2. Development Platform 3. Language Type 4. Organization Type 5. Business Area Type 6. Application Type	IFPUG Function Point

It is important to note that the author (L. Angelis) has studied the impact to form homogenous groups. He has made the categorization based on the values of the factors [26]. The impact of Team Structure and Mature Methodologies is also required to explore [26].

Human are the very important factors that are not taken while performing any study. A recent study shows that Psychometrics should be collected to better perform the empirical study [27].

Study by Martin Shepperd: According to Shepperd, following factors affect the productivity [20]:

1. Task Difficulty
2. Skills of Project Team
3. Level of Non Functional Requirement
4. Ease of Communication with Client

The following table presents different factors that affect productivity [20].

Table 5.3. Factors Affecting Productivity by Martin Shepperd

Data Set	Significant Factors	Base for Size Measurement
Experience Data Set 2004 release	1. Company 2. Process Model 3. Business Sector 4. Year 5. Hardware	Experience Point 2.0 (A variant of Function Point)

The impact of Hardware and Year on the productivity need to be explored [20]. Process Models and Business Sector along with company has an impact on the productivity. It is important to explore that how hardware affects the productivity. The above results are based

on experience data set. The Martin Shepperd suggests that we must generalize his results [20].

Another study in by Shepperd exhibits the following results [28]:

Table 5.4. Factors Affecting Productivity by Martin Shepperd

Data Set	Significant Factors	Base for Size Measurement
25,000 closed projects of a large multinational company	1. The Degree of Technical Innovation, Business Innovation, Application Innovation, 2. Team Complexity 3. Client Complexity 4. Degree of Concurrency 5. Development Team Degree of Experience With Tools, Information Technology, Hardware, or With Adopted Methodology, 6. The Project Management Experience	Function Point

Different industry sectors exhibit the differences in the productivity. It is due to the fact that industry sectors also affect the productivity [28].

Other Studies: A study in on the different Swedish companies shows that following factors affect the estimation [29]:

1. Requirement Volatility (Unclear and Changing Requirement)
2. Unavailability of Templates
3. Lack of coordination between product developed and other parts of the project

Following factors that are considered important from ISBSG data repository, also affect the productivity [30]:

1. Programming Language
2. Team Size
3. Organization Type
4. Application Type

Another study by Barry Boehm shows the following results [31]:

Table 5.5. Factors Affecting Phase Distribution for Software Development Effort

Data Set	Factors	Base for size measurement
China Software Benchmarking Standard Group	1. Development Life Cycle 2. Development Size 3. Software Size 4. Team Size	Length of Code

Detailed comparisons of the different factors that have been identified in the previous studies are presented below:

Table 5.6. Factors Affecting Size-Effort Relationship

Common Factors in Different Studies	Factors Available in ISBSG Whose Affect Need to be Analyzed
1. Application Experience 2. Tools Use and Experience 3. Platform Experience 4. Application Type 5. Development Type 6. Language 7. Business Area 8. Requirement Volatility	1. Productivity 2. Quality (Defects) 3. Development Type 4. Business area Type 5. Application Type 6. Architecture 7. Development Technique 8. Development Platform 9. Language 10. Hardware 11. Team size 12. Year

The significance of the following factors on the functional size – effort relationship is also need to be explored:

1. Requirement Volatility
2. Team Size
3. Team Structure
4. Used Methodologies
5. Quality and understandability of different specification e.g. Requirement Specification etc.

Following are the COCOMO II cost factors whose impact are not yet discovered in the context of Functional Size – Effort relationship:

1. Required Software Reliability
2. Database Size
3. Product Complexity
4. Required Reusability
5. Documentation Match to Life Cycle Needs
6. Execution Time Constraint

7. Main Storage Constraint
8. Platform Volatility
9. Analyst Capability
10. Programmer Capability
11. Personnel Continuity
12. Multisite Development
13. Required Development Schedule

We can analyze the affect of different factors on functional size and effort relationship by using ISBSG data set. ISBSG contains information about a large number of projects [30]. It also contains information about many project related attributes [30]. A detailed discussion about ISBSG datasets is presented in chapter 6.

CHAPTER 6: EMPIRICAL STUDY

This chapter discusses the empirical studies we conducted in order to investigate the significance of different project related factors on the relationship between functional size and development effort. We also investigated that which representation of software functional size (CFP or BFC Types) would better estimate effort. For empirical studies, we selected the projects measured by COSMIC FFP.

In COSMIC FFP [14], the FURs are composed into their elementary components. These elementary components are called Functional Process [11]. A Functional Process is an elementary component of a set of FUR. It comprises of a unique, cohesive and independently executable set of data movement types [11]. There are four kinds of Data Movement Types. These are Entry, Exit, Read and Write. These are also defined as a BFC Type. The definitions of BFC Types are presented below [11]:

- **Entry (E):** It moves a data group from functional user across the boundary into the functional process that requires it.
- **Exit (X):** It moves a data group from functional process across the boundary to the functional user that requires it.
- **Read (R):** It moves a data group from persistent storage within reach of functional process.
- **Write (W):** It moves a data group from functional process to persistent storage.

Following steps are carried out to calculate the Function Points [11]:

1. Identification of BFC Types in Functional Process
2. Calculating the functional size of each BFC by applying a measurement function to the BFC Types
3. Aggregating the results to compute the overall size of the software

We have divided empirical study in two phases: Phase 1 and Phase 2. Each phase is discussed separately.

6.1 Data Collection

We used data from the ISBSG. ISBSG maintains a repository of data about different completed projects [30]. This data is collected from numerous organizations all over the world. The metrics associations from the United States, United Kingdom, Japan, Australia, Italy, Germany, Netherlands, Spain and Italy are the members of this non-profit organization [30]. There are some other countries as well that also work with ISBSG. United States, Japan, Australia and Finland are the major countries that provide project related data to ISBSG [33]. There are different organizations in those countries that provide data about completed projects. These datasets contains different project related attribute e.g. Language Type, Platform, Team Size, Architecture, Tools, Development Techniques etc. It is also important to note that these projects come from those companies whose processes are mature so that they can include software metrics program [30]. This repository does not represent all the software industry but it covers a broad cross section of software industry. This repository contains data about finished projects only.

There are many uses of this data repository. We can use this for different purposes. Some of the uses of this data repository are [33, 34]:

1. Project Benchmarking
2. Best Practice Networking
3. Summary Analyses
4. Help estimation of own future software development projects
5. Assist in evaluating the benefits of changing hardware or software environment
6. Software development and engineering research
7. For academic research with the objective of improvement in different IT practices

6.1.1 Characteristics of ISBSG

Some of the important characteristics about this data set release 10 are given below [33]:

1. The projects have been submitted from 22 different countries. Major contributors are:
    - United States (27%)
    - Japan (20%)
    - Australia (20%)
2. Major types of organizations, who provide data, are:
    - Communication (22%)
    - Insurance (17%)
    - Business Service (12%)
3. Telecommunication (25%), Banking (13) and Insurance (13%) are the major business area.
4. 30% of the total projects have fewer than 100 function points. 21% have 100-200 function points. 13% have 200-300 function points. The average function points are approximately 200 function points.

6.1.2 Variables Definition and Values

Some of the variables of ISBSG that are most relevant to our study are presented below [32]:

Data Quality Rating: This field contains an ISBSG rating code. This code can be A, B, C or D. These are the Data Quality Rating. This rating is applied to the project data after reviewing by the quality personals.

1. **A** = the data was sound and there is no threat to data integrity.
2. **B** = the submitted data is sound but there may be some factors that affect the data integrity.
3. **C** = the significant data is not provided, so we cannot say anything about the integrity of the data.
4. **D** = the data has a very limited credibility due to many factors.

Count Approach: It contains the description of the technique that we use to size the project. Most of the projects in the ISBSG data repository are measured using FSM method based on functional size e.g. IFPUG, NESMA, COSMIC-FFP, FiSMA

Functional Size: It is the count of unadjusted function point. The unit is based on the measurement method that is used to measure the functional size.

Normalized Work Effort: It is the effort that is used in full life cycle. For those projects that have covered less than a complete life cycle effort, this value is an estimate. For those

projects covering the full life cycle and those projects whose development life cycle coverage is not known, this value and value of summary work effort is same.

Summary work effort: It provides the total effort recorded against projects.

Quality (Total Defects Delivered): The number of defects detected in the process in that particular Effort Breakdown or found within the first month of use of the software after implementation.

Development Type: This field tells that whether the development is new, enhanced or re developed.

Organization Type: It is the type of the organization that submitted the projects. It can be manufacturing, banking, retail etc.

Business Type: It is the subset of the organization type. Business type may be same as organization type but it can be different as well e.g. manufacturing, finance etc.

Application Type: It is the type of the application. It can be Management Information System (MIS), Transaction Production System (TPS) etc.

Architecture: It indicates whether the application is stand alone, client server, multi tier or multi tier with public interface.

Development Platform: It defines the primary platform on which the application is developed. It can have values PC, mid range, main frame or multi platform.

Language Type: It defines the language type that is used for the projects. It can be 2GL, 3GL etc.

1st Hardware: It defines the primary hardware that is used to develop or enhance the projects.

Max. Team Size: It defines the maximum number of people that have worked as a team on any project.

Year: It is the implementation date of the project. If implementation date is not available then it is derived from project start date, end date, implementation date, data compilation date. If no project date is available then it is the year when the project data is received.

Entry: Functional size of number of entries

Exit: Functional size of number of exits

Read: Functional size of number of reads

Write: Functional size of number of writes

We have used Normalized Work Effort to perform statistical analyses. We have created a new variable called normalized defects. It is defined as:

Normalized Defects: It is the ratio of Total Defects Delivered and Functional Size. It is denoted by Total Defects Delivered/CFP.

6.2 Statistical Methods and Tools Used

In this study, we used Linear Regression Analysis, Multiple Regression Analysis, Analysis of Variance (ANOVA), and Analysis of Covariance (ANCOVA) to perform statistical analyses. A Step Wise ANOVA technique is used to build multi variable model in Empirical Study: Phase 2 (see section 6.4).

We used Microsoft Office Excel 2007 and Statistical Packages for Social Sciences 17.0 (SPSS) in performing these statistical analyses.

6.2.1 Simple Linear and Multiple Regression Analysis

A Regression Analysis is used to see the effect of independent variables on the dependent variable [39]. The purpose is to see that how much dependent variable is dependent upon the independent variables. We change the value of independent variable and see the resulting change in dependent variable. The goal is to identify at what extent dependent variable is defined by using independent variable.

In a Simple Linear Regression [39], we have one independent and one dependent variable [39]. Mathematically, it can be written as [39]:

$Y = a + bX + C$

Where

Y: Dependent variable
X: Independent variable
b: Coefficient of variable X
a: Y intercept
C: Constant

In a Multiple Linear Regression Analysis [39], more than one independent variable is used to describe the change in dependent variable [39]. The Multiple Regression Model for the n independent variables is:

$Y = a + b_1X_1 + b_2X_2 + b_3X_3 \ldots + b_nX_n + C$

Y: Dependent variable
$X_1, X_2, X_3 \ldots X_n$: n Independent variable
$B_1, B_2, B_3 \ldots B_n$: coefficient of variables X_1, X_2, X_3 to X_n
a: Y intercept
C: Constant

The significance level is usually .05 for the Regression Analysis. If the significance value is > 0.05 then our result would not significant [39].

6.2.2 Analysis of Variance (ANOVA)

ANOVA [38] is a broad class of technique that is used for measuring the variation within a data collection [38]. This technique is used to identify the relation between dependent variable and independent variable. The dependent variable may be either numerical or categorical. The independent variable is categorical [38]. The variation in the means of the independent variable means that it has a significant relation with dependent variable. The significance level is usually .05. If the significance value is > 0.05 then our result would not significant. ANOVA has following variation [38]:

1. One Way ANOVA (One dependent and one independent variable)
2. Factorial ANOVA (One dependent and more than one independent variable)

6.2.3 Analysis of Covariance (ANCOVA)

ANCOVA [38] is a combination of Regression Analysis and Analysis of Variance (ANOVA) [38]. It is applied when we have non categorical dependent variable and both categorical and non categorical independent variables [38]. The significance level is 0.05. If the significance level is greater than 0.05 then the results are not significant otherwise results are significant.

Following table give a guideline to select the appropriate statistical method to perform analysis [38]:

Table 6.1. Statistical Methods Selection Criteria

Statistical Methods	Dependent Variables	Independent variables
ANOVA	Non-Categorical	Categorical
ANCOVA	Non-Categorical	Both Categorical and non Categorical

6.2.4 Step Wise ANOVA

We have also used the Step Wise ANOVA [37] technique to build the multi variable model in Empirical Study: Phase 2. This technique determines the influence of both numerical (non-categorical) and categorical independent variable on the dependent variable effort [37]. This technique requires lots of effort from the person who is making analysis. This technique work as follows [37]:

1. The model is empty at the start and then the variables that are related to effort are added one by one.
2. This process stops when no added variable would improve the model.

6.2.5 Microsoft Office Excel 2007

There are many statistical methods available in Microsoft Office Excel 2007. Single Factor ANOVA, Linear and Multiple Regression Analysis etc are some of them. We used this tool to perform analysis in Empirical Study: Phase 1 (see section 6.3).

6.2.6 Statistical Package for Social Sciences 17.0 (SPSS)

Statistical Package for Social Sciences 17.0 (SPSS) [40] is a very comprehensive tool that provides a number of statistical methods to perform the analysis. We used SPSS 17.0 to perform further analysis in Empirical Study: Phase 2 (see section 6.4).

6.3 Empirical Studies: Phase 1

In this phase of the thesis study, by conducting a number of empirical studies on ISBSG dataset, we investigated the significance of factors that were found to be significant in estimating the Development Effort in various studies.

We used data sets of ISBSG 2007, Release 10. Different sub-data sets were formed after filtering the data based on the cost drivers. The reason behind using different data sets was that we had many missing values in our data repository (ISBSG). Further statistical analyses were carried out when there remained enough data points to perform the statistical analyses.

We performed one way ANOVA, Regression Analysis and Multiple Regression Analysis to observe the effect of different factors on effort estimation. We made our homogenous data sets based on the significant factors and see that whether it would improve the reliability of effort estimation or not. We also investigated whether using the sizes of BFC Types rather than total functional size would improve the effort estimation or not.

We performed six empirical studies on the ISBSG dataset by forming homogenous sub-sets of data. The details of the studies are discussed in the following sections.

6.3.1 Empirical Study 1

First, we performed the following statistical analyses using the ISBSG repository:

1. Regression Analysis between Development Effort and Total Size
2. Multiple Regression Analysis between Development Effort and the sizes of BFC Types

The reason why we performed both was to investigate whether estimation reliability increases if we take into account the size of each BFC Type rather than the total size figure.

A) Data Preparation

Following steps are carried out to prepare the data (see Table 6.2):

Step1 – There are total 4106 projects in ISBSG repository. We selected those projects that have high Data Quality rating (see Section 6.1.2. for the definition of Data Quality rating). We removed projects having either C or D Data Quality Rating. The selected projects have either A or B Data Quality Rating, which are in total 3811 projects..

Step2 – Next step is to select only those projects that have 'Count Approach' COSMIC FFP. The number of the remaining projects is 110.

Step3 – The third step is to remove those projects whose BFC Types (Entry, Exit, Read, and Write) are not reported. We have now remaining 90 projects.

Table 6.2. Filtration of ISBSG Dataset 2007 Release 10

STEPS	ATTRIBUTES	FILTER	PROJECT EXCLUDED	PROJECTS REMAINING
1.	DATA QUALITY RATING	= A OR B	295	3811
2.	COUNT APPROACH	= COSMIC-FFP	3701	110
3.	SIZE ATTRIBUTES (COSMIC ENTRY, EXIT, READ, WRITE)	≠ EMPTY (DATA IS NOT REPORTED)	20	90

B) Statistical Analysis & Results

Fig. 6.1. shows a relationship between Normalized Work Effort and COSMIC Functional Size.

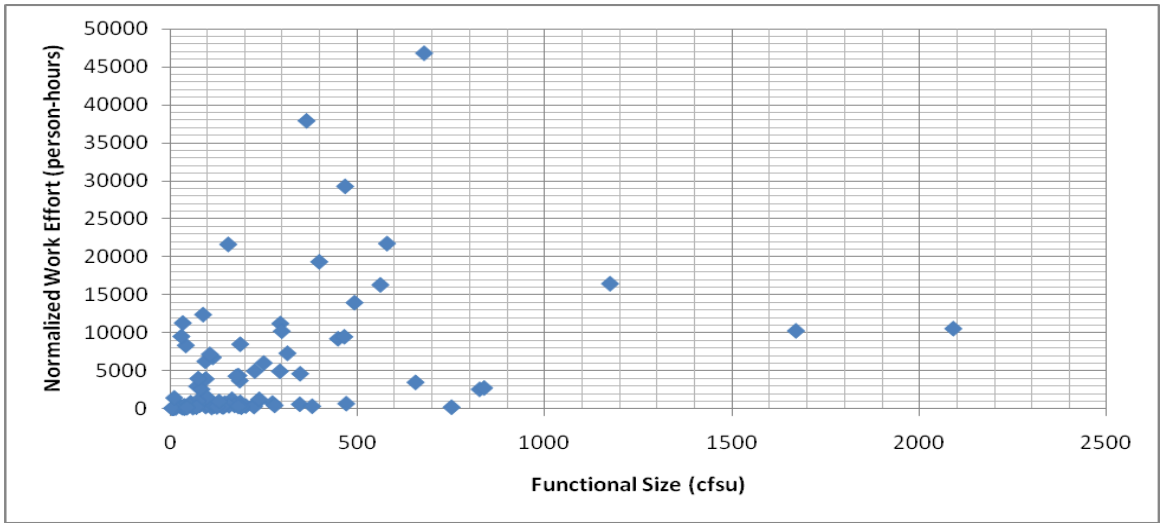

Fig. 6.1.The Relationship between Normalized Work Effort and COSMIC Functional Size

Table 6.3 shows a Linear Regression Analysis between Development Effort and Size. The total result is significant because the significance F value < 0.05. The coefficient Size is also significant because p-value < 0.05. There is a very low adjusted R^2 value (.13) between Development Effort and Size which shows a weak relationship.

Table 6.3. Linear Regression Analysis between Development Effort and Size

SUMMARY OUTPUT					
Regression Statistics					
Multiple R	0.371379448				
R Square	0.137922694				
Adjusted R Square	0.128126361				
Standard Error	7632.174687				
Observations	90				
ANOVA					
	Df	*SS*	*MS*	*F*	*Significance F*
Regression	1	820103747.4	820103747.4	14.07901243	0.000313656
Residual	88	5126007960	58250090.46		
Total	89	5946111708			
	Coefficients	*Standard Error*	*t Stat*	*P-value*	
Intercept	2545.239613	1012.851226	2.512945188	0.013794529	
Size	9.210151353	2.454599667	3.752201011	0.000313656	

Table 6.4 shows a Multiple Regression Analysis between Development Effort and BFC Types. The total result is significant because the significance F value < 0.05. The coefficients Entry (E) and Exit (X) are significant because p-value < 0.05. There is a low Adjusted R^2 value (.18) between Development Effort and BFC Types which shows a weak relationship.

Table 6.4. Multiple Regression Analysis between Development Effort and BFC Types

SUMMARY OUTPUT					
Regression Statistics					
Multiple R	0.461498067				
R Square	0.212980466				
Adjusted R Square	0.175944253				
Standard Error	7419.930288				
Observations	90				
ANOVA					
	Df	*SS*	*MS*	*F*	*Significance F*
Regression	4	1266405642	316601410.5	5.750600468	0.000381612
Residual	85	4679706066	55055365.48		
Total	89	5946111708			
	Coefficients	*Standard Error*	*t Stat*	*P-value*	
Intercept	2327.003277	1021.888413	2.277159862	0.025287872	
E	-34.82285587	17.87001717	-1.948675009	0.054633114	
X	38.53001085	16.4532696	2.341784447	0.02153086	
R	8.870229052	17.03626176	0.520667572	0.603952055	
W	34.44083015	24.08125625	1.43019242	0.156327507	

The adjusted R^2 value increased from .13 to .18, when we considered the sizes of each BFC Types instead of using Total Functional Size. It shows that using BFC types instead of Total Functional Size might better estimate the Development Effort. Although, the adjusted R^2 value increased but we still have a weak relationship between Functional Size and Development Effort.

6.3.2 Empirical Study 2

Second, we performed the following statistical analysis:

1. Regression Analysis between Development Effort and Maximum Team Size

A) Data Preparation

Following steps are carried out to prepare the data:

Step1 – Same as in Empirical Study 1 (see Table 6.2)

Step2 – Same as in Empirical Study 1 (see Table 6.2)

Step3 – The third step is to remove projects having empty Max. Team Size. We have now remaining 60 projects.

A summary of the steps is shown in the following table.

Table 6.5. Filtration of ISBSG Dataset 2007 Release 10

STEP	ATTRIBUTE	FILTER	PROJECT EXCLUDED	PROJECTS REMAINING
3.	MAX. TEAM SIZE	≠ EMPTY (DATA IS NOT REPORTED)	50	60

B) Statistical Analysis & Results

Table 6.6 shows a Linear Regression Analysis between Development Effort and Max. Team Size. The total result is significant because the significance F value < 0.05. The coefficient Max. Team Size is also significant because p-value < 0.05. The adjusted R^2 value is .57 between Development Effort and Max. Team Size which shows a strong relationship.

Table 6.6. Linear Regression Analysis between Development Effort and Max. Team Size

SUMMARY OUTPUT					
Regression Statistics					
Multiple R	0.760789198				
R Square	0.578800204				
Adjusted R Square	0.571538139				
Standard Error	7017.783975				
Observations	60				
ANOVA					
	Df	*SS*	*MS*	*F*	*Significance F*
Regression	1	3925260715	3925260715	79.70187108	1.74632E-12
Residual	58	2856458931	49249291.92		
Total	59	6781719646			
	Coefficients	*Standard Error*	*t Stat*	*P-value*	
Intercept	1852.271977	1115.410863	1.660618556	0.102187364	
Max.TeamSize	587.7127848	65.83106479	8.927590441	1.74632E-12	

6.3.3 Empirical Study 3

Third, we performed the following statistical analysis:

1. Regression Analysis between Development Effort and Quality (Total Defects Delivered/CFP)

A) Data Preparation
Following steps are carried out to prepare the data:

Step1 – Same as in Empirical Study 1 (see Table 6.2)

Step2 – Same as in Empirical Study 1 (see Table 6.2)

Step3 – The third step is to remove those projects whose Total Defects Delivered are not reported. After this filtration, 44 projects remained.

A summary of the steps is shown in the following table.

Table 6.7. Filtration of ISBSG Dataset 2007 Release 10

STEP	ATTRIBUTE	FILTER	PROJECT EXCLUDED	PROJECTS REMAINING
3.	TOTAL DEFECTS DELIVERED	≠ EMPTY (DATA IS NOT REPORTED)	66	44

B) Statistical Analysis & Results
Table 6.8 shows a Linear Regression Analysis between Development Effort and Total Defects Delivered/CFP. The total result is not significant because the significance F value > 0.05. There is not a significant relationship between Development Effort and Total Defects Delivered/CFP.

Table 6.8. Linear Regression Analysis between Development Effort and Total Defects Delivered/CFP

SUMMARY OUTPUT					
Regression Statistics					
Multiple R	0.036333842				
R Square	0.001320148				
Adjusted R Square	-0.022457944				
Standard Error	9234.486381				
Observations	44				
ANOVA					

	Df	*SS*	*MS*	*F*	*Significance F*
Regression	1	4734467.418	4734467.418	0.055519512	0.814868254
Residual	42	3581581026	85275738.72		
Total	43	3586315494			
	Coefficients	*Standard Error*	*t Stat*	*P-value*	
Intercept	6349.58308	1708.073128	3.717395337	0.000589546	
Total Defects Delivered/CFP	12736.2847	54053.01678	0.235625789	0.814868254	

6.3.4 Empirical Study 4

In this study, we performed the following statistical analyses:

1. One Way ANOVA between ln Development Effort and Architecture
2. One Way ANOVA between ln Development Effort and Development Platform
3. One Way ANOVA between ln Development Effort and Language Type
4. One Way ANOVA between ln Development Effort and Development Type
5. One Way ANOVA between ln Development Effort and Application Type

We also used categorical independent variable in our analyses. ANOVA is performed when we have categorical independent variable (see Table 6.1). We performed one way ANOVA to investigate the affect of single categorical independent variable on the dependent variable. We also transformed the data to the logarithmic scale to make date normally distributed.

A) Data Preparation

Following steps are carried out to prepare the data:

Step1 – Same as in Empirical Study 1 (see Table 6.2)

Step2 – Same as in Empirical Study 1 (see Table 6.2)

Step3 – In this step, we performed filtration for each of the analysis independently.

For the first analysis, we removed those projects whose Architecture is not reported. 85 projects left after the filtration.

For the second analysis, we removed those projects whose Development Platform is not reported. 80 projects left after the filtration.

For the third analysis, we removed those projects whose Language Type is not reported. 79 projects left after the filtration.

Table 6.9. Filtration of ISBSG Dataset 2007 Release 10

STEP 3	ATTRIBUTE	FILTER	PROJECT EXCLUDED	PROJECTS REMAINING
A	ARCHITECTURE TYPE	= SINGLE TIER OR MULTI TIER OR STAND ALONE	25	85
B	DEVELOPMENT PLATFORM	≠ EMPTY (DATA IS NOT REPORTED)	5	80
C	LANGUAGE TYPE	≠ EMPTY (DATA IS NOT REPORTED)	1	79

It is important to note that we removed the project having Language 2GL while performing One Way ANOVA between ln Development Effort and Language Type because 2GL has only one project data. We have 78 records while performing One Way ANOVA between ln Development Effort and Language Type.

Another important consideration is that we selected the Application Types having data points greater then 4 (n > 4) while performing ANOVA between ln Development Effort and Application Type. Following Application Types were filtered:

1. Customization to a Product Data Management System (n = 14)
2. Operating System or Software Utility (Re-usable Component) (n = 7)
3. Financial Transaction Process/Accounting (n = 8)
4. Document Management; Financial Transaction Process/Accounting; Image, Video or Sound Processing (n = 12)

B) Statistical Analysis & Results

Table 6.10 shows a One Way ANOVA between ln Development Effort and Development Type. The p-value > 0.05 means that Development Type has no effect on ln Development Effort. Development Type is not a significant factor.

Table 6.10. One Way ANOVA between ln Development Effort and Development Type

Anova: Single Factor						
SUMMARY						
Groups	*Count*	*Sum*	*Average*	*Variance*		
Enhancement	38	279.8215	7.363723	4.18214		
New Development	35	260.8411	7.452604	3.835714		
Re-Development	6	52.07256	8.67876	1.5208		
ANOVA						
Source of Variation	*SS*	*Df*	*MS*	*F*	*P-value*	*F crit*
Between Groups	9.120492	2	4.560246	1.183842	0.311680654	3.116981837

Within Groups	292.7575	76	3.852072			
Total	301.878	78				

Table 6.11 shows a One Way ANOVA between ln Development Effort and Architecture. The p-value < 0.05 means that Architecture has an effect on ln Development Effort. Architecture is a significant factor.

Table 6.11. One Way ANOVA between ln Development Effort and Architecture

Anova: Single Factor						
SUMMARY						
Groups	*Count*	*Sum*	*Average*	*Variance*		
Client Server	38	308.3226582	8.113754	2.707697709		
Multi Tier	22	148.6804434	6.758202	4.036495096		
Stand Alone	19	135.7320748	7.143793	4.894248275		
ANOVA						
Source of Variation	*SS*	*Df*	*MS*	*F*	*P-value*	*F crit*
Between Groups	28.83027281	2	14.41514	4.012304232	0.022053098	3.116981837
Within Groups	273.0476812	76	3.592733			
Total	301.877954	78				

Table 6.12 shows a One Way ANOVA between ln Development Effort and Development Platform. The p-value > 0.05 means that Development Platform has no effect on ln Development Effort. Development Platform is not a significant factor.

Table 6.12. One Way ANOVA between ln Development Effort and Development Platform

Anova: Single Factor						
SUMMARY						
Groups	*Count*	*Sum*	*Average*	*Variance*		
MF	10	84.00998239	8.400998239	2.119020969		
MR	5	41.99427968	8.398855936	1.811274739		
Multi	30	221.5061895	7.383539652	4.513454464		
PC	34	245.2247247	7.212491904	3.918095079		
ANOVA						
Source of Variation	*SS*	*Df*	*MS*	*F*	*P-value*	*F crit*
Between Groups	15.3743493	3	5.1247831	1.341549377	0.267326443	2.726589185
Within Groups	286.5036047	75	3.820048063			
Total	301.877954	78				

Table 6.13 shows a One Way ANOVA between ln Development Effort and Language Type. The p-value > 0.05 means that Language Type has no effect on ln Development Effort. Language Type is not a significant factor.

Table 6.13. One Way ANOVA between ln Development Effort and Language Type

Anova: Single Factor						
SUMMARY						
Groups	*Count*	*Sum*	*Average*	*Variance*		
3GL	69	519.6294	7.530861	3.993731		
4GL	9	64.68288	7.186986	3.563212		
ANOVA						
Source of Variation	*SS*	*Df*	*MS*	*F*	*P-value*	*F crit*
Between Groups	0.941451	1	0.941451	0.238438	0.626743	3.96676
Within Groups	300.0794	76	3.948414			
Total	301.0209	77				

Table 6.14 shows a One Way ANOVA between ln Development Effort and Application Type. The p-value < 0.05 means that Application Type has an effect on ln Development Effort. Application Type is a significant factor.

Table 6.14. One Way ANOVA between ln Development Effort and Application Type

Anova: Single Factor						
SUMMARY						
Groups	*Count*	*Sum*	*Average*	*Variance*		
Customization to a Product Data Management System	14	77.37280175	5.526628696	1.965231327		
Financial Transaction Processing/Accounting	8	62.06155342	7.757694178	2.351268004		
Document Management; Financial Transaction Process/Accounting; Image, Video or Sound Processing	12	112.2660425	9.355503543	0.474231775		
Operating System or Software Utility (Re-usable Component)	7	37.1120462	5.301720885	0.173569504		
ANOVA						
Source of Variation	*SS*	*Df*	*MS*	*F*	*P-value*	*F crit*
Between Groups	121.6740319	3	40.55801063	31.09191055	3.26026E-10	2.858796061
Within Groups	48.26484982	37	1.304455401			

Total	169.9388817	40				

In Empirical Study 4, single factor ANOVA was performed. The result shows that Architecture and Application Type affects the effort estimation. Development Type, Development Platform and Language Type were found insignificant. There is a possibility that these factors do not individually affect the effort estimation, but they may affect when we combine these factors.

Although Application Type and Architecture are found significant but there is a possibility that when we see the combine effect of both these then one of the factors may be found significant (see Empirical Study: Phase 2).

6.3.5 Empirical Study 5

Based on the results of Empirical Study 4, we formed a sub-dataset considering Architecture Type and performed Multiple Regression Analysis between ln Development Effort and BFC Types.

A) Data Preparation

Following steps are carried out to prepare the data:

Step1 – Same as in Empirical Study 1 (see Table 6.2)

Step2 – Same as in Empirical Study 1 (see Table 6.2)

Step3 – Same as in Empirical Study 1 (see Table 6.2)

Step3 – The fourth step is to remove those projects whose Architecture Type is missing. 73 projects remained after this filtration. Architecture has three types: client server, multi tier and stand alone.

A summary of the steps is shows in the following table.

Table 6.15. Filtration of ISBSG Dataset 2007 Release 10

STEP	ATTRIBUTE	FILTER	PROJECT EXCLUDED	PROJECTS REMAINING
4.	ARCHITECTURE TYPE	= CLIENT SERVER, MULTI TIER, STAND ALONE	17	73

B) Statistical Analysis & Results

Table 6.16 shows a Multiple Regression Analysis between ln Development Effort and BFC Types for the dataset formed according to Architecture Type: client server. The total result is significant because the significance F value < 0.05. The adjusted R^2 is .16 which shows a weak relationship between ln Development Effort and BFC Types. Entry (E), Exit (X) and Write (W) coefficients are significant.

Table 6.16. Multiple Regression Analysis between ln Development Effort and BFC Types based on Architecture (Client Server)

SUMMARY OUTPUT					
Regression Statistics					
Multiple R	0.4974921				
R Square	0.24749839				
Adjusted R Square	0.161498205				
Standard Error	1.479626628				
Observations	40				
ANOVA					
	Df	*SS*	*MS*	*F*	*Significance F*
Regression	4	25.202131	6.30053275	2.877882091	0.036788056
Residual	35	76.6253235	2.189294957		
Total	39	101.8274545			
	Coefficients	*Standard Error*	*t Stat*	*P-value*	
Intercept	7.527461556	0.329106331	22.87243009	1.23926E-22	
E	-0.015009751	0.007407366	-2.026327552	0.050405213	
X	0.016686098	0.007344638	2.271874869	0.0293484	
R	-0.00990784	0.005894319	-1.680913614	0.101684209	
W	0.018259321	0.007440312	2.45410683	0.019241879	

Table 6.17 shows a Multiple Regression Analysis between ln Development Effort and BFC Types for Architecture: multi tier. The total result is significant because the significance F value < 0.05. The adjusted R^2 is .49. The adjusted R^2 value increased from .16 to .49 for multi tier architecture. Only the Write (W) coefficient is significant.

Table 6.17. Multiple Regression Analysis between ln Development Effort and BFC Types based on Architecture (Multi Tier)

SUMMARY OUTPUT					
Regression Statistics					
Multiple R	0.765645183				
R Square	0.586212547				
Adjusted R Square	0.488850793				
Standard Error	1.436402195				
Observations	22				

ANOVA					
	Df	*SS*	*MS*	*F*	*Significance F*
Regression	4	49.69113	12.42278	6.020973586	0.003307594
Residual	17	35.07527	2.063251		
Total	21	84.7664			
	Coefficients	*Standard Error*	*t Stat*	*P-value*	
Intercept	4.949546962	0.62138	7.965412	3.87154E-07	
E	0.016097772	0.01856	0.867328	0.397840348	
X	-0.000675774	0.01631	-0.04143	0.967433594	
R	-0.003033565	0.014344	-0.21149	0.835019449	
W	0.085957321	0.035122	2.44739	0.025548441	

Table 6.18 shows a Multiple Regression Analysis between ln Development Effort and BFC Types for Architecture: stand alone. The total result is not significant because the significance F value > 0.05.

Table 6.18. Multiple Regression Analysis between ln Development Effort and BFC Types based on Architecture (Stand Alone)

SUMMARY OUTPUT					
Regression Statistics					
Multiple R	0.26419835				
R Square	0.069800768				
Adjusted R Square	-0.550332053				
Standard Error	2.097166671				
Observations	11				
ANOVA					
	Df	*SS*	*MS*	*F*	*Significance F*
Regression	4	1.980164955	0.495041239	0.112557771	0.97341654
Residual	6	26.38864827	4.398108045		
Total	10	28.36881322			
	Coefficients	*Standard Error*	*t Stat*	*P-value*	
Intercept	6.091269416	0.982073433	6.202458198	0.000809821	
E	-0.074127262	0.133471077	-0.555380714	0.598714778	
X	0.04511987	0.092298184	0.488848945	0.642302297	
R	0.004335016	0.033953982	0.127673278	0.90257913	
W	0.053074972	0.097091926	0.54664661	0.604339595	

In Empirical Study 5, we formed homogenous data sets based on Architecture Type. The main purpose was to observe the affects of these factors as well as whether using BFC types as a functional size instead of Total Size would improve the effort estimation reliability or not. For Architecture: client server, adjusted R^2 is decreased from .18 to .16. The adjusted R^2 value is improved from .18 to .49 for Architecture: multi tier. We did not find any significant result for the Architecture: stand alone. An interesting observation is that E, X and W in

Architecture: client server and only W in Architecture: multi tier can model the normalized work effort.

6.3.6 Empirical Study 6

Based on the results of Empirical Study 4, we formed a sub-dataset considering Application Type and performed Multiple Regression Analysis between ln Development Effort and BFC Types.

A) Data Preparation

Following steps are carried out to prepare the data:

Step1 – Same as in Empirical Study 1 (see Table 6.2)

Step2 – Same as in Empirical Study 1 (see Table 6.2)

Step3 – Same as in Empirical Study 1 (see Table 6.2)

Step4 – We have included only those Application Types in which we have more than ten observations. (n>10):

1. Customization to a Product Data Management System
2. Document Management; Financial Transaction Process/Accounting; Image, Video or Sound Processing
3. Management Information System; Linguistic Software - Glue software

Table 6.19. Filtration of ISBSG Dataset 2007 Release 10

STEP	ATTRIBUTE	FILTER	PROJECT EXCLUDED	PROJECTS REMAINING
4.	APPLICATION TYPE	= Customization to a Product Data Management System OR Document management; Financial transaction process/accounting; Image, video or sound processing OR Management Information System; Linguistic Software - Glue software	50	40

B) Statistical Analysis & Results

Table 6.20 shows a Multiple Regression Analysis between ln Development Effort and BFC Types for Application Type: Customization to a Product Data Management System. The total result is not significant because the significance F value > 0.05.

Table 6.20. Multiple Regression Analysis between ln Development Effort and BFC Types based on Application Type (Customization to a Product Data Management System)

SUMMARY OUTPUT					
Regression Statistics					
Multiple R	0.728514359				
R Square	0.530733172				
Adjusted R Square	0.322170137				
Standard Error	281.6709857				
Observations	14				
ANOVA					
	Df	*SS*	*MS*	*F*	*Significance F*
Regression	4	807575.4596	201893.8649	2.544713506	0.112556638
Residual	9	714046.8975	79338.54417		
Total	13	1521622.357			
	Coefficients	*Standard Error*	*t Stat*	*P-value*	
Intercept	146.4027554	162.8858605	0.898805796	0.392167994	
E	-1.500492336	4.277558545	-0.350782419	0.733822365	
X	-2.705153264	4.626555356	-0.584701372	0.573112279	
R	8.814321574	4.052036713	2.175281765	0.057615673	
W	-19.42450633	40.89579792	-0.474975604	0.646118861	

Table 6.21 shows a Multiple Regression Analysis between ln Development Effort and BFC Types for Application Type: Document Management; Financial Transaction Process/Accounting; Image, Video or Sound Processing. The total result is not significant because the significance F value > 0.05.

Table 6.21. Multiple Regression Analysis between ln Development Effort and BFC Types based on Application Type (Document Management; Financial Transaction Process/Accounting; Image, Video or Sound Processing)

SUMMARY OUTPUT					
Regression Statistics					
Multiple R	0.630627868				
R Square	0.397691508				
Adjusted R Square	0.053515227				
Standard Error	10245.60506				
Observations	12				
ANOVA					

	Df	*SS*	*MS*	*F*	*Significance F*
Regression	4	485177434.5	121294358.6	1.155487843	0.405614201
Residual	7	734806961.1	104972423		
Total	11	1219984396			
	Coefficients	*Standard Error*	*t Stat*	*P-value*	
Intercept	-2811.34366	8905.331169	-0.315692208	0.761440199	
E	288.7813394	700.5317175	0.412231641	0.692501262	
X	94.75179888	609.9011379	0.155355996	0.880925637	
R	-72.75070972	560.8826025	-0.129707553	0.900446302	
W	27.97158502	485.4204651	0.057623415	0.955658998	

Table 6.22 shows a Multiple Regression Analysis between ln Development Effort and BFC Types for Application Type: Management Information System; Linguistic Software - Glue software. The relation is significant because the significance F value < 0.05. The adjusted R^2 is .64 which shows a strong relationship. Only the Write (W) coefficient is significant.

Table 6.22. Multiple Regression Analysis between ln Development Effort and BFC Types based on Application Type (Management Information System; Linguistic Software - Glue software)

SUMMARY OUTPUT					
Regression Statistics					
Multiple R	0.867556604				
R Square	0.752654462				
Adjusted R Square	0.642723112				
Standard Error	113.3278567				
Observations	14				
ANOVA					
	Df	*SS*	*MS*	*F*	*Significance F*
Regression	4	351728.3864	87932.09661	6.846586	0.008166398
Residual	9	115588.8278	12843.20309		
Total	13	467317.2143			
	Coefficients	*Standard Error*	*t Stat*	*P-value*	
Intercept	77.91085507	59.23927658	1.315189171	0.220966	
E	2.197135021	2.100938778	1.045787266	0.322933	
X	0.322095431	0.53763103	0.599101266	0.56388	
R	-5.301791235	3.445485526	-1.538764623	0.158244	
W	10.95950475	4.292930704	2.552919091	0.031046	

In Empirical Study 6, we formed homogenous data sets based on Application Types. The main purpose was to observe the affects of these factors as well as whether using BFC Types instead of Total Size would improve the effort estimation reliability or not. For the Application Type: Management Information System; Linguistic Software - Glue software, the adjusted R^2 value is improved from .18 to .64. Only the write (W) coefficient can model the normalized work effort.

A comparison of different analyses using BFC Types instead of total size is presented in the following table:

Table 6.23. Comparison of BFC Types and Total Functional Size

Sr. No.	Statistical Analysis	Adj. R^2	Significance
1.	Regression Analysis between Development Effort and Total Size	.13	Yes
2.	Regression Analysis between Development Effort and BFC Types	.18	Yes
3.	Regression Analysis between ln Development Effort and BFC Types, w.r.t. Architecture (Client Server)	.16	Yes
4.	Regression Analysis between ln Development Effort and BFC Types, w.r.t. Architecture (Multi Tier)	.49	Yes
5.	Regression Analysis between ln Development Effort and BFC Types, w.r.t. Architecture (Stand Alone)	-.55	No
6.	Regression Analysis between ln Development Effort and BFC Types, w.r.t. Application Type (Customization to a Product Data Management System)	.32	No
7.	Regression Analysis between ln Development Effort and BFC Types, w.r.t. Application Type (Document Management; Financial Transaction Process/Accounting; Image, Video or Sound Processing)	.05	No
8.	Regression Analysis between ln Development Effort and BFC Types, w.r.t. Application Type (Management Information System; Linguistic Software - Glue software)	.64	Yes

The result shows that Max. Team Size, Application Type, Architecture and Size have a significant relation with normalized work effort when considering each of them independently.

In the second phase of this thesis study, we investigated the affect of these factors independently as well as collectively to observe whether adjusted R^2 would improve or not. We also investigated whether using BFC Types instead of Total Size would improve the effort estimation reliability or not. The details of Phase 2 empirical studies are discussed in the next section.

6.4 Empirical Studies: Phase 2

In this phase, we investigated the affect of Size, Application Type, Architecture, Quality (Total Defects Delivered/CFP) and Max. Team Size on the effort. We investigated which of the factors affects Development Effort. The dependent variable is Development Effort while the independent variables are Size, Application Type, Architecture, Quality (Total Defects Delivered) and Max. Team Size.

6.4.1 Data Preparation

Following steps are carried out to prepare the data:

Step1 – There are total 4106 projects. We have selected those projects which have high Data Quality Rating. We have removed projects having either C or D Data Quality Rating. The selected projects have A or B Data Quality Rating. After this filtration, 3811 projects remained.

Step2 – The next step is to select only those projects having Count Approach COSMIC FFP. The remaining projects are 110.

Step3 – The third step is to remove those projects whose BFC Types (Entry, Exit, Read and Write) are not reported. The remaining projects are 90.

Step4 – The next step is to filter those projects whose Architecture is not reported. The remaining projects are 73.

Step5 – At step 5, we filtered those projects whose Max. Team Size is not reported. The remaining projects are 55.

Step6 – We filtered those projects whose Total Defects Delivered are not reported. The remaining projects are 37.

Table 6.24. Filtration of ISBSG Dataset 2007 Release 10

STEP	ATTRIBUTE	FILTER	PROJECT EXCLUDED	PROJECTS REMAINING
1.	DATA QUALITY RATING	= A OR B	295	3811
2.	COUNT APPROACH	= COSMIC-FFP	3701	110
3.	SIZE ATTRIBUTES (COSMIC ENTRY, EXIT, READ, WRITE)	= EMPTY (DATA IS NOT REPORTED)	20	90
4.	ARCHITECTURE	= EMPTY (DATA IS NOT REPORTED)	17	73
5.	MAX TEAM SIZE	= EMPTY (DATA IS NOT REPORTED)	18	55
6.	TOTAL DEFECTS DELIVERED	= EMPTY, NOT KNOWN (DATA IS NOT REPORTED)	8	37

The dependent variable is Development Effort. We used natural logarithm of Development Effort to make it normally distributed. Size is also converted into natural logarithm to make it normally distributed.

After initial filtration, we filtered out 37 projects. The characteristics of these 37 projects are described in the following table.

Table 6.25. Characteristics of Projects After Initial Filtration

APPLICATION TYPE	OBSERVATION
DOCUMENT MANAGEMENT; FINANCIAL TRANSACTION PROCESS/ACCOUNTING; IMAGE, VIDEO OR SOUND PROCESSING/ VIDEO OR SOUND PROCESSING	7
OPERATING SYSTEM OR SOFTWARE UTILITY;(RE-USABLE COMPONENT)	7
ONLINE ANALYSIS AND REPORTING	3
FINANCIAL TRANSACTION PROCESS	7
OTHERS	< 3 OBSERVATION

Step7 - We have included only those Application Types in which we have 3 or more than three observation.

Table 6.26. Further Filtration of ISBSG Dataset 2007 Release 10

STEP	ATTRIBUTE	FILTER	PROJECT EXCLUDED	PROJECTS REMAINING
7.	APPLICATION TYPE	= DOCUMENT MANAGEMENT; FINANCIAL TRANSACTION PROCESS/ACCOUNTING; IMAGE, VIDEO OR SOUND PROCESSING;/VIDEO OR SOUND PROCESSING OR OPERATING SYSTEM OR SOFTWARE UTILITY;(RE-USABLE COMPONENT) OR ONLINE ANALYSIS AND REPORTING OR FINANCIAL TRANSACTION PROCESS	13	24

6.4.2 Statistical Analysis

A) Determine Best One Variable Model

We wanted to find the best one variable model. Our aim was to find out which variable, ln Size, Max. Team Size, Total Defects Delivered/CFP, Architecture Type, Application Type, explains the most variation in ln Development Effort. We performed Regression Analyses for the numerical variables and ANOVA for the categorical variables. We had 24 observations.

Table 6.27 shows a Linear Regression Analysis between ln Development Effort and ln Size. The total result is significant because the significance of regression is .004 which is < 0.05. The R^2 value is .32. The coefficient ln Size is also significant because sig. < 0.05.

Table 6.27. Linear Regression Analysis between ln Development Effort and ln Size

Model Summary

Model	R	R Square	Adjusted R Square	Std. Error of the Estimate
1	.566[a]	.320	.289	1.64560

a. Predictors: (Constant), lnSize

ANOVA[b]

Model		Sum of Squares	Df	Mean Square	F	Sig.
1	Regression	28.022	1	28.022	10.348	.004[a]
	Residual	59.576	22	2.708		
	Total	87.598	23			

a. Predictors: (Constant), lnSize

b. Dependent Variable: lnDevelopmentEffort

Coefficients[a]

Model		Unstandardized Coefficients		Standardized Coefficients	t	Sig.
		B	Std. Error	Beta		
1	(Constant)	2.678	1.507		1.777	.089
	lnSize	.992	.309	.566	3.217	.004

a. Dependent Variable: lnDevelopmentEffort

Table 6.28 shows a Linear Regression Analysis between ln Development Effort and Max. Team Size. The total result is significant because the significance of regression is .000 which is < 0.05. The R^2 value is .60. The coefficient Max. Team Size is also significant because sig. < 0.05.

Table 6.28. Linear Regression Analysis between ln Development Effort and Max.TeamSize

Model Summary

Model	R	R Square	Adjusted R Square	Std. Error of the Estimate
1	.776[a]	.603	.585	1.25741

a. Predictors: (Constant), MaxTeamSize

ANOVA[b]

Model		Sum of Squares	Df	Mean Square	F	Sig.
1	Regression	52.814	1	52.814	33.403	.000[a]
	Residual	34.784	22	1.581		
	Total	87.598	23			

a. Predictors: (Constant), MaxTeamSize

b. Dependent Variable: lnDevelopmentEffort

Coefficients[a]

Model		Unstandardized Coefficients		Standardized Coefficients	t	Sig.
		B	Std. Error	Beta		
1	(Constant)	6.021	.351		17.170	.000
	MaxTeamSize	.172	.030	.776	5.780	.000

a. Dependent Variable: lnDevelopmentEffort

Table 6.29 shows a Linear Regression Analysis between ln Development Effort and Total Defects Delivered/CFP. The total result is not significant because the significance of regression is .64 which is > 0.05.

Table 6.29. Linear Regression Analysis between ln Development Effort and Total Defects Delivered/CFP

Model Summary

Model	R	R Square	Adjusted R Square	Std. Error of the Estimate
1	.099[a]	.010	-.035	1.98557

a. Predictors: (Constant), TotalDefectsDelivered/CFP

ANOVA[b]

Model	Sum of Squares	Df	Mean Square	F	Sig.

1	Regression	.863	1	.863	.219	.644[a]
	Residual	86.735	22	3.942		
	Total	87.598	23			

a. Predictors: (Constant), TotalDefectsDelivered/CFP

b. Dependent Variable: lnDevelopmentEffort

Coefficients[a]

Model		Unstandardized Coefficients		Standardized Coefficients	t	Sig.
		B	Std. Error	Beta		
1	(Constant)	7.251	.518		13.988	.000
	TotalDefectsDelivered/ CFP	6.867	14.677	.099	.468	.644

a. Dependent Variable: lnDevelopmentEffort

ANOVA is performed when we have categorical independent variable (see Table 6.1). We had Architecture as categorical independent variable. To see the affect of Architecture, we performed ANOVA. Table 6.30 shows a One Way ANOVA between ln Development Effort and Architecture. The significance < 0.05 that means that Architecture has an effect on ln Development Effort. Architecture is a significant factor.

Table 6.30. One Way ANOVA between ln Development Effort and Architecture

ANOVA

lnDevelopmentEffort

	Sum of Squares	Df	Mean Square	F	Sig.
Between Groups	44.867	2	22.433	11.025	.001
Within Groups	42.731	21	2.035		
Total	87.598	23			

We had Application Type as categorical independent variable. To see the affect of Application Type, we performed ANOVA. Table 6.31 shows a One Way ANOVA between ln Development Effort and Application Type. The significance < 0.05 that means that Application Type has an effect on ln Development Effort. Application Type is a significant factor.

Table 6.31. One Way ANOVA between ln Development Effort and Application Type

ANOVA

lnDevelopmentEffort

	Sum of Squares	Df	Mean Square	F	Sig.
Between Groups	63.517	3	21.172	17.584	.000
Within Groups	24.081	20	1.204		

Total	87.598	23			

B) Determine Best Two Variable Model

In determining best one variable model, we found that Max. Team Size explained the most variation in ln Development Effort. For the best two variable models, we wanted to determine which variable, ln Size, Total Defects Delivered/CFP, Architecture Type, Application Type, in addition to Max. Team Size explains the most variation in ln Development Effort. We looked the factors collectively because some factors might be insignificant individually, but they might affect the Development Effort, when we combined them with other factors. We performed Regression Analyses for the numerical variables, ANOVA for the categorical variables and ANCOVA for both numerical and categorical independent variables. We had 24 observations.

Table 6.32 shows a Multiple Regression Analysis between ln Development Effort, Max. Team Size and ln Size. The total result is significant because the significance of regression is .000 which is < 0.05. The adjusted R^2 value is .62.The coefficients Max. Team Size is also significant because sig. < 0.05.

Table 6.32. Multiple Regression Analysis between ln Development Effort, Max. Team Size and ln Size

Model Summary

Model	R	R Square	Adjusted R Square	Std. Error of the Estimate
1	.808[a]	.653	.619	1.20392

a. Predictors: (Constant), lnSize, MaxTeamSize

ANOVA[b]

Model		Sum of Squares	df	Mean Square	F	Sig.
1	Regression	57.160	2	28.580	19.718	.000[a]
	Residual	30.438	21	1.449		
	Total	87.598	23			

a. Predictors: (Constant), lnSize, MaxTeamSize

b. Dependent Variable: lnDevelopmentEffort

Coefficients[a]

Model		Unstandardized Coefficients		Standardized Coefficients	t	Sig.
		B	Std. Error	Beta		
1	(Constant)	4.120	1.148		3.588	.002
	MaxTeamSize	.145	.032	.656	4.484	.000
	lnSize	.444	.257	.253	1.732	.098

a. Dependent Variable: lnDevelopmentEffort

Table 6.33 shows a Multiple Regression Analysis between ln Development Effort, Max. Team Size and Total Defects Delivered/CFP. The total result is significant because the significance of regression is .000 which is < 0.05. The adjusted R^2 value is .61.The coefficients Max. Team Size is also significant because sig. < 0.05.

Table 6.33. Multiple Regression Analysis between ln Development Effort, Max. Team Size and Total Defects Delivered/CFP

Model Summary

Model	R	R Square	Adjusted R Square	Std. Error of the Estimate
1	.803[a]	.644	.610	1.21838

a. Predictors: (Constant), TotalDefectsDelivered/CFP, MaxTeamSize

ANOVA[b]

Model		Sum of Squares	Df	Mean Square	F	Sig.
1	Regression	56.425	2	28.212	19.005	.000[a]
	Residual	31.173	21	1.484		
	Total	87.598	23			

a. Predictors: (Constant), TotalDefectsDelivered/CFP, MaxTeamSize

b. Dependent Variable: lnDevelopmentEffort

Coefficients[a]

Model		Unstandardized Coefficients		Standardized Coefficients	T	Sig.
		B	Std. Error	Beta		
1	(Constant)	6.210	.361		17.216	.000
	MaxTeamSize	.190	.031	.858	6.118	.000
	TotalDefectsDelivered/ CFP	-15.123	9.697	-.219	-1.560	.134

a. Dependent Variable: lnDevelopmentEffort

ANCOVA is performed when we want to see the affect of categorical and non-categorical independent variables collectively (see Table 6.1). We had Max. Team Size as numerical variable and Application Type as categorical variable. To see the affect of these two independent variables collectively, we performed ANCOVA.

Table 6.34 shows a ANCOVA between ln Development Effort, Max. Team Size and Application Type. The total result is significant because the significance of model is .000 which is < 0.05. The adjusted R^2 value is .76. The coefficients Max. Team Size and Application Type are significant because sig. < 0.05.

Table 6.34. ANCOVA between ln Development Effort, Max. Team Size and Application Type

Tests of Between-Subjects Effects					
Dependent Variable:lnDevelopmentEffort					
Source	**Type III Sum of Squares**	**df**	**Mean Square**	**F**	**Sig.**
Corrected Model	70.089[a]	4	17.522	19.015	.000
Intercept	387.376	1	387.376	420.373	.000
MaxTeamSize	6.573	1	6.573	7.132	.015
AppType	17.275	3	5.758	6.249	.004
Error	17.509	19	.922		
Total	1402.570	24			
Corrected Total	87.598	23			
a. R Squared = .800 (Adjusted R Squared = .758)					

Table 6.35 shows a ANCOVA between ln Development Effort, Max. Team Size and Architecture. The total result is significant because the significance of model is .000 which is < 0.05. The adjusted R^2 value is .72. The coefficients Max. Team Size and Architecture are significant because sig. < 0.05.

Table 6.35. ANCOVA between ln Development Effort, Max. Team Size and Architecture

Tests of Between-Subjects Effects					
Dependent Variable:lnDevelopmentEffort					
Source	**Type III Sum of Squares**	**df**	**Mean Square**	**F**	**Sig.**
Corrected Model	66.477[a]	3	22.159	20.984	.000
Intercept	451.481	1	451.481	427.528	.000
MaxTeamSize	21.611	1	21.611	20.464	.000
Architecture	13.663	2	6.832	6.469	.007
Error	21.121	20	1.056		
Total	1402.570	24			
Corrected Total	87.598	23			
a. R Squared = .759 (Adjusted R Squared = .723)					

C) Determine Best Three Variable Model

In determining best two variable model, we found that Max. Team Size and Application Type explained the most variation in ln Development Effort. For the best three variable models, we wanted to determine which variable, ln Size, Total Defects Delivered/CFP,

Architecture, in addition to Max. Team Size and Application Type explains the most variation in ln Development Effort.

Table 6.36 shows a ANCOVA between ln Development Effort, Max. Team Size, Application Type and ln Size. The total result is significant because the significance of model is .000 which is < 0.05. The adjusted R^2 value is .82. The coefficients, Max. Team Size, ln Size and Application Type are significant because sig. < 0.05.

Table 6.36. ANCOVA between ln Development Effort, Max. Team Size, Application Type and ln Size

Tests of Between-Subjects Effects					
Dependent Variable:lnDevelopmentEffort					
Source	**Type III Sum of Squares**	**df**	**Mean Square**	**F**	**Sig.**
Corrected Model	75.108[a]	5	15.022	21.648	.000
Intercept	17.332	1	17.332	24.978	.000
MaxTeamSize	3.604	1	3.604	5.195	.035
lnSize	5.019	1	5.019	7.233	.015
AppType	17.948	3	5.983	8.622	.001
Error	12.490	18	.694		
Total	1402.570	24			
Corrected Total	87.598	23			
a. R Squared = .857 (Adjusted R Squared = .818)					

Table 6.37 shows a ANCOVA between ln Development Effort, Max. Team Size, Application Type and Total Defects Delivered/CFP. The total result is significant because the significance of model is .000 which is < 0.05. The adjusted R^2 value is .75. The coefficients Max. Team Size and Application Type are significant because sig. < 0.05.

Table 6.37. ANCOVA between ln Development Effort, Max. Team Size, Application Type and Total Defects Delivered/CFP

Tests of Between-Subjects Effects					
Dependent Variable:lnDevelopmentEffort					
Source	**Type III Sum of Squares**	**df**	**Mean Square**	**F**	**Sig.**
Corrected Model	70.263[a]	5	14.053	14.591	.000
Intercept	382.621	1	382.621	397.291	.000
MaxTeamSize	5.706	1	5.706	5.925	.026
TotalDefectsDelivered/ CFP	.173	1	.173	.180	.676
AppType	13.838	3	4.613	4.790	.013
Error	17.335	18	.963		
Total	1402.570	24			

Corrected Total	87.598	23			
a. R Squared = .802 (Adjusted R Squared = .747)					

Table 6.38 shows a ANCOVA between ln Development Effort, Max. Team Size, Application Type and Architecture. The total result is significant because the significance of model is .000 which is < 0.05. The adjusted R^2 value is .76. The coefficient Max. Team Size is significant because sig. < 0.05.

Table 6.38. ANCOVA between ln Development Effort, Max. Team Size, Application Type and Architecture

Tests of Between-Subjects Effects					
Dependent Variable:lnDevelopmentEffort					
Source	**Type III Sum of Squares**	**df**	**Mean Square**	**F**	**Sig.**
Corrected Model	71.164[a]	5	14.233	15.589	.000
Intercept	203.174	1	203.174	222.533	.000
MaxTeamSize	5.561	1	5.561	6.091	.024
AppType	4.686	2	2.343	2.566	.105
Architecture	1.075	1	1.075	1.177	.292
AppType * Architecture	.000	0	.	.	.
Error	16.434	18	.913		
Total	1402.570	24			
Corrected Total	87.598	23			
a. R Squared = .812 (Adjusted R Squared = .760)					

The results of the analyses are summarized into the following table.

Table 6.39. Summary Table

VARIABLE	NUM OBS.	ADJ R^2
1-VARIABLES MODELS		
LN SIZE	24	.29
MAXTEAM SIZE	24	.59
ARCHITECTURE	24	
APPLICATION TYPE	24	
2-VARIABLES MODELS WITH MAX TEAM SIZE		
ARCHITECTURE	24	.72
APPLICATION TYPE	24	.76
3-VARIABLES MODELS WITH MAX TEAM SIZE, APPLICATION TYPE		
LN SIZE	24	.82

The best three variables model is Max. Team Size, ln size and Application Type explains 82% of the effort. To further see the affect of Quality (Total Defects Delivered/CFP) and Architecture on Development Effort, we have made ANCOVA by using Max. Team Size, ln Size, Total Defects Delivered/CFP, Application Type and Architecture as independent variable and ln Development Effort as a dependent variable. We want to see that whether effort prediction will be increased or not by including Total Defects Delivered/CFP and Architecture in our best three variable model. The result of the final models is presented below:

Table 6.40. ANCOVA between ln Development Effort, ln Size, Total Defects Delivered/CFP, Max. Team Size, Application Type and Architecture

Tests of Between-Subjects Effects					
Dependent Variable:lnDevelopmentEffort					
Source	**Type III Sum of Squares**	**Df**	**Mean Square**	**F**	**Sig.**
Corrected Model	78.768[a]	7	11.253	20.389	.000
Intercept	2.435	1	2.435	4.412	.052
MaxTeamSize	.333	1	.333	.604	.448
lnSize	7.318	1	7.318	13.261	.002
TotalDefectsDelivered/ CFP	2.475	1	2.475	4.485	.050
AppType	7.307	2	3.653	6.620	.008
Architecture	.749	1	.749	1.357	.261
AppType * Architecture	.000	0	.	.	.
Error	8.830	16	.552		
Total	1402.570	24			
Corrected Total	87.598	23			
a. R Squared = .899 (Adjusted R Squared = .855)					

From the table 6.40, we observe that total model is significant because significance value is .000 which is <.05. ln Size, Total Defects Delivered/CFP and Application Type are found to be significant variables. The adjusted R2 is also increased. The three variable model ln Size, Total Defects Delivered/CFP and Application Type explains 86% variation in effort. Table 6.41 gives the values of the coefficients that are found significant.

Table 6.41. Coefficient Values of Significant Variables found (see Table 6.40)

Parameter Estimates						
Dependent Variable:lnDevelopmentEffort						
Parameter	**B**	**Std. Error**	**t**	**Sig.**	**95% Confidence Interval**	
					Lower Bound	**Upper Bound**
Intercept	.908	1.231	.738	.471	-1.701	3.516
MaxTeamSize	.043	.055	.777	.448	-.074	.161
lnSize	.881	.242	3.642	.002	.368	1.393
TotalDefectsDelivered/CFP	20.094	9.488	2.118	.050	-.020	40.208

[AppType=1]	2.677	.643	4.163	.001	1.314	4.040
[AppType=2]	3.787	.845	4.481	.000	1.995	5.579
[AppType=3]	1.891	1.013	1.866	.080	-.257	4.039
[AppType=4]	0[a]	.	.	.	.	.
[Architecture=1]	-1.122	.963	-1.165	.261	-3.163	.920
[Architecture =2]	0[a]	.	.	.	.	.
[Architecture =3]	0[a]	.	.	.	.	.
[AppType=1] * [Architecture =1]	0[a]	.	.	.	.	.
[AppType=1] * [Architecture =2]	0[a]	.	.	.	.	.
[AppType=2] * [Architecture =1]	0[a]	.	.	.	.	.
[AppType=3] * [Architecture =1]	0[a]	.	.	.	.	.
[AppType=4] * [Architecture =3]	0[a]	.	.	.	.	.
a. This parameter is set to zero because it is redundant.						

Where

- Application Type = 1 = Document Management; Financial Transaction Process/ Accounting; Image, Video or Sound Processing;
- Application Type = 2 = Financial Transaction Process/Accounting
- Application Type = 3 = Online Analysis and Reporting
- Application Type = 4 = Operating System or Software Reusability (Re-usable Components)

By using table 6.40, we can make the following equation

ln Development Effort = f (ln Size, Application Type, Total Defects Delivered/CFP)

From Table 6.41, we derived the following effort estimation model:

ln Development Effort = .881*ln Size + 20.094*Total Defects Delivered/CFP + appTypeCoeff

For Application Type = 1
appTypeCoeff = 2.677

For Application Type = 2
appTypeCoeff = 3.787

for Application Type = 3
appTypeCoeff = 1.891

For Application Type = 4
appTypeCoeff = 0

Table 6.42 shows a ANCOVA between ln Development Effort, BFC Types, Total Defects Delivered/CFP, Max. Team Size, Application Type, Architecture. The models is significant because sig < .05.

Table 6.42. ANCOVA between ln Development Effort, BFC Types, Total Defects Delivered/CFP, Max. Team Size, Application Type and Architecture

Tests of Between-Subjects Effects					
Dependent Variable:lnDevelopmentEffort					
Source	**Type III Sum of Squares**	**df**	**Mean Square**	**F**	**Sig.**
Corrected Model	77.137[a]	10	7.714	9.586	.000
Intercept	148.845	1	148.845	184.975	.000
E	.121	1	.121	.150	.705
R	.031	1	.031	.039	.847
X	.152	1	.152	.189	.671
W	.151	1	.151	.188	.672
TotalDefectsDelivered/ CFP	.339	1	.339	.421	.528
MaxTeamSize	.423	1	.423	.526	.481
AppType	4.079	2	2.039	2.534	.118
Architecture	.314	1	.314	.391	.543
AppType * Architecture	.000	0	.	.	.
Error	10.461	13	.805		
Total	1402.570	24			
Corrected Total	87.598	23			
a. R Squared = .881 (Adjusted R Squared = .789)					

6.4.3 Results and Discussion

The study explores the relationship between effort and functional size. We investigated whether considering significant factors along with functional size would improve the effort estimation reliability or not.

For the best one variable model, Max. Team Size is the variable that explains the most variation in ln Development Effort. The adjusted R^2 is .59. It means that Max. Team Size explains the 59% of the variation in ln Development Effort.

For the best two variable model, Application Type along with Max Team Size have a high adjusted R^2 value that is .76. It shows a strong relationship. Application Type along with Max. Team Size explains 76% variation in ln Development Effort. The ln Size is significant in determining best one variable model, but it is not significant in this two variable model. The adjusted R^2 is increased from .59 to .76 when we consider both Max. Team Size and Application Type.

The best three variable model is Max. Team Size, Application Type and ln Size. It explains almost 82% of the variation in effort. ln Size is also significant. Further, we have performed analysis using ln Development Effort, Max. Team Size, Total Defects Delivered/CFP,

Application Type and Architecture. The value of adjusted R^2 increased from .82 to .86. ln Size along with Application Type and Total Defects Delivered/CFP explains the 86% variation in effort. Mathematically, it can be written as:

ln Development Effort = f (ln Size, Application Type, Total Defects Delivered/CFP)

Max Team Size and Architecture are not significant in our final model. Both the size and effort have positive effect on effort estimation. As the size increases, the effort to complete the projects increases. It is also obvious from the relationship between Total Defects Delivered/CFP and the Development Effort has a positive effect on effort estimation. Our model shows that Application Type also has a significant effect on the effort estimation.

The adjusted R^2 value is decreased from.82 to .79 when we use BFC Types instead of total size. No BFC Types is significant.

Our first research question was: what factors affect the relationship between effort and functional size? We have found that effort is a function of Total Size, Total Defects Delivered/CFP and Application Type. It means that size of the software affect the effort estimation. Another important factor is the Application Type. If we ignore the Application Type, then we may not be able to find better estimation. Total Defects Delivered/CFP is also an important factor. All these three factors affect the effort estimation and explain almost 86% variation in ln Development Effort.

Our second research question was: What kind of representation of size can better be used for effort estimation? Our second research question addresses the representation of functional size. We have total COSMIC function size and BFC Types. If we use BFC Types instead of total COSMIC functional size then the adjusted R^2 value is decreased from .86 to .79. This contradicts with the Phase 1 results. In the Phase 1, we did not investigate the affect of numerical and categorical factors collectively. The results of the Phase 2 show that there are some factors that are collectively more significant than the BFC Types.

Our third research question was: Does representing software functional size in a different way and considering the other factors improve the effort estimation reliability? The effort estimation reliability has been improved from .13 to .86 if we use total size along with Application Type and Total Defects Delivered/CFP instead of using only total size. The effort estimation reliability would improve from .36 to .86 if we use total size along with Application Type and Total Defects Delivered/CFP instead of using only BFC Types. The effort estimation reliability would decrease from .86 to .79 if we use BFC Types along with application type and Total Defects Delivered/CFP instead of using total size along with Application Type and Total Defects Delivered/CFP.

CHAPTER 7: CONCLUSION AND FUTURE WORK

In this thesis, we explore the relationship between effort and size. We investigated what factors affect the nature of the relationship between effort and size as well as increase the reliability of effort estimation. We also explored whether different representations of software size instead of using one total size figure would improve the estimation reliability.

After performing a comprehensive literature survey, we found that Application Type, Architecture, Language and Development Type are the factors which have significant effects on effort estimation.

Accordingly, we performed a number of empirical studies utilizing the ISBSG dataset release 10 to investigate the effects of these factors on effort estimation. We made the analyses on the projects, which were measured by COSMIC Function Points. A number of filtration criteria were used when forming the data sets. Here, we considered Data Quality Rating attribute of ISBSG data and selected only high quality data for further analyses. We investigated the significance of Functional Size, Size of BFC Types, Max Team Size, Quality (Total Defects Delivered/CFP), Development Type, Architecture, Application Type, Language and Development Platform on estimating the Development Effort.

The empirical studies were conducted in two major phases. In phase 1, we performed 6 analyses. We performed a one way Analysis of Variance (ANOVA) to observe the effects of Development Type, Architecture, Application Type, Language and Development Platform on Normalized Work Effort. Application Type and Architecture are found to be significant. Then, we re-formed datasets based on these two factors. For the Architecture: client server, the adjusted R^2 is decreased from .18 to .16 when we compare adjusted R^2 of Effort and BFC Types, with adjusted R^2 of Effort and BFC Types based on Architecture Type. The adjusted R^2 is improved for multi tier from .18 to 49. The adjusted R2 is increased from .18 to .64 only for the Application Type: Management Information System; Linguistic Software - Glue software.

We further investigated the size-effort relationship by applying Analysis of Co-variance (ANCOVA) to find significant factors. We performed Step Wise ANOVA to build multi variable model. Application Type, Size, Architecture, Quality (Total Defects Delivered/CFP) and Max. Team Size were the factors that we considered. The study shows that Application Type, Functional Size and Quality (Total Defects Delivered/CFP) explains the most variation in Normalized Work Effort.

The result in the phase 2 shows that total COSMIC size along with other significant factors would better estimate the effort as compared to using only BFC Types for effort estimation. The adjusted R^2 is increased from .18 to .86. The study also shows that the adjusted R^2 value is decreased from .86 to .79 if we use total COSMIC size along with significant factors instead of using BFC Types with significant factor.

This study only considered four Application Types due to the limited available data in the ISBSG dataset. There is a need to further investigate the affect of different Application Types on Software Size and Development Effort relationship. As more data will be collected in different repositories, these studies can be replicated to observe whether the results can be generalized or not.

Another important point is that we used the Application Types defined in ISBSG data sets. There may be different definition of Application Types in other standards and data sets. More research is also required in standardizing the definition of different significant factors.

There is a need to further investigate the affect of different representation of Functional Size on effort estimation. Further research is also required for investigating different representations of software functional size.

REFERENCES

1. N. E. Fenton and S. L. Pfleeger, 1997. Software Metrics: A Rigorous and Practical Approach, 2nd Edition Revised ed. Boston: PWS Publishing.
2. Abran, A. 1999. Functional Size Measurement for Real Time and Embedded Software. *In Proceedings of the 4th IEEE international Symposium and Forum on Software Engineering Standards (May 17 - 21, 1999). ISESS*. IEEE Computer Society, Washington, DC, 259.
3. Forselius, P. 2004. *Finnish Software Measurement Association Functional Size.* Finnish Software Metrics Association, Finland.
4. Gencel, C. and Demirors, O. 2008. Measuring the Software Functional Size as a Vector of Measures. ACM Publishers.
5. Leung, H. and Fan, Z. 2002. *Software Cost Estimation, Handbook of Software Engineering*, Hong Kong Polytechnic University.
6. Boehm, Barry B., and Fairley, Richard E., Software Estimation Perspectives, IEEE Software, November-December 2000, pp.22-26
7. Gencel, C. and Demirors, O. 2008, Functional size measurement revisited. *ACM Trans. Softw. Eng. Methodol.* 17, pp. 1-36.
8. Kemerer, C. F. 1987. An empirical validation of software cost estimation models. *Commun. ACM* 30, pp. 416-429.
9. Gencel, C. and Buglione, L. 2008. Do Base Functional Component Types Affect the Relationship between Software Functional Size and Effort? Springer-Verlag, Berlin.
10. Kitchenham, B. 1997. The Problems with Function Points. IEEE software, 14, 2, pp. 29-31.
11. Gencel, C. 2008. How to use COSMIC Functional Size in Effort Estimation Models, to be published in the Proc. Of Mensura/IWSM/Metrikon 2008 conference, LNCS 2008.
12. *The United Kingdom Software Metrics Association: MkII Function Point Analysis Counting Practices Manual v. 1.3.1*, (1998).
13. *IFPUG, Function Point CPM, Release. 4.2.1,* (2005).
14. *The Common Software Measurement International Consortium FFP, version 2.2, Measurement Manual*, (2003).
15. ISO/IEC 24570:2005, Software Engineering – NESMA functional size measurement method version 2.1 – Definitions and counting guidelines for the application of Function Point Analysis, International Organization for Standardization (2005).
16. ISO/IEC 29881:2008, Software Engineering -- FiSMA functional size measurement method version 1.1, International Organization for Standardization, (2008).
17. B. W. Boehm, E. Horowitz, R. Madachy, D. Reifer, K. C. Bradford, B. Steece, A. W. Brown, S. Chu-lani, and C. Abts, 2000. Software Cost Estimation with COCOMO II. Prentice Hall, New Jersey.
18. Rollo, T. 2006. Functional size measurement and COCOMO—a synergistic approach. *In Proceedings of Software Measurement European Forum* (SMEF), Rome, Italy, pp. 259–267.
19. Neumann, R. and Santillo, L., 2006. Experience with the Usage of COCOMO II Effort Estimation in Embedded Software Context, *In the Proceedings of SMEF* 2006.
20. Premraj, R., Shepperd, M., Kitchenham, B., and Forselius, P. 2005. An Empirical Analysis of Software Productivity over Time. In *Proceedings of the 11th IEEE international Software Metrics Symposium* (September 19 - 22, 2005). METRICS. IEEE Computer Society, Washington, DC, 37.
21. Gencel, C. and Demirors, O. 2007. Conceptual Differences Among Functional Size Measurement Methods. In *Proceedings of the First international Symposium on Empirical Software Engineering and Measurement* (September 20 - 21, 2007). ESEM. IEEE Computer Society, Washington, DC, pp. 305-313.
22. L. M. Laird, and M. C. Brennan, 2006. Software Measurement and Estimation: A Practical Approach, Wiley-IEEE Computer Society Pr, ISBN: 0-471-67622-5.
23. Forselius, P., 2004. Moving from Function Point Counting to Better Project Management and Control, IWSM/MetriKon Presentation.
24. ISO, 1998. International Standard ISO/IEC 14143-1, Information Technology – Software Measurement – Functional Size Measurement Part 1: Definition of Concepts.

25. Maxwell, K. D. and Forselius, P. 2000. Benchmarking Software-Development Productivity. *IEEE Softw.* 17, 1 (Jan. 2000), pp. 80-88.
26. Angelis, L., Stamelos, I., and Morisio, M. 2001. Building A Software Cost Estimation Model Based On Categorical Data. In *Proceedings of the 7th international Symposium on Software Metrics* (April 04 - 06, 2001). METRICS. IEEE Computer Society, Washington, DC, 4.
27. Feldt, R., Torkar, R., Angelis, L., and Samuelsson, M. 2008. Towards individualized software engineering: empirical studies should collect psychometrics. In *Proceedings of the 2008 international Workshop on Cooperative and Human Aspects of Software Engineering* (Leipzig, Germany, May 13 - 13, 2008). CHASE '08. ACM, New York, NY, pp. 49-52.
28. Liebchen, G. A. and Shepperd, M. 2005. Software Productivity Analysis of a Large Data Set and Issues of Confidentiality and Data Quality. In *Proceedings of the 11th IEEE international Software Metrics Symposium* (September 19 - 22, 2005). METRICS. IEEE Computer Society, Washington, DC, 46.
29. Magazinovic, A. and Pernstål, J. 2008. Any other cost estimation inhibitors?. In *Proceedings of the Second ACM-IEEE international Symposium on Empirical Software Engineering and Measurement* (Kaiserslautern, Germany, October 09 - 10, 2008). ESEM '08. ACM, New York, NY, pp. 233-242.
30. Lokan, C., Wright, T., Hill, P. R., and Stringer, M. 2001. Organizational Benchmarking Using the ISBSG Data Repository. *IEEE Softw.* 18, 5 (Sep. 2001), pp. 26-32.
31. Yang, Y., He, M., Li, M., Wang, Q., and Boehm, B. 2008. Phase distribution of software development effort. In *Proceedings of the Second ACM-IEEE international Symposium on Empirical Software Engineering and Measurement* (Kaiserslautern, Germany, October 09 - 10, 2008). ESEM '08. ACM, New York, NY, pp. 61-69.
32. ISBSG Dataset 10, Field Description (2007), http://www.isbsg.org/
33. ISBSG Dataset 10, Demographics (2007), http://www.isbsg.org/
34. ISBSG Data set 10, Guidelines (2007), http://www.isbsg.org/
35. Creswell, W., 2002. Research Design - Qualitative, Quantitative and Mixed Method Approaches, Sage Publications.
36. C. Wohlin, P. Runeson, M. Host, M. C. Ohlsson, B. Regnell, A. Wesslen, 2000. Experimentation in Software Engineering: An Introduction, Kluwer Academic Publishers, ISBN: 0-7923-8666-3.
37. K. D. Maxwell, 2002. Applied Statistics for Software Managers, Prentice Hall PTR, ISBN 0-13-041789-0.
38. S. K. Kachigan, 1986. Statistical Analysis: An Interdisciplinary Introduction to Univariate and Multivariate Methods, Radius Press, ISBN 0-942152-99-1.
39. E. J. Pedhazur, 1997. Multiple Regression in Behavioral Research: Explanation and Prediction, Third Edition, Harcourt Brace College Publishers, ISBN: 0-03-072831-2.
40. SPSS 17.0, http://www.spss.com/
41. ISBSG Dataset 10 (2007), http://www.isbsg.org/
42. Albrecht, A. J. 1979. Measuring application development productivity. In *Proceedings of the IBM Applications Development Symposium*, October 14–17, Monterey, California, pp. 83–92.
43. Albrecht, A. J. and Gaffney J. E. 1983. Software function, source lines of code, and development effort prediction: A software science validation. *IEEE Trans. Softw. Eng. SE-9*, 6, pp. 639–648.
44. Albrecht, A. J. 1984. *AD/M Productivity Measurement and Estimate Validation*. IBM Corporate Information Systems, IBM Corp., Purchase, NY.
45. IFPUG. 1999. *IFPUG Counting Practices Manual - Release. 4.1*, International Function Point Users Group, Westerville, OH.
46. ISO. 2003c. *ISO/IEC 20926: Software Engineering - IFPUG 4.1 Unadjusted FSM Method - Counting Practices Manual.*
47. Demarco T. 1982. *Controlling Software Projects*, Yourdon press, New York.
48. Jones, T. C. 1987. *A Short History of Function Points and Feature Points*, Software Productivity Research Inc., USA.
49. Symons, C. 1988. Function Point analysis: Difficulties and improvements. *IEEE Trans. Softw. Eng. 14*, 1, pp. 2–11.
50. ISO. 2002c. *ISO/IEC 20968: Software Engineering - MkII Function Point Analysis – Counting Practices Manual*.

51. NESMA. 1997. *Definitions and Counting Guidelines for the Application of Function Point Analysis*, v.2.0.
52. ISO. 2005b. *ISO/IEC 24570: Software Engineering - NESMA Functional Size Measurement Method v.2.1 - Definitions and Counting Guidelines for the Application of Function Point Analysis*.
53. Reifer, D. J. 1990. Asset-R: A function point sizing tool for scientific and real-time systems. *J. Syst. Softw. 11*, 3, pp. 159–171.
54. Whitmire, S. A. 1992. 3D Function Points: Scientific and real-time extensions to Function Points. In *Proceedings of the Pacific Northwest Software Quality Conference*.
55. Banker, R., Kauffman, R. J.,Wright, C., and Zweig, D. 1994. Automating output size and reuse metrics in a repository based computer aided software engineering (CASE) environment. *IEEE Trans. Softw. Eng. 20*, 3, pp. 169–187.
56. Kauffman, R. and Kumar, R. 1997. Investigating object-based metrics for representing software output size. In *Proceedings of the Conference on Information Systems and Technology* (CIST). In the INFORMS 1997 Annual Conference, San Diego.
57. Matson, J. E., Barret, B. E., and Mellichamp, J. M. 1994. Software development cost estimation using Function Points. *IEEE Trans. Softw. Eng. 20*, 4, pp. 275–287.
58. Abran, A., ST-Pierre, D., Maya, M., and Desharnais J. M. 1998. Full function points for embedded and real-time software. In *Proceedings of the UKSMA Fall Conference*, London, UK, 14.
59. Meli, R. 1997a. Early and extended Function Point: A new method for Function Points estimation. In *Proceedings of the IFPUG-Fall Conference*, 15–19 September, Scottsdale, Arizona.
60. Meli, R. 1997b. Early Function Points: A new estimation method for software projects. In *Proceedings of ESCOM 97*, Berlin, Germany.
61. Conte, M., Iorio, T., Meli, R., and Santillo, L. 2004. E&Q: An early and quick approach to functional size measurement methods. In *Proceedings of Software Measurement European Forum* (SMEF), Rome, Italy.
62. Caldiera, G., Antoniol, G., Fiutem, R., and Lokan, C. 1998. Definition and experimental evaluation for object oriented systems. In *Proceedings of the 5th International Symposium on Software Metrics* (METRICS 98), Nov. 20–21, Bethesda MD, pp. 167–178.
63. Teologlou G. 1999. *Measuring OO Software with Predictive Object Points*, Shaker Publ., ISBN 90-423-0075-2.
64. Abran, A. 1999. COSMIC FFP 2.0: An Implementation of COSMIC functional size measurement concepts. In *Proceedings of the 2nd European Software Measurement Conference* (FESMA'99), (Oct. 7), Amsterdam.
65. ISO. 2003b. *ISO/IEC 19761: COSMIC Full Function Points Measurement Manual*, v.2.2.
66. Meli, R., Abran, A., Ho, V. T., and Oligny, S. 2000. On the applicability of COSMIC-FFP for measuring software throughout its life cycle. In *Proceedings of the Escom-Scope*, 2000.
67. Pastor, O., Abrahao, S. M., Molina, J. C., and Torres, I. 2001. A FPA-like measure for object oriented systems from conceptual models. In *Proceedings of the 11th International Workshop on Software Measurement* (IWSM'01), Montr´eal, Canada, Shaker Verlag, pp. 51–69.
68. Kammelar, J. 2000. A sizing approach for OO-environments. In *Proceedings of the 4th International ECOOP Workshop on Quantitative Approaches in Object-Oriented Software Engineering*.
69. Standish Group, *The Standish Group CHAOS Report*, 2003.