EFES: AN EFFORT ESTIMATION METHODOLOGY

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF INFORMATICS
OF
THE MIDDLE EAST TECHNICAL UNIVERSITY

BY

SEÇKİN TUNALILAR

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY
IN
THE DEPARTMENT OF INFORMATION SYSTEMS

September 2011

Approval of the Graduate School of Informatics

Prof. Dr. Nazife BAYKAL
Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of Doctor of Philosophy.

Prof. Dr. Yasemin YARDIMCI ÇETİN
Head of Department

This is to certify that we have read this thesis and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Doctor of Philosophy.

Assoc. Prof. Dr. Onur DEMİRÖRS
Supervisor

Examining Committee Members

Prof. Dr. Semih BİLGEN	(METU, EEE)	
Assoc. Prof. Dr. Onur DEMİRÖRS	(METU, II)	
Prof. Dr. İbrahim AKMAN	(ATILIM, COMPE)	
Dr. Ali ARİFOĞLU	(METU, II)	
Assoc. Prof. Dr. Altan KOÇYİĞİT	(METU, II)	

ABSTRACT

EFES : AN EFFORT ESTIMATION METHODOLOGY

Tunalılar, Seçkin

Ph.D., Department of Information Systems

Supervisor: Assoc. Prof. Dr. Onur Demirörs

September 2011, 167 pages

The estimation of effort is at the heart of project tasks, since it is used for many purposes such as cost estimation, budgeting, monitoring, project planning, control and software investments. Researchers analyze problems of the estimation, propose new models and use new techniques to improve accuracy. However up to now, there is no comprehensive estimation methodology to guide companies in their effort estimation tasks. Effort estimation problem is not only a computational but also a managerial problem. It requires estimation goals, execution steps, applied measurement methods and updating mechanisms to be properly defined. Besides project teams should have motivation and responsibilities to build a reliable database. If such methodology is not defined, common interpretation will not be constituted among software teams of the company, and variances in measurements and divergences in collected information prevents to collect sufficient historical information for building accurate models. This thesis proposes a methodology for organizations to manage and execute effort estimation processes. The approach is based on the reported best practices, empirical results of previous studies and solutions to problems & conflicts described in literature. Five integrated processes: Data Collection, Size Measurement, Data Analysis, Calibration, Effort Estimation processes are developed with their artifacts, procedures, checklists and templates. The validation and applicability of the methodology is checked in a middle-size software company. During the validation of methodology we also evaluated some concepts such as Functional Similarity (FS) and usage of Base Functional Components (BFC) in effort model on a reliable dataset. By this way we evaluated whether these subjects should be a part of methodology or not. Besides in this study it is the first time that the COSMIC has been used for Artificial Neural Network models.

Keywords: Effort Estimation Methodology, Base Functional Components, , Artificial Neural Networks, Multivariate Regression, Functional Similarity,

ÖZ

EFES : EFOR KESTİRİM METODOLOJİSİ

Tunalılar, Seçkin

Doktora, Bilişim Sistemleri Bölümü

Tez Yöneticisi: Doç. Dr. Onur Demirörs

Eylül 2011, 167 sayfa

Efor kestirimi projede gerçekleştirilen maliyet kestirimi, bütçe planlaması, proje takibi ve planlaması, proje kontrolü ve yazılım harcamaları işlemlerinin tam kalbinde yeralmaktadır. Pek çok araştırma efor kestirim problemlerini analiz etmekte, yeni teknikler ve efor modelleri ile kestirimin doğruluğunu arttırmaya çalışmaktadır. Fakat, organizasyonların tüm kestirim çalışmalarını yönlendirecek geniş kapsamlı bütüncül bir metodoloji henüz tanımlanmamıştır. Efor kestirim problemi sadece hesaplama değil, aynı zamanda bir yönetim problemidir ve kestirim amaçlarının, işlem adımlarının, ölçüm metotlarının ve güncelleme mekanizmalarının uygun bir şekilde tanımlanmasını gerektirir. Ayrıca proje ekiplerinin güvenilir bir veritabanı oluşturma amacı ile sorumlulukları ve motivasyonu olmalıdır. Böyle bir methodoloji olmazsa, şirketin yazılım ekipleri arasında ortak fikir birliği oluşmaz ve ölçümlerdeki farklılıklar ve toplanan bilgilerdeki farklı dağılımlar doğru model üretebilmesi için doğru ve yeterli data toplanamamasına sebep olur.

Bu çalışma, efor kestirimi için bir metodoloji önermektedir. Metodoloji, şirketlerin en iyi pratiklerini yayınladıkları raporlar ile önceki çalışmalardan elde edilen veriler, problem ve çatışmalara önerilen çözümler üzerine kurulmuştur. Beş entegre süreç; ilgili çıktıları, kontrol listeleri, prosedürleri ve şablonları ile tanımlanmıştır: Veri Toplama, Büyüklük Ölçümü, Veri Analizi, Kalibrasyon ve Efor Kestirimi süreçleri. Metodolojinin geçerliliği ve uygulanabilirliği orta büyüklükte bir organizasyonda geriye dönük olarak gerçekleştirilmiştir. Geçerliliğinin değerlendirilmesi sırasında Fonksiyonel Benzerlik yöntemi ve fonksiyonel alt-parçalarının efor modelinde kullanımı da güvenilir bir veri-seti ile sınanmıştır. Böylece bu konuların metodolojinin bir parçası olup olmayacağı netleştirilmiştir. Ayrıca ilk kez COSMIC ölçümü Yapay Sinir Ağları modeli ile birlikte kullanılmıştır.

Anahtar Kelimeler: Efor Kestirim Metodolojisi, Fonksiyonel Büyüklük Parçaları, Yapay Sinir Ağları, Çok Değişkenli Regresyon, Fonksiyonel Benzerlik

DEDICATION

I have been truly blessed to have such wonderful parents.

Dear Mom, Dad..

Your love, support and encouragement led me through this amazing journey. I could never have completed this degree without you

I Love You and I dedicate this thesis to you ...

Life is an opportunity, benefit from it..

Life is beauty, admire it..

Life is bliss, taste it..

Life is a dream, realize it..

Life is a challenge, meet it..

Life is a duty, complete it..

Life is a game, play it..

Life is a promise, fulfill it..

Life is sorrow, overcome it..

Life is a song, sing it..

Life is a struggle, accept it..

Life is a tragedy, confront it..

Life is an adventure, dare it..

Life is luck, make it..

Life is too precious, do not destroy it..

Life is life, fight for it..

ACKNOWLEDGEMENTS

The completion of my PhD would not have been possible without the encouragement and support of my supervisor, family, friends, and colleagues. It was a long journey and many people contributed to the development of this thesis through their gift of time, advice and belief.

I would like to express deepest gratitude and sincere respect to my supervisor, Dr. Onur DEMİRÖRS. I'm heartily thankful to him for his expert guidance, his patience, constructive criticism, careful supervision and continuous encouragement throughout the years of my study. It has been a privilege and honor to work with him.

I am grateful to my committee members Dr. Semih BİLGEN and Dr. Ali DOĞRU, Dr. Ali ARİFOĞLU, Dr. Altan KOÇYİĞİT and Dr. İbrahim AKMAN for their time, helpfulness, encouraging words and also for their critical suggestions and advice. Their criticism and suggestions effectively helped to shape my work.

I am appreciative of the support from my colleagues in the Informatics Institute: Semra YILMAZ TAŞTEKİN, Dr. Çiğdem GENCEL, Dr. Pınar Onay DURDU, Dr. Yasemin KARAGÜL. I am grateful for their friendship and collaboration. Without them my adaptation to institute would have been much more difficult.

I am thankful to the members of the Software Management and Research Group for many brainstorming meetings. My special thanks goes to Barış ÖZKAN, Özden Özcan Top, Erdir UNGAN and Gökçen YILMAZ for their invaluable discussions and for the validation of this study.

I also appreciate insightful discussions with my friend, who is also my work colleague, Gamze EROĞLU. She always reviewed my study and offered her honest thoughts and critisim. I would never forget the help I got from her during difficult times in preparation of this study.

I am indebted to my old friends and work colleques Alptekin ISIKLAR, Dr. Devrim ANIL to support me and for creating an enjoyable and motivating work environment. They always had time to discuss issues I was struggling with and always provided me with valuable insights and solutions at each step of this study.

I would like to give great thanks to my friends, particularly, Dr. Handan YÜKSELGÜNGÖR, Füsun DEMİR, Figen BOZKURT, Sadiye BAKTIR who have given lots of laughter and confidence in completing this thesis. I have been fortunate to have their friendship.

I would like to thank all the staffs and faculty members of the Informatics Institute for assisting me in various ways.

Finally, I wish to thank my family for their endless love, tolerance, encouragement and support. My parents, Ayten and Ömürcan TUNALILAR, without whom, I would not be the fighter I have become; nor learned to go after my dreams regardless of situation. I cannot adequately express my gratitude to my dear brother Tuğrul and beloved sister Nurlan TUNALILAR for proof reading of the entire thesis and my publications. Their patience, motivation and insightful comments undoubtedly resulted in significant contributions to the development of this thesis. Warm thanks to my amazing niece Burcu TUNALILAR, her smile always lightens up my days.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

Abbreviations	**Definition**
AI	Artificial Intelligence
ANN	Artificial Neural Network
AQAP	Allied Quality Assurance Publications
AVE	Average
BFC	Base Functional Component
BSP	Board Support Package
CBR	Case Based Reasoning
CDR	Critical Design Review
CFP	Cosmic Function Point
CM	Configuration Management
CMMI	Capability Maturity Model Integrated
COSMIC	North Atlantic Treaty Organization (NATO) Security Category
COTS	Commercial Of the Shelf
CSU	Configuration Software units
DET	Data Element Type
DO-178	Software Considerations in Airborne Systems and Equipment Certification
DOD	Department Of Defense
EFES	Effor Estimation Methodology
EIA	Electronic Industries Association
EIF	External Interface File

Abbreviations	Definition
ES	Embedded Systems
ESA	European Space Agency
FFP	Full Function Point
FISMA	Finnish Software Measurement Association
FP	Function Points
FPA	Function Point Analysis
FS	Functional Similarity
FSM	Functional Size Measurement
FUR	Functional User Requirements
GUI	Graphical User Interface
HW	Hardware
I/O	Input/Output
IEEE	Institute Of Electrical And Electronics Engineers
IFPUG	International Function Point Users Group
ILF	internal logical file
ISBN	International Standard Book Number
ISBSG	International Software Benchmarking Standards Group
ISO	International Standards Organization
KLOC	Kilo Lines of Code
LED	Light Emitting Diode
LOC	Lines Of Code
LSR	Least Squares Regression
M&A	Measurement and Analysis Group

Abbreviations	Definition
MIS	Management Information System
Mil-STD	Military Standard
MMRE	Mean Magnitude Relative Error
MRE	Magnitude Relative Error
MSE	Mean Square Error
NESMA	Netherlands Software Metrics Association
NN	Neural Network
OLS	Ordinary Least Squares
PDR	Preliminary Design Review
PRED	Prediction Level
QA	Quality Assurance
R&D	Research And Development
R^2	Coefficient Of Determination
RET	Record Element Types
RTOS	Real Time Operating System
SCI	Software Configuration Item
SDD	Software Design Document
SEER	Software Effort Estimation Research (Company Name)
SEERSEM	Software Estimation Modeling Tool
SLOC	Source lines of code
SPR	Software Productivity Research
SRS	Software Requirements Specification
UK	United Kingdom

Abbreviations	Definition
UML	Unified Modeling Language
VAR	Variance
WBS	Work Breakdown Structure

CHAPTER 1

1. INTRODUCTION

Measurement is a fundamental part in every scientific and engineering discipline that enables the information to be collected and be used for qualifying the product and process. It enables to understand the current situation, allows making comparison with previous projects, and leads to making a good estimate. Results of measurement remind you of where you have been and where you plan to go.

Project managers' main concern is related to how long their projects will last, what the projects will cost, and whether the product will function as intended when released. In order to answer these questions, they need measurement methods.

Effort estimation is at the heart of these questions, since it connects measurements to cost and schedule of the project. The determination of the required effort during project initiation phase allows us to plan adequately any forthcoming activities. It is performed for a variety of reasons: project selection, resource planning, scheduling, monitoring status, assessment of team performance, controlling remaining activities etc. Compared to other engineering disciplines, accurate estimation is difficult for software development since it involves a number of interrelated factors that affect development effort and productivity, like environment, team skills, tools, properties of product. Therefore effort estimation models have been investigated by many researchers and practitioners since 1960s [1][2][4].

During the years, the focus for the studies on effort estimation has been changing. Researchers analyze the reasons for estimation errors [6][21][22][23][24], the ways of improving accuracy in estimations by using new size measurement methods or model building approaches [16] [17] [18], and the solution of dataset-related problems [12][13]. However there is no defined estimation methodology to guide companies in their effort estimation tasks. "Methodology" is a comprehensive approach. It not only defines the methods or techniques applied for measurement, but also the execution steps, updating mechanisms, procedures, artifacts, templates etc that prevents errors.

In the early days, for effort estimation expert judgment techniques were very popular. In 1966, Delphi method was suggested by Helmer [5]. This method was recommending forming a group from experts to predict an effort estimate. It was improved by Boehm in 1981 and named as Wideband-Delphi method. It is an inexpensive and easy method, but may suffer from manipulation of people [7].

In 1970s methods to organize and to structure the project related information became popular like top down and bottom up methods. Bottom up approach needed the system and its components to be known sufficiently. Top down approach needed very similar projects to be accurate and to create cost information database [30][31].

At the beginning of 80s, the estimation equations were generated that were based on historical data and statistical analysis techniques like regression analysis [3] [7]. In most of these effort models it was assumed that size could be measured accurately and be used to estimate the effort needed. Models were typically using Kilo Lines of Code (KLOC) as size parameter. Even it was an easy measure to obtain, it had some drawbacks: it was not available at the beginning of the project, definitions were controversial, depended on the developer performance etc [33].

IBM realized that, complexity of software projects created a necessity to do the work in defined life-cycle phases or tasks and in a controlled manner. When they discovered that documentation task had a tremendous effect on cost overruns, Alain Albrecht and his colleagues were assigned to develop a new measure, that also could count the non-coding work. This study was the first Function Point measures study [32]. Then a number of size measurement methods have appeared: such as IFPUG, MkII, Cosmic FFP, Unified Modeling Language (UML) based functional size etc [34][35][36][37]. These size measurement techniques were either proposed according to the changing development environments or modified in order to resolve difficulties that had been raised in preceding models. Although some of these measurement methods have gained raising popularity in the industry, deficiency in applicability to all software domain types have caused to emerge such new variations.

To mention a few: In 1988, Symons devised his concerns and difficulties with Albrecht's FPA, and proposed a new variant called Mark II Function Point Analysis [38]. He found Albrecht’s method classification very simple, the origin validity and objectivity of the weights very doubtful and the handling of internal complexity as very complicated. Now this method is an accepted and applicable standard by United Kingdom. However, its usage for real-time systems is found limited [39]. Similarly, in 1986, Software Productivity Research (SPR) Practitioners recognized that some type of applications did not show any beneficial results since function points were originally developed for the measurement of MIS type systems. They developed “feature points” method aimed to deal with real-time process control, mathematical algorithms and various embedded systems [40]. Due to difficulty in assigning and deciding weights of the complex algorithms method was not caught on [41]. Another size measure, Object Points, was suggested for Integrated Development Environments that use objects for development like screens, reports, modules etc [42]. By using Delphi estimation technique the weights of different objects was decided. However, there still a shortage of historical database for this measurement [43].

In 1999, The COSMIC group reviewed some of the existing methods. After studying the commonalities of existing methods, the method was proposed as a new generation of software functional size measurement (FSM) method. Due to its applicability to business type applications, real-time software and hybrids of these, applicability for software in any layer and applicability at any phase in project life-cycle made it very popular in industry [36]. A more detailed history and definitions for FSMs are given in Chapter 2.

Other than LOC and Functional size measurements, some new size measurement methods like UML points [44] and Used Case points [45] were suggested by some studies in 2000s. The reason for these new techniques was the trend in development tools. These tools provided graphical notations and frameworks for object oriented analysis & design used for describing the systems at the initial phase of the projects.

In the mean time, that was at the end of 90s, limitations in dataset characteristics and the aim to incorporate expert knowledge into models caused researchers to find better building approaches. Data analysis methods have been applied to effort estimation area for creation of an effort model and compared: such as case-based reasoning, classification &regression trees, neural networks, bayesian networks [14][16][17][18]. But the results of these studies diverge. Berlin and Raz [46], Stensrud [47] and Briand [48] identified regression based models as more reliable compared to ANN networks, however Tronto and Silva recommended the usage of neural network based models [49]. Gray and MacDonell found neural network outperformed compared to fuzzy-logic [50]. Finnie and Wittig [14] compared Case Based Reasoning (CBR) with different regression models using FP and ANNs on a large database. They reported a better performance of CBR when compared with different regression models based on function points. ANNs, on the other hand, outperformed the CBR approach. Finally, in a new study Marza and Teshnehlab [51] proposed a neuro-fuzzy approach. They claimed that this method had a superior performance than multivariate regression, neural network and pure fuzzy approach.

Results of these studies are valuable and have considerably positive effect on effort estimation process. The differences on results can be attributed to the specific data-sets they have utilized. Each recommended solution on these studies is constructed by the assumption that, their historical data is very reliable. But without a reliable database, these solutions are neither repeatable, nor usable even for the different teams of the same company.

In 2000s, Software community mostly focused on data collection problems [8] [9] [10] [12] [68]. Some questions raised on this subject were; whether cross-company data-sets were usable, how the missing data problem could be solved, which attributes of projects would be more practical to collect for cross company datasets etc. Most of the companies apply the results of cross-company datasets, since data collection is a time-consuming process. However, the advantage of cross-company data sets is still under debate [8] [9] [10] [11] [12] [13]. Variance in processes, tools, lifecycles, collected information of different companies prevents to use these datasets for building effort models. Using single company data will minimize some differences in processes and tools. However, if companies don't have well-established standardized processes, and common interpretation is not constituted among software teams the same problems in cross-company cases will be observed.

The classical method for estimations is to collect size measurements like LOC or FPs as projects are deployed and use those measurements into a statistical relation for building method. The method is certainly valuable and has received substantial attention in the Software Engineering community. However, if processes are not managed well, then information for estimation could not be inferred well across different project teams, different processes, variable development technologies and environments. It would be very difficult to identify factors that drive the significant and relevant effort in a meaningful way, if only limited reliable historical project data is available [52]

Heemstra [53] explained this problem from another point of view. He claims that estimation accuracy problem is not only technical but also a managerial problem. Therefore handling this problem by using computational models is not enough. Managerial problems like creating motivation and commitment to success should also be addressed. He advised, putting managerial decisions related to the estimation goals in real-life, the ways that estimation processes are carried out and the proper definition of the updating mechanisms.

In this study we developed a process methodology that enables us to manage core processes of effort estimation approach. This methodology will provide us to build meaningful and reliable information for effort estimation while removing inconsistencies and problems in related processes and data. To create "meaningful" data, attributes of the projects are previously defined in processes. To remove inconsistencies “review and validation meetings” are included. The well established processes are aimed to create common interpretation among software teams.

This thesis investigates the problems in building a relation between size and effort, the solutions to the problems that prevents the constitution of reliable information and proposes an Estimation Methodology, EFES. EFES methodology includes the explanations of activities in processes, their related procedures, artifacts, checklists, templates etc. for an organization such that, companies may implement a similar or modified approach easily.

The thesis includes a multiple case study to analyze the problems by using empirical analysis method. In the first case study: we investigated some concepts defined in our methodology and evaluated their benefits by making empirical researches to decide their inclusion. Primarily two subjects “Functional Similarity” and “Base Functional Component usage” in effort models were evaluated by using a sample dataset. By inquiring their results in the second case study validation was performed by applying methodology on a large project dataset.

In the following paragraphs of this chapter, first the context for the effort estimation process will be given, then the problems we aimed to solve will be summarized, and finally our solution approaches will be explained.

1.1. The Context

The estimation of work effort is at the heart of tasks, since it is used for purposes such as cost estimation, budgeting, monitoring, project planning and control and software investments analysis [54] as it is shown in Figure1.

Projects have cost and schedule overruns. Even the studies report different percentages; the trend is the same, most of the projects encounter overruns in terms of cost and schedule. In some surveys executed in 1970 and in1994, average schedule overrun values are announced as 41% to 258% , and cost overrun values from 97% to 151% [55][56]. A similar tendency was found by the Standish Group. They reported that 84% of projects completed after the planned schedule [57]. In another study performed by Molokken and Jorgensen it was announced that 60-80% of the time 30% more effort is needed [23].

Estimation is a complicated process with errors, however it turns out to be an advantage if it is executed in an appropriate way. Results of Standish Group’s CHAOS surveys for the years 1995, 1998, 2000, and 2002 show evidence of improvement in estimations. The percentages of cost overruns were noticed as decreasing in those years: 142%, 69%, 45%, and 43%. Based on these results Molokken and Ostvold points out that even improvement is issued, accurately estimating projects is as difficult and challenging today as it was thirty years ago [58].

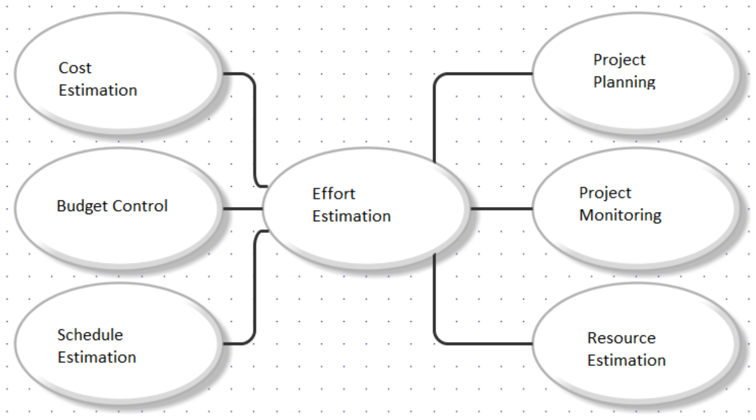


Figure 1 Effort estimation is at the core of Project Tasks

The contract between company & customer is based on mainly cost and schedule estimates. Cost estimate is vital for both companies and customers. It is a continuing activity which starts at the proposal stage and continues throughout the development of the project. It can be used for the evaluation of projects, feasibility analysis for proposals, contract negotiations etc. Underestimating the costs at the proposal stage may result in underdeveloped functions of the product that are not tested well. This results in poor quality of products. Overestimating on the other hand results in losing the contract at the proposal stage. A large percentage of cost comes from the personnel expenses. Therefore even the aim is the “cost estimation”, the output values are not usually represented in terms of currencies [59]. The cost estimation parameters are often predictions of the expected effort needed for the development of the project and the required time to finish it. Based on the effort information, cost for the development is estimated, and then other cost issues are appended for the final calculation.

The descriptions of the cost and effort estimation, project planning, project monitoring and control concepts are not separate or discrete. All concepts are applied during the project life-cycle for the evaluation of project status and for the decision of future actions. When we have an initial effort estimate of the project that is to be developed, we have to define a project schedule that is in line with that estimate. In other words, the output of the effort estimation process is one of the project planning inputs.

Project planning is the process of selecting a strategy to produce the aimed products and defining the related activities to be executed, to attain this goal and deciding the timing and overlapping of these activities. Project manager selects the right personnel for task assignment, tracks and reviews the results, and makes appropriate and necessary modifications on the plan when it is required. Therefore schedule and resource plans also are build upon the results of effort estimation.

Once the effort decided that is required for a project, it is possible to assign resources to determine how long the project will take and estimate human resources and other resource costs. The schedule for the project can be arrived at based on the activities, interdependence of these activities and effort required to complete each of them. Resources would include human resources, computer resources, and monetary resources, therefore apart from effort need, adjusting resources may affect the schedule. The models for software resource estimation are tied to specific cost and effort models developed and the improvements in sizing techniques [60][61].

When project planning is complete, we start to monitor the project. Therefore, the inputs of the monitoring process will be project estimate and schedule, plus the real data collected as the project evolves. If we fail to keep to the original schedule, the project can be re-planned during the monitoring process. For example, a change in the project can result in a large deviation during monitoring. In this case, we will have to re-estimate and re-plan the project accordingly. To succeed in a software project, project manager compromises three parameters, “resource”, “features of the product”, and “schedule” to comply with the plan. If any one of these has a change the others must be modified and the plan has to be tailored.

Projects normally have a “budget” and continuous control and cost calculations are necessary to ensure that expenses are in line with the planned budget. At any time during the project the amount of the remaining effort is estimated for controlling the budget plans. By using these estimations, during the development of the projects, companies classify and prioritize their projects, determine what resources to commit to the project and how well these resources will be used [62]. In other words, during cost estimation, cost of each task and product are estimated considering the development model and historical data. For construction of the budget this information is embraced with the major milestones of the project, scheduling assumptions and constraints, predecessor/successor relationships of tasks.

One of the significant subjects of effort estimation research in software engineering is related to the size measurement since it is considered as main driving parameter in effort models [63][64]. Frequently, effort estimation requires measuring the size of the software in source lines of code (SLOC), Function Point (FP) etc. Functionality based size measurement methods excel due to their early availability and independence of the language, tools, techniques and technology utilized. The FP based measurement methods generally was based on counting the BFCs of software such as inputs, outputs, inquiries, logical files and interfaces.

1.2. The Problem

For most of the effort estimation studies, size is the primary driver . Therefore, in literature a number of effort estimation model proposals exist, that relates the size and other parameters with effort [3][25][65][66][67][68]. On the other hand many researchers also accept that: despite ongoing efforts there is not a single model defining size-effort relationship that would provide accurate estimates and that would be accepted by range of software projects and organizations [70] [71] [157].
If we group the researches that are searching for solutions to the relation problem, four groups of subjects appear that are highly interrelated as shown in Figure 2 :

- "Supporting & Extra Effort" that creates problems in planning or controlling the projects.
- "Data Reliability" and "Formation of Dataset
- "Size Measurement" problems and improvement
- "Effect of Selected Data Analysis Method" on building Relationship

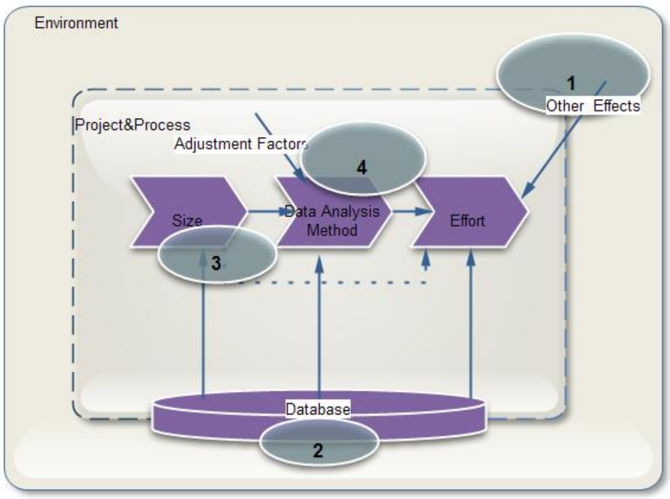


Figure 2 Effort Estimation Problematic Areas

In this figure effort is shown as a function of the measurable artifact of software: *size*. A kind of data analysis technique produces a functional equation for representing their relationship. For this purpose, classical approach is used: data from past projects are collected in a database to be utilized by specific analysis tools.

Most of the studies in literature concentrate on only one of the groups on this figure and try to bring out an applicable solution for a specific dataset. For example, in order to improve the accuracy of the effort model, many researchers have been addressing the new size estimation techniques that results better effort correlations. We have shown them in region 3 in Figure 2. Similarly, several ways of data

analysis like ANN, Fuzzy, Bayesian networks are evaluated and compared to find best handling method [48][49][50] as depicted in region 4. Each recommended solution is constructed by the assumption that, the historical data is very reliable. But without a reliable database, these solutions are neither repeatable, nor usable even for the different teams of the same company. Studies on reliability of database are pointed in region 2. Region 1 indicates, "Other effects" that effect project's execution and data, especially "supporting and extra" effort that is not a function of the size of the projects: such as unplanned tasks, unexpected problems or overlooked activities that effects project's execution.
In following paragraphs, we presented a summary of related studies on these subjects:

"*Supporting & Extra Tasks" creates problems in planning or controlling the projects*.

In literature many studies warn us about the unplanned activities, tasks, unexpected problems that are included in effort value but not as a part of the development process. Phan [72], made a research by interviewing with the company participants for the reasons of this overruns. Phan's results were as follows: cost overruns were most often caused by over-optimistic estimates (51%), changes in design (50%). For the schedule overruns according to the respondents optimistic planning (44%), and major (36%) and minor (33%) changes of specifications were the reasons. McConnell [26] proposes that 20-30% of the effort estimation errors come from not estimating certain activities. Genuchten also investigated problems that alter project plans [19]. His findings are: overruns in previous activities, interruptions of maintenance activities of other projects, late delivery of hardware components, insufficient hardware, project staffing etc. Jorgenson et al. [23] made a statistical analysis of relations between characteristics of the projects and effort estimation errors. They also reported: unexpected events and overlooked tasks, change requests, resource allocations are found as the problems of effort estimation.

"Data Reliability" and formation of Dataset effects estimation accuracy

An organization can use several ways for effort modeling. One of them is to use "generic models' that are proposed in the literature or are in-use by software community. Another way for effort estimation is to use historical data of the organization and generating "specific models" that have been retrospectively established for the company. Most of the time the technologies used for the development of the software product undergo significant changes from one generation to the next. This means that in engineering software, we frequently design and maintain systems of unprecedented size and complexity in a rapidly changing environment or with rapidly changing tools. To deal with this situation, if company does not have enough historical data, utilizing cross-company or benchmarking datasets may be possible since data collection is a time-consuming process. However, data collection methods are not uniform across different companies. The differences in the processes and practices are not reflected well in these datasets.

A number of comparison studies exist for cross-company and single company datasets. Mendes and Lokan [68] performed an analysis on cross-company database. They found that even only 21 parameters collected in International Software Benchmarking Standards Group (ISBSG) had an effect on effort, 88 different variables were collected that were not applicable nor practical. They also reported that % 40 missing information prevented to perform good analysis. Dery and Abran [9] executed another analysis on the ISBSG dataset, and reported the missing data problem again that led to smaller usable samples for further analysis. Besides they described some contradictory recorded information for phase efforts. Abran, Symons and Oligny mentioned that, project phases of some companies and the ones defined for ISBSG dataset didn't correspond to each other [73].

The term 'Reliable data' is used to mean that collected information for effort data, size and other sound measurements are consistent and repeatable across the organization, in terms of personnel, projects and processes [74]. For the collected data some studies have recommendations. For example, to make a better resource management, Briand and Wieczorek proposed the characterization of effort expenditures by recording the work spent for each phase and activity [61]. Yang and Bohem found different effort distribution patterns for Chinese software than COCOMO, and claimed that poor effort estimation reason is these process variations. They suggested companies to recognize and understand these process variations [75]. Wiegers, similarly, suggested tracking trends in effort expenditure in work activities for future planning and estimation of tasks [76].

Size measurement problems and improvement

The question about how size should be measured and represented in effort equation is seen as the main problem for solving relationship problem. To solve this issue, many studies proposed using different size measurement techniques, but each solved the problems of different application domains or development environments. For example for MIS type systems MKII size measurement is a valuable way whereas for real-time systems COSMIC is suggested [39][77]. In fact, their strengths and weaknesses are often complimentary to each other. One improvement in size measurement methods is using a scale type for the development complexity among functional components. Some measurement methods assign some values for these components that represent their respective contribution to total effort. For example, components in the FPA method are assigned as the low, average and high in terms of complexity. However, the addition of these components, to reach final functional size and consequently effort has no meaning, since they are only a ranking. MKII on the other hand applies relative weights to differentiate the productivity differences required for each sizing component. However these values are fixed and not arranged according to the environment, conditions or selections of the project. These weights for MKII were diverted only once a couple of years ago according to an industrial project database [78], and it is not applicable to several types of projects or different companies. For the Cosmic FFP, that allows measuring each component and level separately, there is no such assignment for components. So the development difficulty of each component due to project characteristics or environment etc. is not reflected in size measurement.

It is well known that, even certified measurers may encounter conflicting situations. If measurements are experimented independently by different people, or by the same measurers at different times, results may not be very repeatable. The Company, Renault, [27][28] used COSMIC method and made it mandatory to use for size measurement for its embedded teams and its subcontractors. In those reports, Stern et al, announced that, they wrote COSMIC measurement textual guides, defined very clearly and without any ambiguity, the way of measuring functional sizes. They reached a 1% error reproducable measurements by using these guidelines even measurements performed independently by different people. To improve measurement repeatibility, Khelifi and Abran [29] presented a way for developing a standard etalon for the software size measurement and illustrated it by using COSMIC. They proposed a template that included the whole information for Cosmic measurement process. They claimed that the usage of software measures should be integrated in a complete process of verification to improve the quality of measurements. The focus of their study is developing a methodology for size measurement part of effort estimation.

Although the functional size of software product can be measured precisely, there are still difficulties in measuring the functional size, which directly affect the magnitude of effort required to develop software products. One of these difficulties is related with the reusability of software entities. During last decade, several studies examined the concept of functional similarity [79][81] for distinguishing similarities, quantifying reuse potential and investigating similarity‘s effect on development effort [183][184]. But this structural aspect of the software and its influence on development effort has not been well reflected in existing estimation models.

Effect of selected Data Analysis method on building Relationship

Finally, as the last group of studies, in order to increase the accuracy of the estimation, several methods are evaluated and compared. Most of the studies found ANN network as better method compared to the others [14][49][50]. The major differences are their different datasets, the selected inputs for analysis methods and processes. Kitchenham and Kansala [82] suggested that Function Points were not well-formed measures because there was a correlation between their constituent elements. By using multivariate regression they found that the best fitting equation included only two elements: input and output function points. Gencel and Buglioni [83][84], compared the effort estimation based on BFC types of COSMIC by applying multivariate regression analysis, and found that estimation accuracy was improved. Even ANN has been announced as a valuable tool, up to now, ANN method usage with BFC types is not applied in effort data analysis.

Although size and effort estimation usage has gone a long way, and a lot has been learned by industry, significant improvement studies are still needed for the successful implementation by organizations. However concentrating on one group in figure does not solve our problem. Since these are not separate processes, we need a complete applicable solution for software organizations.

1.3. The Solution Approach

Many studies concentrate on only one of the regions in Figure 2. However the problem is not solved by the solution of these regions. All regions need improvement and consequently improvement needs quantitative data. The quantitative data in literature or in cross-company datasets has some problems due to the different execution or understandings of companies [8][9][10]. Data analyzers and users of these databases encounter the following problems: "data is not sufficient", "data is not correct", "data is missing", "data is not tailorable for current technology". On the other hand companies that provide data claim "definition of parameters is complicated" "overweight number of parameters is expected to be provided", "project phases does not match to our case", "we didn't collect that parameter" etc.

So, we need a solution to prevent these problems, to obtain reliable and better data in terms of quality and to improve estimation performance: To do this

- We need to create common aims and understanding in data collection,
- We should develop control and communication mechanisms among collectors, users and data providers.
- We need a set of guidelines and standards not only for size measurement techniques, but also for all other steps of effort estimation.

Software measurement and estimation processes are mentioned under the Capability Maturity Model Integrated (CMMI) [185] concept in general. For example an apparent "need" for effort&cost estimation method exist at the beginning levels under "Software Project planning" process area. At more advanced levels, that is at level 4, Quantitative Process Management is defined, that recommends the use of "past estimation data". These concepts and requirements are commonly known. Actually, the standards like CMMI, ISO 15539-02 [179] only defines the necessity of measurement and analysis concepts. "How should these process and activities be executed is not defined in those standards. Therefore operational part and implementation belongs to the company itself. ". For effort estimation that is at the heart of all the tasks, the planning and management activities, the existence of a specified execution way and assured data is a necessity. Otherwise the consistency and reliability of the data is not possible. However in literature, a fully defined methodology that defines "how to do this" still does not exist, that is in parallel with the accepted and applied process and life cycle standards.

Therefore, we developed an “EFfort EStimation” (EFES) methodology to integrate the necessary measures, tools, techniques, and heuristics in form of processes for software organizations. Under the subject of measures, we mean information used to establish a common understanding of processes and project like attributes of effort collection, parameters of projects, defined measurements for size etc. To prevent misunderstandings and for the benchmarking of following studies, we selected IEEE-12207 [155] “Software life cycle processes” as a standard. Our methodology provides a comprehensive solutions to size and effort relationship problems with worldly accepted lifecycle processes and best practices in literature. By using the “phase definitions” of IEEE 12207 and by studying the reasons of errors in effort estimation, we created special data collection mechanisms. By investigating the improvements in functional size measurement and data analysis we developed analysis, tailoring and calibration requirements.

In order to deal with the problematic areas shown in Figure 1, EFES methodology covers all regions with integrated processes. We searched literature to find out how we can resolve problematic situations mentioned in Part 1.2. In the Table 1 we summarized the problems, related requirements in methodology, and EFES’s integrated solution to this problem.

Our methodology eliminates the problems and major differences in executions of different teams, enables repeatability in effort estimation by defined guidelines. For this aim it provides Asset Library by defining component types and templates to prevent overweight data, to perform reviews and to guarantee the quality of the data.

EFES methodology definition provides a group of processes with their inputs, outputs, steps, roles and responsibilities, tools and techniques. So before execution “what will be done” “what will be measured” or “what will be analyzed” are agreed by development team and analysis & measurement group. Any company will follow those basic steps with minimum effort.

Creation of different processes will provide flexibility for further improvement. By this way, methodology enables to merge new improvements in size measurement and data analysis without re-defining the whole methodology and related assets. Methodology includes review, control points to improve accuracy and to check problems during execution. By this way, calibration of assets according to the requirements of new projects is included in methodology.

The methodology includes detailed definition of related processes with their procedures, assets, tools and measures to be used in these processes. The processes are generic enough so that different organizations can tailor and integrate into their development practices and are specific enough to enable consistent implementation.

Table 1 Solutions to problems

Problem	Methodology Requirement	Solution
Unexpected problems and activities, unplanned tasks, requirement changes, staffing problems effect project timeline, schedule, effort etc. Some of the expended efforts for tasks(e.g.: documentation, training) or overlooked tasks (meetings, demos)have no relation with functionalities,	The rate of such subjects should be recorded for future plans. Effect of these items should be reflected in effort estimations. Development Effort and Supporting & Extra effort should be differentiated.	Prepare a predefined Effort Collection template. Use Data Analysis tools for analysis of their effect and include them at the final calculations.
Team responsibility and motivation is necessary	Development team should be a part of data collection processes.	Data collection should be periodically performed by project personnel and controlled by measurement group.
Data collection is not uniform among software teams, missing data problems or contradictory information appear in dataset.	Activity and phase information applicable to all projects should be defined and controlled by responsible people periodically for removing ambiguities.	Predefined WBS template is needed for Activity or task list. A group should be responsible for managing the methodology.
Different size measurement components have different effect on total effort.	Functional Component Effects should be represented in model that represents company statistics.	A specific tool may be used, that uses a number of BFC components as input. Effort model is constructed based on this analysis.
Different size measurements may appear by different size measurers	Repeatable measurements should be satisfied	An additional textual guide will be defined. Validation is performed by discussions among measurement group. Measurements should be written in templates for future uses.

Table 1 (cont.)

FS Consideration is not included in effort models	Similarity can be analyzed and included in models by using size measurements.	An updated size measurement table will be produced for FS inclusion and used as input to Data Analysis tool.
Data Analysis tools have contradictory results for different datasets.	Analysis tools may be compared for this dataset. Company selects the best applicable solution	Preparing a common measurement template enables to make comparison of analysis tools.

There is no requirement for companies to be mature enough to apply this methodology. All assets are specifically defined and can be directly used by any type of company. Without changing the guidelines, procedures, analysis tools, company may update or tailor its specific needs on assets and may form its own datasets. With this methodology the aim is not only collecting “data” or measuring the software for analysis, the aim is to identify and extract the “core” meaningful information for the company that is sufficient to make an effort estimation accurately and to apply required techniques and developing mechanisms for doing this.

We performed an exploratory case study to define the requirements of this methodology. While deciding some concepts of size measurement and data analysis we performed empirical investigations on small datasets. After the development of methodology we validated our results with another data set.

1.3.1.The Effort Estimation Methodology

In order to cover all the regions in Figure 1, specific processes that are integrated with each other are needed. In the following paragraphs, our comprehensive approach that we embrace the processes, tools and artifacts are summarized.

Data Collection Process: We need the correct and consistent information to be collected. To do this, a detailed predefined WBS structure that defines project activities and tasks will be necessary as an asset for the process. Then, by using this structure all efforts for different phases and tasks, unplanned, unexpected or unwanted activities should be recorded and tracked for future analysis. In addition, effort expended for demo, documentation, training tasks etc. should be recorded by development team. This WBS structure not only includes project related effort data, but also allows other types of efforts, that the project team spends. Such effort collection process and analyzing the available effort data combine the advantages of expert estimation techniques and computational effort estimation methods in effort modeling. If tailoring of the WBS structure according to the project’s needs and initial forecasting are done by employees in the project, who are responsible for the development; they will probably get to know the project better than anyone else, collects and controls their efforts with a higher motivation. In order to build that predefined-WBS structure and other information, a worldwide standard for software development will be used: IEEE 12207 [155]. By this way a common understanding, not only among teams at the organization but also at the subcontractors is also created. Project team record their efforts on appropriate items periodically during the project.

A responsible group for overall EFES methodology will periodically check the appropriateness of the recorded data. EFES methodology only defines the activities, tasks and other information for data collection. Selection of the tool for the process depends on the company. A basic excel sheet will be enough for data collection which is used in our validation case. However, companies may prefer to execute this process on a MIS based tools.

Size Measurement Process: To satisfy measurement quality, standardization and repeatability: we need a measurement guidance to be defined as an asset of the process. This guidance text provides the measurement concepts to be understood well. It is important that the assumptions and formulas are documented when the size measurement of the project is completed. This will enable more thorough review and will make it easier to revise later. The guidance should include tailoring concepts for specific application domains.

Size measurement can be applied at any time, either at the feasibility or during the project. However, the quality of the measurements depends on the quality of the specification documents. Therefore, for the pre-review of the specification documents, a review checklist would be a valuable asset for starting the process.

An expert group for size measurement improves the quality of the measurements and the requirements by applying the review and the validation process. This expert group can manage all activities in EFES methodology.

Data Analysis Process: To deal with the analysis of functional size components and to investigate the variable effects of these components on effort model, ANN or multi-variate regression analysis can be used. There exist several commercial and free tools for statistical analysis, fitting and calibration of software effort models. Responsible personnel will be the same as the size measurement group. They should be trained on the usage of these tools to create better models

.

Data analysis process is applied at company level to produce effort models. Other than the effort models created, one other artifact generated by this process is the analyzed percentages of all the types of efforts, especially supporting or extra tasks. Based on the WBS based collected information of all projects, distribution percentages for overlooked tasks, unexpected events etc. are calculated For example: effort percentages for hardware (HW) related effort, for documentation task, for requirement changes, or for specifically applied standards are analyzed. Their effects will be reported either at the company or at the application type level.

Calibration: Since EFES methodology requires continuing processes, information in WBS template and effort models should be updated based on the new or finished projects data. For example if new projects requires a specific procedure or standard to be applied in projects, WBS structure should be reviewed and updated.

Effort Estimation of a new project: This process uses the artifacts of other processes.The Project Manager and the EFES methodology expert group execute the process together.The expert group brings the artifacts of Data Analysis process: i.e final effort models created and final percentages of unexpected, overlooked, unplanned tasks etc. Besides, they perform a size measurement by using its process. At the end ,by using the effort models they create a base effort value. The base effort value is the development effort that can be derived from functional size of software. Other percentages of supporting &extra tasks are added upon this base value to estimate the required final effort.

.

1.4. Validation of the EFES methodology

We conducted a multiple-case study involving two case studies in order to evaluate the EFES methodology. The organization, where we carried out our multiple case study, is certified by ISO standards. The main business of the company includes design and manufacture of software intensive systems. It has a well-trained and experienced development staff and the usage of current software development methodologies and tools are encouraged. The number of the software developers of the overall company is over 300 and there exist several sub-departments for software. We performed our case study on one of these departments where more than 90 developers work. The software development is on several types of application domains. The projects use defined software processes of the organization. As a result of this, projects have defined outputs like software requirement specifications, design documents and software test specifications.

In the first case study, we performed studies for method development. For this reason, we investigated some concepts defined in the methodology such as: FS and BFC usage in effort model creation. We used a sample data set consisting of 18 projects. Most of these projects belong to Graphical User Interface projects (GUI) and very few belong to Embedded Systems (ES) and Board Support Package (BSP) Type. These projects' functional sizes are changing between 50 – 2040 Cosmic Functional Size. The expended effort range is between 29 day to 696 day. In parallel to this investigation, we searched the answers of quality and reliability issues of dataset in the Company and we developed data collection, size measurement, calibration requirements of the methodology and created assets, flowcharts, procedures.

In the second case study we validated our EFES methodology by using 22 new projects. The new projects are evenly distributed to all application domains. New projects' functional sizes are changing between 38 – 666 Cosmic Functional Size. The expended effort range is between 67 day to 402 day.

During validation we investigated answers for the following questions:

a. "Is methodology applicable?"
b. "Is it effective in building relation between size and effort"

1.5. Organization of the Thesis

The remainder of the thesis is structured into five chapters.

In Chapter 2, related researches on effort estimation modeling, functional size measurement methods and problems of effort estimation are surveyed. The summaries of the literature results for the identification or solutions of the effort estimation problems are given.

Chapter 3 forms the hearth of the thesis and describes the proposed methodology in detail. The model applied, the processes, their steps and the artifacts to be produced are described. Pre-required artifacts, roles and responsibilities are discussed. Also as an attachment to this chapter, templates applied for our methodology is given in APPENDIX 1 to 8.

Chapter 4presents the implementation of the approach in multiple case study. The chapter gives the details of our empirical executions, their results and discussions.

Chapter 5presents the conclusions and summarizes the contribution of this research. In this chapter, new questions that are raised by our research and the topics that inquire more investigations are also reported.

CHAPTER 2

2. RELATED RESEARCH

This chapter summarizes the literature related to Effort estimation modeling. The first section of the chapter describes the effort estimation modeling ways in detail. Section 2.2 describes the BFCs of existing size measurement methods. In section 2.3, literature researches are grouped under four subjects. Section 2.3.1 summarizes studies on estimation errors, 2.3.2 discusses the improvement opportunities of the relationship of size & effort estimation. Specifically, we focused on the studies about FS, Variable effects of BFCs, Application Domain effects on size and effort relationship. At Section 2.3.3 the need for database reliability, lessons learned by other companies are described. Finally, Section 2.3.4 summarizes the Data Analysis tools and commonly used accuracy parameters for estimation models.

2.1. Effort Estimation

To decide the feasibility of a project, accurate estimate is necessary. Because the customer, the project management and the top-management of the company must agree to the boundaries of the project in terms of cost, time, quality, and capability. For the companies "cost" is the main issue in general. A low cost estimate may cause compromise on the quality of the software delivered. In this case, the software is either in partially functional or insufficiently tested state. Therefore, to reach the final software that satisfies the requirements, high maintenance costs will be paid. Considering the importance, software cost is an activity that starts at the proposal stage and continues throughout the lifetime of a project.

The main components of project costs are hardware costs, travel and training costs, effort costs for the development. Among them the effort cost is the most difficult one to estimate and control. Although a number of effort estimation models are proposed, none of these models are accepted by the whole community since none of them performs well-enough in different environments [70]. A generalized model that we can use in different environments of various organizations does not exist. There are some reasons for this:

- The development environment evolves continuously, not stable.

- The complexity of the system is not easy to be reflected by measures

- There exist many interrelated factors affecting the project development process. The impact of these factors must be discovered and analyzed.

We have undertaken literature reviews to study effort estimation models. There are number of ways to determine the required effort in software development projects. In some studies there exists a precise classification of existing models, methods and techniques [54][86][87][88].We merged their approaches and summarized the models as in Figure 3.

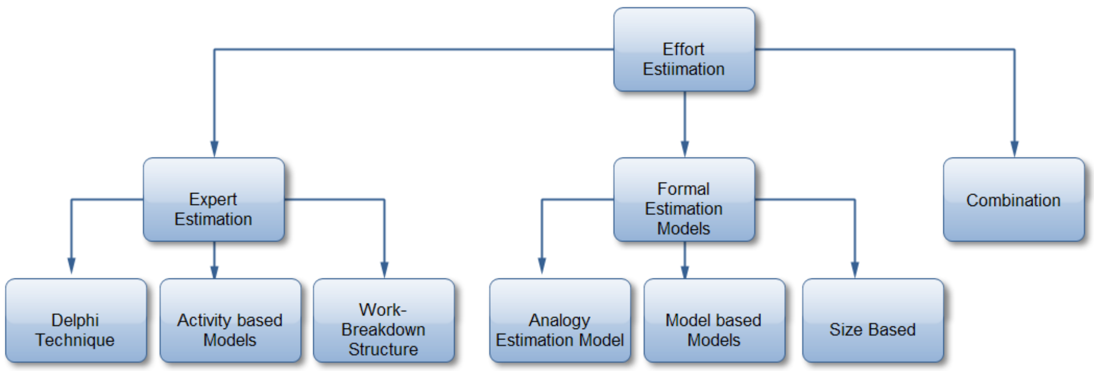


Figure 3 Classification of Effort Estimation Models

The first differentiation level is whether the method is a “Formal Estimation” method or “Expert Estimation”. Although a combination of these method is possible the essential difference between them is the final step that transforms the input into the effort estimate. Formal effort estimation models are based on a mechanical quantification step such as using a formula. On the other hand, expert estimation methods, such as work-breakdown structure methods, are based on a judgment-based quantification step. Expert estimation is not only a feeling but is based on some structured, historical data and checklists. In other words, the application of this method is not selecting the best expert.

2.1.1.Expert Estimation

These techniques involve consulting with software estimation experts or a group of the experts to use their previous experience to arrive an estimate for effort & cost. These experts usually use their understanding of a new project and available past information like design requirements, source code, software tools, development resources, complexities of the functions. According to the Barry Boehm using calibrated algorithmic models, i.e. formal methods, enables better negotiation rather than by a contest of wills between self-described experts [89]. However, as Jorgensen claimed in the same discussion and in the review study [90], several independent surveys rate it to be the preferred method [93].

The general way of applying this technique is the group estimates. The main concern in group method is the involvement of the estimators. Estimation groups may be designated to perform this task as DeMarco suggested [91]. In this case, the group members do not participate in the development process. So they are much less prone to personal or political biases. Besides they can improve their estimation skill in over time [91]. The other alternative is that employees in the project are responsible for the estimates. In this case they will probably get to know the project better than anyone else does, that results in a higher motivation for a thorough project estimation analysis [92]. A group estimate is one of this kind that involves the risk that people with stronger personalities may dominate.

Other variation of this estimation method is Work Breakdown Structure (WBS) [31] and freeform expert estimating [94]. Although, this approach has advantages like making a fast estimate, it has also drawbacks that may prevent being used by large projects; first, it is not repeatable, since means of driving are not implicit. For every new project, it may not be easy to find experienced people for that specific subject.

2.1.1.1. Delphi Technique

This is a group consensus method [4] [96]. It provides a sufficiently broad communication bandwidth for the experts to exchange the information necessary to calibrate their estimates with each other. The estimating steps are as follows:

1. An estimation form including specifications is prepared by the coordinator and presented.
2. A group meeting is held to discuss estimation issues with the coordinator and each other.
3. Experts fill out forms without knowing others' determinations.
4. A summary of the estimations will be distributed on an iteration form.
5. With another group meeting that focuses on the estimates varied widely is held.
6. Experts fill out forms, again, this process continues until they reach a consensus.

The experts can easily analyze the differences between past projects and the new project. Therefore, they can reflect impacts of new technologies, architectures, tools and languages involved in the future project. Besides, factors in exceptional personnel characteristics and interactions may also be included in decisions. However, it is not easy to quantify or document their approaches. Even a group consensus is aimed, final conclusions may be some biased, optimistic, and pessimistic, that results in wrong judgments [92][95]. If a new or unknown technology is used, some researchers like Kaczmarek et al. especially suggest Delphi method [97].

2.1.1.2. Work Breakdown Structure

Another approach in expert effort estimation is the usage of WBS and decomposing the project tasks. These methods organize and structure the phases, actions, tasks and other information. There are many ways to decompose a project into tasks. The project tasks can be organized by feature, by project phase (requirement tasks, design tasks, coding tasks, testing tasks), or by some combinations of them.

2.1.1.3. Activity Based Models

In this kind of model, estimators break down the task at hand into unit activities for which one can easily estimate the cost or effort required. Therefore, for experts this approach requires detailed knowledge of the product and process. It should include effort for all products whether internally developed or externally subcontracted. Besides documentation should also be included whether it is used in the company or delivered to the customer.

The low level estimates of each task can be either derived from expert judgment, or historical data. If cost is directly estimated then it is called as activity-based costing. If effort is estimated as in WBS method, it can be named as bottom-up estimating because they are derived from engineering estimates of size, effort for all products and activities in a project. Estimates are then aggregated to produce a project-level estimate.

Compared to other group expert estimation methods activity based models are applied by companies that has defined processes. Companies may create some specific templates for estimation. These models require that historical information from prior projects be accurately tracked.

2.1.2. Formal Estimation Model

We have grouped the formal estimation methods in three main categories: “Analogy”, “Size based”, “Parametric”.

2.1.2.1. Analogy Estimation Model

This method may be assumed as a systematic implementation of expert judgment. Because, experts often search for similar or analogous situations to check and validate their opinion. In this technique project is characterized to form similar or analogous projects that have been completed for which effort is known. So for some describers to find analogies are necessary to define the project. Then effort values that belongs to previous projects are used to generate a predicted value, by using some adjustments. For this second part the way of assessing the degree of similarity should also be defined. Selection of one or two past projects will be enough to apply this method, on the basis of their close similarity to the proposed project. Describers might be the type of application domain, the number of inputs, the number of distinct entities referenced, the number of screens etc. The choice of variables must be restricted to the estimation phase. Therefore, at the beginning of the project LOC is generally unsatisfactory for the application of this method.

Prerequisites of Analogy estimation process were listed by Chemuturi [98]. A summary of this process is as follows:

- The organization ought to have executed a number of projects: In order to find a similar project, a large database is necessary. One or two similar projects will be enough to apply this method.
- The organization should be keeping characterizing records of the projects.
- The organization should be keeping causes for variances and the actual values validated depending on the causes.
- The organization should maintain a repository so that it is feasible to locate similar past projects and extract the related information.
- The estimators should be trained in drawing analogies accurately to extrapolate the information for the new project.

The most crucial aspect of this estimation method is the selection of the right set of past projects. For the choices organization may use either judgment of estimators or an analysis method. As an analysis method "k-nearest neighbor" method is generally used [99][100]. The aim is to classify similar projects depending on the similarity of the new projects' characterizing properties to the properties of "k sample projects" in the dataset. Effort of the similar projects are used as an initial estimate, then comparing the known measurement values for the new and executed projects, effort estimate is adjusted to compensate the differences.

An automated environment named as ANGEL was proposed by Sheppard and Kitchenham to support the collection, to store and to identify the most analogous projects. They also recommended choosing at least one variable as a size driver, for instance number of inputs or screens or classes [101]. With this tool they provided templates for recording data. But prescription of the describers is performed by the organization to suit the individual data collection environment. Tool allows searching for one, two, or three analogous projects and calculates an un-weighted mean of their effort values to estimate effort for the new project. Estimation by analogy is a popular technique which is heavily researched [47][101][102]. However, in this method it is not possible to create a definitive model about the relationship between the project data and the effort.

2.1.2.2. Model Based

Model-based estimation makes use of a mathematical model to produce effort estimates. It is also known as parametric estimation since there are a number of variable parameters within the model that must be determined. These models estimate effort or cost based on mainly software size, and other productivity factors known as effort driver attributes. An algorithm or analysis method is used to form a final model based on the results obtained from previous projects. Most of the well-known existing models belong to this section. These models mainly are based on a measurement metric like Lines of Code, Function Points etc. Some of the most widely known estimation techniques are explained in following paragraphs.

2.1.2.2.1. Constructive Cost Model (COCOMO)

It is an algorithmic cost model that uses a basic regression formula, with parameters that are derived from historical database and current project characteristics [25]. Basic COCOMO computes the effort as a function of program size. Program size is expressed in estimated thousands of lines of code. It applies to three classes of software projects by using the Formula : Effort= $a\text{Size}^b$ where a, b changes according to the types of projects: Organic projects, Semi-detached projects, Embedded projects - developed within a set of "tight" constraints (hardware, software, operational etc.)

For quick estimate of software costs and effort this easy method will be applicable. However it does not take into account the differences in hardware constraints, personnel quality and experience, used tools and techniques etc. Therefore an extended version named as “Intermediate COCOMO” is proposed, that considers a set of four "cost drivers", each with a number of subsidiary attributes [167]. Intermediate Cocomo formula now changed into the form where EAF is the adjustment factor for the effort estimation.

$$E=a\text{Size}^{b}.\text{EAF}$$

For EAF calculation, 15 attributes of the project are included as driving factor. These factors are given in Table 2. The table is from study Boehm and Horowitz [167] Each of these attributes receives a rating on a six-point scale that ranges from "very low" to "extra high". Finally all of these ratings are multiplied with each other to obtain EAF. The COCOMO model has still been under improvement. Even more, some other new models are combined into COCOMO suit like COSYSMO (Constructive Systems Engineering Cost Model), COSECMO (COnstructive SEcurity Cost MOdel), COCOTS (COnstructive Commercial Of The Shelf Cost Model) [103][54][105][107].

2.1.2.2.2. SEERSEM

It is a commercial parametric estimation model developed by Galorath Inc. [106]. Since it is a commercial tool, the mathematical equations used in SEER are not available to the public, but Jensen made the basic equations available in his paper [108]. It takes into account more than 50 parameters to be arranged according to the project and company specifics. A large number of controls increase the complexity of SEERSEM and also the uncertainty of these outputs.

Therefore the increased number of controls results in difficulty in building relationship between input and output parameters. It allows project elements to be included as WBSs for convenient planning and control. Risk analysis can also be applied for each of the project element.

2.1.2.3. Size Based Models

There is a large body of literature on effort estimation models in which discussions on the relationship between software size and effort as a primary driver has been included [3][25][83][109]. This relationship is affected by many factors [110]. To create a model that represents this relationship these factors must be investigated and their effect must be analyzed. For the representation of this relation definition and granularity of software size is important.

Table 2 COCOMO II software cost drivers (adapted from Boehm and Horowitz)

Product - Required Software Reliability - Data Base Size - Product Complexity - Developed for Reusability - Documentation Match to Lifecycle Needs	Personnel - Analyst Capability - Programmer Capability - Personnel Continuity - Application Experience - Platform Experience - Language and Toolset Experience
Platform - Time Constraint - Storage Constraint - Platform Volatility	Project - Use of Software Tools -Multisite Development -Required Development Schedule

2.1.2.3.1. Lines of Code (LOC)

It was the first commonly used and accepted measure, being used as the basis for measuring programming productivity and effort [112]. However the definition is sometimes controversial; while some definitions include comment lines, some others don't, some include logical lines, while others consist only of physical lines. In the 60s a couple of thousands of LOC were enough to develop complex software, however after 2000s a couple of millions of LOC is necessary with the increasing and entangled requirements. Compared to other engineering disciplines software projects' complexity and intricacy increased sharply during last decades. At the end of 70s, LOC was assumed as a sign of productivity and many models based on Lines of Code have been calibrated for different sets of historical data. Waltson Felix [113], Moher-Schneider [114], COCOMO [25] are some of them.

In those years using LOC as a base for effort calculation was meaningful, since the programs were small and coding work comprised about 90% of the total effort. As higher level programming languages appear, the usage of this measure slightly disappeared. Programs written in different languages couldn't be directly compared. Although each language has assigned a level, or comparison tables for different languages have been formed, LOC approach lost its dominance due to its drawbacks. The main disadvantage of LOC is; it cannot be known until the project is finished. Size measurement is needed at very early stages, however reckoning the size of a project in terms of LOC at the beginning of a project is almost impossible. Historically based effort estimation methods appeared at this point, but evolving rate of software tools created new problems like effects of development tools, environments, language types and automated generation of codes. Among the historical models that use LOC, three model based estimations gained popularity: COCOMO, PRICE-S, SLIM [115][116]. After years, PRICE_S and SLIM reached the commercial market as measurement tools.

2.1.2.3.2. Function Points

The aim of this new sizing method is to measure the functionality of the software that is independent of its implementation. Albrecht claimed that Function Points has some advantages compared to LOC [32]: First with this new method, earlier measurement is possible at the phase of software requirements analysis and preliminary design. The second advantage is measurement can be performed by non-technical project members. Besides it is independent from implementation language and developer experience.

In this method, each system is considered in terms of the number of inputs, outputs, inquiries, files and external system interfaces that it contains. Based on the number of data elements or file types referenced a complexity weighting factor is assigned and multiplied by each of these components of the method. This method has become a *de facto* standard in the MIS community. Currently, the International Function Point User Group is maintaining the official guidelines for counting function points [69]. Starting with the Albrecht’s FPA[32], many FSM methods have been proposed as it was summarized in Figure 4.

Albrecht function model is refined in 1983 by Albrecht and Gaffney [3]. Their model is accepted as the first function point based effort estimation model. They found a linear correlation between function point size and effort in hours. This new version of function point included three levels of function complexity. Some rules were outlined for evaluating complexity by function type and accompanying weights were tabulated. Two subtypes; “the internal logical file” and “the external interface file” was specified as “file type”. The function types in this version were identified as External Input, External Output, External Inquiry, Internal Logical File and External Interface File. There is a popular software effort estimation package, “Checkpoint”, based on their results.

In 1984, the International Function Point Users Group (IFPUG) was established to maintain Albrecht's FPA. Since then manuals that clarify and modify standard rules for the application of Function Point Analysis have been published. In 1984 Symons devised his concerns and difficulties with Albrecht's FPA, and proposed a new variant called Mark II Function Point Analysis [38]. He found Albrecht’s method classification very simple, the origin validity and objectivity of the weights very doubtful and the treatment of internal complexity as very complicated. Today, the United Kingdom Metrics Association maintains Mark II FPA. It was announced as the UK Government’s preferred technique for measuring the functional size of software applications and projects.

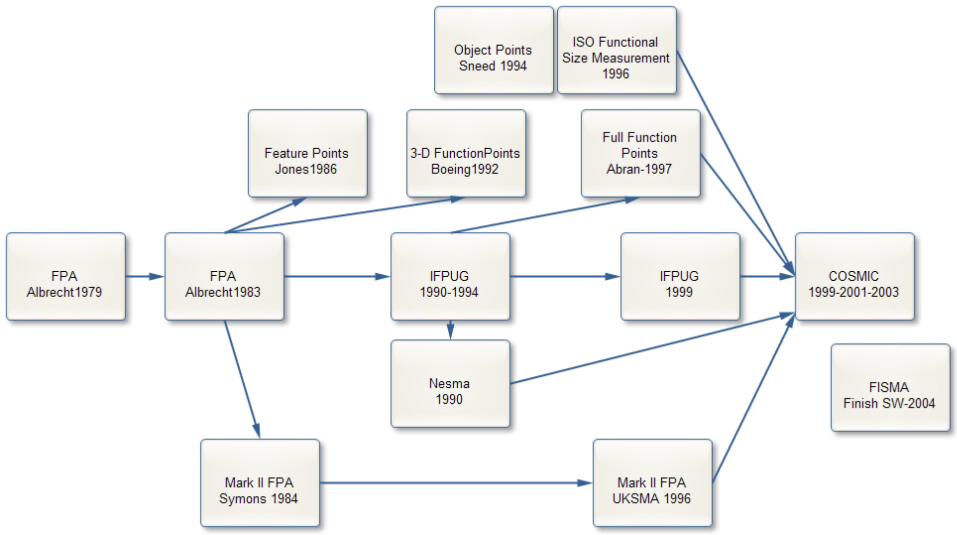


Figure 4 History of Functional Size

In 1986, Feature points were developed by SPR Practitioners of function point recognized that some type of applications did not show any beneficial results [40]. These applications are real-time process control, mathematical algorithms and various embedded systems. To compensate the difference in effort, some of the weights of function point components were modified. Typically these applications have higher sizes when measured with feature points than with function points. On the other hand, when function points have been applied to typical business applications, the same size value is achieved whether calculated with function points or feature points. Capers Jones has documented this method in detail in his book [118]. The difference compensates for the fact that productivity for these applications usually appears to be lower.

In 1987, Kemerer [65] used projects from business applications domain, for the comparison of SLIM, COCOMO, ESTIMACS, and Function Points (FP). He accounted that methods that are not based on KLOC as a size measure (FP and ESTIMACS) performed better than KLOC-based methods (SLIM, COCOMO). Besides he claimed that Albrecht and Gaffney's model [3] is not a linear model and is not appropriate for effort estimation and proposed a nonlinear new model.

In 1990, NESMA FPA was proposed as a variant of IFPUG FPA with the aim of simplifying some of the IFPUG FPA sizing rules [117]. FPA usage was particularly for measuring productivity. Practitioners aimed NESMA approach to be used for budgeting purposes. Therefore they adapted some of FPA's counting guidelines so they could be applied to logical models. This inevitably led to a number of differences in how the NESMA FPA and the IFPUG FPA counted FP.

In 1992, Whitmire in Boeing Company proposed 3D function points to address the problems associated with measuring complex scientific and real-time systems [120]. However it was found less valuable for effort prediction. So the method had not gained much popularity or widespread utilization.

In 1994, Matson, Barret and Mellichamp [66] advised another method that was an alteration of Albrecht's FPA. This research was a follow up study, that compared two models above, and his new model. According to their results new proposed model showed a superior performance compared with Albrecht &Gaffney's model. Their model had again a non-linear relationship between the development efforts and the FPs. They found that both of the previous models had some limitations. In this method, raw function counts were calculated by considering a linear combination of five basic software components.

In 1997, Abran [121] suggested the Full Function Points (FFP) technique as an extension to the IFPUG standard in order to capture the functional size of real-time applications. It provided additional dimensions to the five original FP dimensions. With the addition of six new dimensions the complexity of communicating processes and their synchronization were also taken into account.

Table 3 Applicable Domains for Functional Size measurements (from study Ebert et al.)

Method	**Algorithm**	**MIS**	**Real Time**	**Control Systems**
Feature Points	***X***			
3D Points	***X***		***X***	
IFPUG		***X***		
Mark II FPA		***X***		
FFP			***X***	***X***
COSMIC		***X***	***X***	***X***

In 1999, Version 2.0 of the FFP approach was published by the Common Software Measurement International Consortium (COSMIC) [36]. The COSMIC group reviewed existing methods (IFPUG, MarkII [34], NESMA [117] and version 1.0 of the FFP methods. After studying the commonalities, this method was proposed as a new generation of software FSM method.

In 2004, FiSMA method was developed by a working group of Finnish Software Measurement Association (FiSMA) [122]. It was designed to be applied to all types of software. The difference from other methods was that, FiSMA FSM was service-oriented instead of process-oriented. In process oriented methods, all functional processes supported by the software need to be identified. In this method, similarly, all different services provided by the software need to be identified.
For non-MIS environments, such as real-time, Web, Object Oriented, and data warehouse systems, other types of measures were suggested to measure the functionality.

Although a number of measurement methods have been suggested, mainly four of them have been recognized as standards: IFPUG FPA, MK II FPA, NESMA FPA, and COSMIC FFP.
In Table 3 we summarized the applicable domains for these methods. This table is presented in study [119] of Ebert, Dumke, Bundschuh, and Schmietendorf.

2.1.3. Composite

Models are built based on combining above methods. Some examples are as follows: modeling based on expert knowledge elicitation [123], and modeling based on expert opinion and project data [18].

2.2. Base Functional Components of Size Measurements

Although a variety of methods are proposed, all function point measurements depend upon the same principles. For functional size measurements, it is necessary to define the boundary of the system. Boundary is a kind of logical line that separates users from a system. By looking at the requirements for the software, actions and data that are meaningful to the user are identified and relegated according to some kind of complexity criteria. Most methods consist of component classes such as inputs, outputs and file references, referred to as BFC types. These components are then assigned values and they put up their respective value to the total functional size. In following paragraphs BFC approach of some popular size measurement methods are explained.

2.2.1. IFPUG -FPA

The method brought in a specific way of measuring the size of a software system. It distinguished data functions and transactional functions. Data functions (DF) are separated as internal and external logical files (ILF and EIF). Different weights are assigned to these each data function type. For the transactional functions (TF); functionality of the system is represented with three abstract components, external inputs (EI), external outputs (EO) and external inquiries (EQ) as it is shown in Figure 5. Each of these measurement components contribute to the final estimated size by their respective complexity value. The complexity value is determined by the number of simple data elements named Data Element Type (DET) or structured elements named Record Element Types (RET). Definitions of these components are given in Table 4.

The measurement process can be summarized in the following steps: First, the functional components existing in each prescribed function is identified. Then, for each function a ranking level of simple, medium or complex level is assigned by considering the number of components found and using the complexity ranking table. In the next step, the results are summed up to generate an unadjusted initial function point count. Then, the total degree of influence on the general system characteristics is determined. Finally, the final function point is calculated by using the unadjusted function point count and the influence adjustment factor.

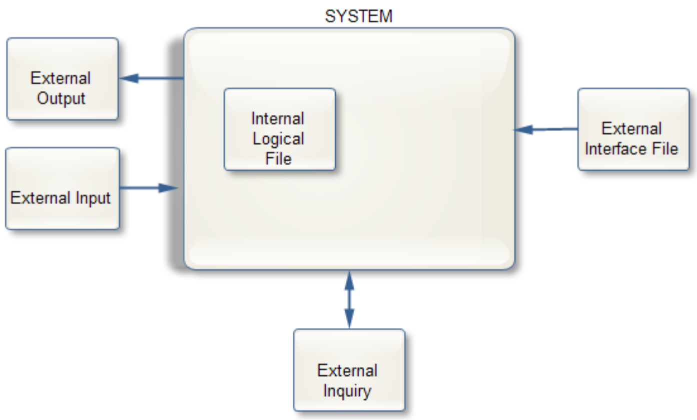


Figure 5 IFPUG Measurement

2.2.2.Mark II Function Point

For the MKII size measurement [34], the application size is counted as a collection of logical transactions. Each transaction consists of an input, a process and an output component as shown in Figure 6. With the aid of the boundary definition, logical transactions such as inputs and exits that cross boundary are determined during the interaction between the user and the system. Examples of input transactions create new records, update or delete them. Exit type of transactions are processes that take information from the system and show them to user such as reports, notifications or searches.

The size of the application is the sum of the sizes of logical transactions - each transaction is counted once even though it may be executed from more than one point in the application.

Two critical definitions for this method exist; “DET” and “Data Entity type”. Data Entity type is a fundamental thing of relevance to the user, about which information is kept [159]. Data Element Type is a unique user recognisable, non-recursive item of information.

The number of data element types is used to determine the input and output size of each Logical Transaction. Similarly, the size of the processing component is the count of data entity types.

For the final size count a weighted computing method is used. Size can be expressed as following:

$$\text{Size} = Wi*\sum Ni + We*\sum Ne + Wo*\sum No$$

In the equation above $\sum Ni$, $\sum Ne$ and $\sum No$ are total numbers of inputs, exits and process within the system. Weighting coefficients assumed for these size components are as $Wi = 0.58$, $We = 1.66$, and $Wo = 0.26$. They are accepted as average value obtained by the analysis of business information systems.

MK II is primarily used for domain of business information systems. For the other type of domains, components with complex algorithms should be evaluated differently, since sizing rules do not take into account their contribution. It has only one logical component, however it has constituents as given in Table 4.

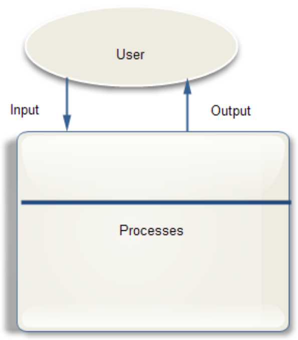


Figure 6 Mark II Functional Size Measurement

2.2.3.3D Function Points

The method recommends "three dimensions" in sizing: data, function and control. The data dimension is similar to Albrecht's function points. The function dimension measures the complexity of algorithms. The control dimension adds transitions, which enumerate changes in application state. Characteristics of all 3 dimensions are counted, quantified and transformed into a measure that provides an indication of the functionality.

Table 4 BFC components of Functional Size Measurement

Method	**BFC Components**
IFPUG	External Inputs: *unique user data or control input that adds or changes data*
	External Outputs: *unique user data or control output that goes out the boundaries of the system,*
	External Inquiries: *unique input that creates immediate output*
	Internal Logical Files: *internally maintained logical group of data,*
	External Interface Files: *file passed or shared between applications*
FFP	*Data Function Types:*
	Update Control Group: *group of control data updated by the application*
	Read-only control Group: *group of control data used, but not updated, by the application being counted*
	Transactional Function Types
	Ext. Ctr.Entry : *control data that are coming from outside of the boundary.*
	External Control Exit: *control data goes outside the application's boundary*
	Internal Control Read: reads control data
	Internal Control Write: *writes control data*
Mark II Points	Input : *event, user query, timed trigger*
	Processing *: references to retained data*
	Output: *report, display or response*
3-D Points	- Data Components
	Internal Data: *Internal files,or references*
	External Data: *External references*
	Inputs: *Same with IFPUG*
	Outputs *Same with IFPUG*
	Inquiries *Same with IFPUG*
	Functional transformations*: the number of internal operations required to transform input to output data*
	Control transitions: *counting the number of transitions between states*
Nesma	External Inputs: *data into the application without performing data manipulation*
	External Outputs: *moves data towards the user, and performs some data manipulation*
	External Inquiries: *moves data towards the user, and does not perform data manipulation*
	Internal Logical Files*: persistent data maintained by the application through the use of EIs*
	External Interface Files: *persistent data used by the ap-plication, but not maintained by it*
Cosmic	Entry: *from user to system*
	Exit:*from system to user*
	Write: *from system to persistent data*
	Read: *from persistent data to system*
Fisma	Interactive end-user navigation and query services
	Interactive end-user input services
	Non-interactive end-user output services
	Interface services to other application
	Interface services from other applications
	Data storage services
	Algorithmic and manipulation services

2.2.4. NESMA

In NESMA-FPA [117], the BFCs are transactions. Each transaction has a certain type and functionality. Both NESMA and IFPUG differentiate the same five types of user functions. But there are a few exceptions for determining the type and complexity of a function. For example: for IFPUG, an External Inquiry is specified as a function that delivers data to a user from a logical file without experiencing additional processing, such as calculations, updates to an Internal Logical File, etc, otherwise it is regarded as an External Output. However, for NESMA, additionally, a unique selection key must have been entered and the output must be fixed in scope. Therefore, while IFPUG will consider a transaction as an External Inquiry, NESMA assume the same function as an External Output.

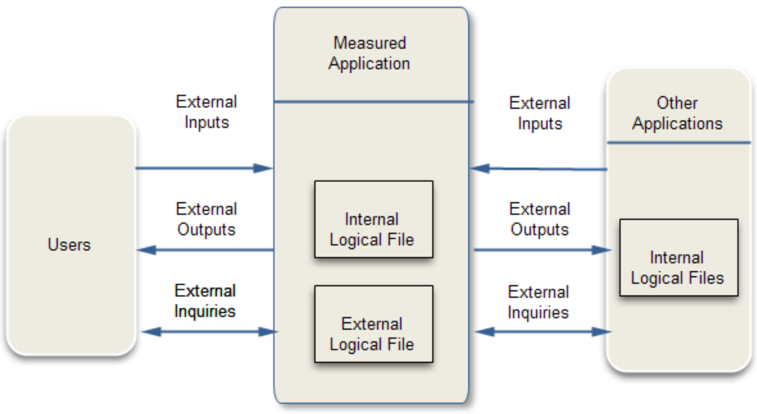


Figure 7 Nesma Functional Size Measurement

2.2.5. Full Function Point

FFP uses the IFPUG FPA rules for business application software. However, it adds six additional data and function types for sizing real-time software applications. There are four new “Control Transactional Function Types” that address the sub-process of real-time software. Besides, two new Control Data Function Types are included. Definition bases on the information whether single or multiple occurrences of data are concerned. Functional components are shown in Figure 8.

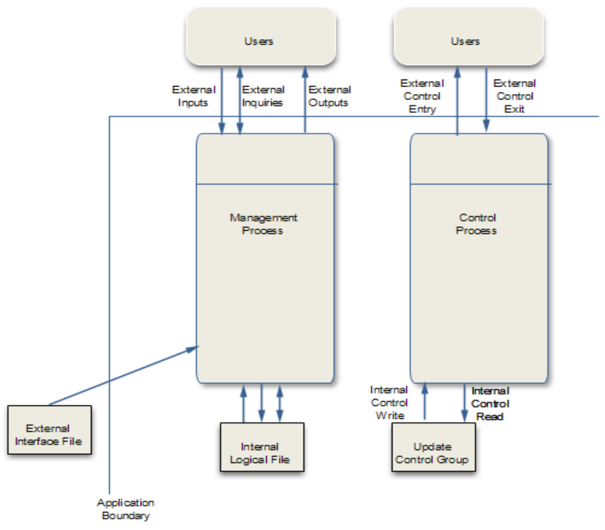


Figure 8 Full Function Point Measurement Method

2.2.6.FISMA

FiSMA FSM is service-oriented instead of process-oriented. So, as BFCs of measurement, services are defined. It defines seven BFC classes. Each BFC class of FiSMA 1.1 further decomposes into several BFC types. After identifying each service, the size of each service is found by applying rules of method. Finally, a total functional size is estimated by adding up the sizes of all services. Totally, there exist 28 BFC types as shown in Figure 9.

2.2.7.COSMIC

The COSMIC Measurement Process consists of three main phases[]:

- Setting the Measurement Strategy
- Mapping the ‘Functional User Requirements’ (or ‘FUR’) of the software to be measured to the COSMIC concepts
- Measuring the resulting COSMIC model of the FUR

In the first step, the purpose and the scope of the measurement is determined. The software artifacts; statements of requirements, physical screens etc. are defined. The functional users and the boundary of each piece of software is determined. Examples of functional users are human beings, devices or other software. Besides, the required level of granularity for the measurements are specified. For example, if the strategy is to estimate a distributed software system and if the various functionalities are aimed to be implemented on different technical platforms, then the FUR must be separated into a number of separate measurements.

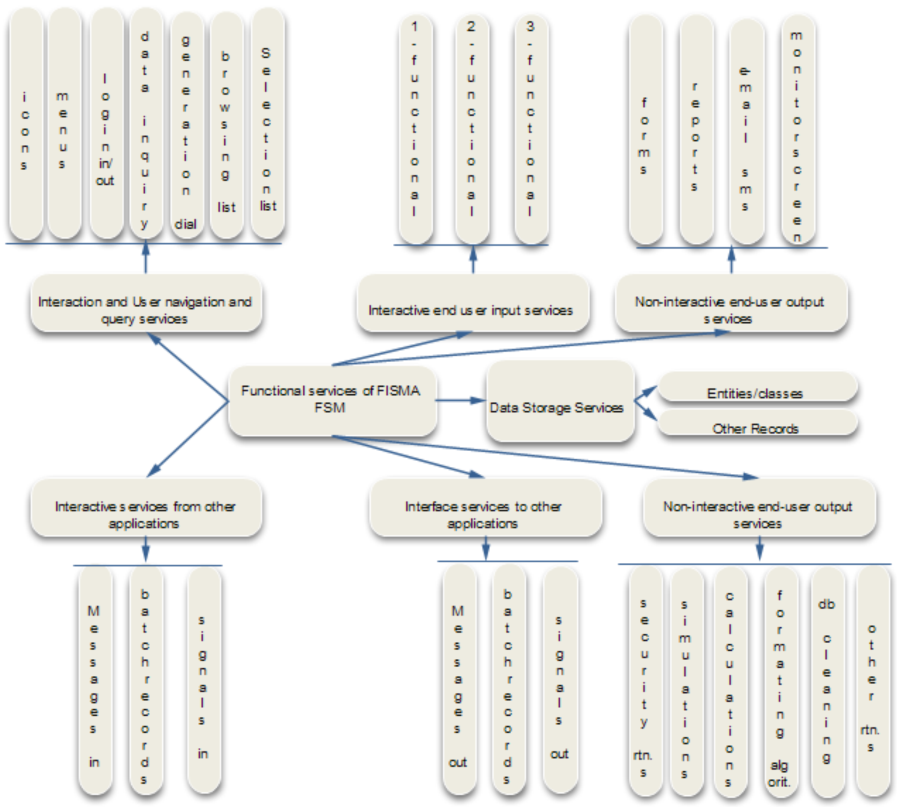


Figure 9 FISMA Functional Size Measurement

During the second phase software layers are established. Software layers operate on a specific level of abstraction based on the functional exchanges among the modules of software. At that point, events of the functional users that the software must respond and functional processes are identified. And as a next step, objects of interest and their attributes are determined. Objects of interest are the key players interacting within a functional process. Each of those processes covers a unique set of data movements or data manipulations as shown in Figure 10. A 'data movement' is the movement of a single 'data group'. Data groups includes one or more 'data attributes' about a single 'object of interest'. The Cosmic model identifies four types of data movements:

- Entry; data movement occurs from the functional user across the process boundary to inside the functional process.
- Exit; data movement occurs from inside the functional process across the process boundary to the functional user.
- Read; this movement provides data inside the process from a persistent data store (for example, a database).
- Write; this movement provides data from inside the process to a persistent data store.

Any functional process comprises ‘sub-processes’ that either move or manipulate data as it is shown in Figure 11. In the last step, the number of data movements within each process are counted. Each data movement equals a 1 Cosmic Functional Size Unit (Cfsu).

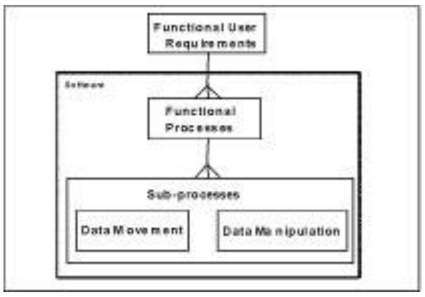


Figure 10 COSMIC Generic Model [104]

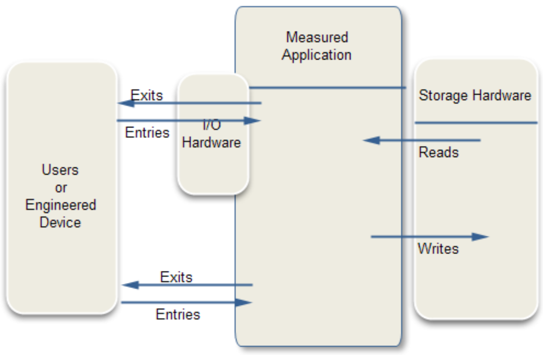


Figure 11 BFCs of COSMIC

2.3. Effort Estimation Studies

We grouped “Effort Estimation improvement” related studies in four groups:

a. Studies, investigating “Reasons for Effort Estimation Errors”

b. Studies, searching for “Improvements and problems on Size & Effort Relationship”

c. Researches, examining the “Need for Database Reliability and Effect of Datasets on Accuracy”

d. Comparison studies on “Data Analysis methods”

2.3.1. Reasons for Effort Estimation Errors

In literature many studies warn us about the unplanned activities, tasks, unexpected problems that are included in effort value but not a part of the development process. We represent them in "Supporting & Extra Tasks" in Figure 1. They are different from commonly known adjustment factors of Cocomo[25].

Jorgensen and Molokken [23] performed a study to understand why errors occur in software effort estimation. The related information is collected through interviews with estimation responsible employees in different roles and estimation experience reports from 68 completed projects. They made a statistical analysis of relations between characteristics of these projects and estimation errors. They also investigated literature on this subject. A summary of their results and other related researches is depicted in Table 5. They acquired from the experience reports that inclusion of a large buffer to deal with such unexpected events and/or changes in specification will improve the accurate estimates.

A similar study was performed by Genuchten [19]. He investigated the reasons for differences between plans and reality and recommended that every department should find its own reasons for this effort estimation error. He classified his findings under five subjects that each of them has specific situations in it. "Capacity related issues" include; overruns in previous activities, unplanned maintenance activities, unplanned demonstrations, and other unplanned activities. "Input related issues" include; late delivery of HW, insufficient hardware, late requirement delivery from customer. "Product related issues" include; changing requirements, changing interfaces, complexity of application, performance requirements. "Organization related issues" include; Project staffing, interruptions, etc. "Personnel related issues" include; "inexperienced team", "too little experience in development environment". Morgenshtern, Raz and Dvir [6] claimed that planning at a more detailed level, i.e. defining shorter activities and smaller tasks, increases the accuracy of the estimates and reduces the size of the estimation errors. Besides, using the estimates for project monitoring and control increases the commitment of the estimators (project team) and reduces the occurrence of estimation errors. Another claim in their study is related to estimator experience. Their results show that experience contributes to reducing estimation errors. Moreover, the number of projects that the estimator has managed is found highly correlated with estimation accuracy compared to the number of years of experience in managing projects.

McConnell [26] proposes that 20-30% of the errors come from not estimating certain activities. He categorizes omitted activities into three categories that is given in Table 34 in Appendix. Besides, in his book, he recommends to use such checklist as a part of effort plan.

We underlined some subjects on Table 5. These subjects are the ones that was mostly mentioned as the reasons in estimation errors in mentioned studies: “requirement or design changes”, “missing activities”, “unexpected activities”, “unavailability of developer”, “unavailability of hw”, “more detailed task planning”, “reuse” etc.

2.3.2. Improvements and Problems on Size and Effort Relationship

2.3.2.1. Functional Similarity

Table 5 Results of Effort Estimation Error Studies

Jorgensen Molokken “Experience Report-based Reasonsfor Inaccurate Estimates”[23]	• Unexpected events and overlooked tasks (›) • Change requests from clients or “functionality creep” • Simpler task or more skilled developer than expected • Resource allocation problem • Poor requirement specification or problems with communication with the client • More reuse than expected from other projects • Too little effort on estimation work • High priority on quality, cost accuracy not of high importance
Jorgensen Molokken Project Related Characteristics[23]	• Skilled project managers • Stronger involvement in the project work leads to more accurate estimates • Time is necessary for the development of proper requirement specifications
Phan et Al.[72]	• Unrealistic over-optimism • Frequent changes
Van Genuchten[19]	• Frequent changes • More time spent on other work than planned • Complexity of application underestimated • Capacity related:overruns of other work • Late delivery of Hw Components • Maintenance activities interruption
Lederer and Prasad[20]	• Frequent requests for changes by users • Users lack of understanding of their own requirements • Overlooked tasks.
Standish Group[57]	• Lack of user input • Incomplete/changing requirements and specifications,
Subramanian and Brewslawski[21]	1. Requirement change/addition/deletion, 2. Design changes, scope, complexity 3. Programmer or team member experience, 4. Turnover
McConnel[26]	5. Missing requirements 6. Missing software development activities and 7. Missing non-software-development activities.
Raz Dvir[6]	8. Need for commitment of Estimators 9. Need for More detailed task planning 10. Need more experience on Estimation

In FP calculations every software module can be represented by several functional processes. Regardless of the aimed logical functionality, if functional processes of modules are similar to any other process, effort needed to develop new ones will decrease, and total effort will not be proportional to the functional size.

Reusability may occur in several ways and several terminologies exist for the specific usages. “External reuse”, “internal reuse”, “commercial of the shelf (COTS) reuse”, and “replication” are some of them. Reuse is named “internal”, when a module or process created for a system is used multiple times within the same system and “external” when a module from a different system is used one or more times within a new system [123][125]. In organizations, the external reuse is commonly considered to be the inclusion of the functional similarities and the reusability percentage of the product is determined by the ratio of reused modules to the overall project.

As external reuse, internal reuse also has a significant impact on the total required development effort and time. Although the most of the software products have functionally similar modules or similar functional processes, it is not always easy to determine functionally similar software entities at the beginning of the projects. Moreover it is not clear whether the functionally similar entities require exactly the same components for the development or not, and what the impact of the similarity on the development effort should be.

In the literature, there are a few approaches to determine functional similarities. These approaches can be grouped into two categories: One of them is comparing the components of functional processes. The other approach consists of the entity generalization/specialization practices which are widely used in object oriented methodologies and can be used in the grouping of the similar functional processes into one and as a result can be used to eliminate the replication of the same/similar functions [126]. The second approach can be applied at the beginning of the projects only by using expert judgment and inheritance relationships in the design artifacts can be utilized as a base for the determination of the functional similarities in the subsequent phases.

In terms of functionality, similarities on software applications have been subject to research projects and defined by using different terminologies. Fenton [125] defined a concept called “private reuse” as the extent to which modules within a product are reused within the same product. Cruickshank and Gaffney[124]also made first distinction for “internal” and “external” reuse in the literature from economical perspective.

Functional similarity (FS) concept was first defined by Meli as the re-utilization existent logical data structures and functionalities to build up new logical features [81]. But how the similarity should be considered for effort and size calculations was not defined. Santillo and Noce [80] defined the same concept, and suggested a model for calculating the size considering the similarities as “Worked function point model”. This Model was used to achieve a more significant “work size” to be correlated with effort. Model included “reuse”, “replication” and “similarity” adjustments and used predefined reuse adjustment coefficients for every reused-function. However the method was not validated by means of industrial experiments. A similar set of coefficients was also proposed by NESMA for enhancement projects [117]. A similar study was performed by Top [128], that assigned specific adjustments for reusable components.

Whatever the terminology is, the FS concept has a significant effect on all phases of the life-cycle of the projects. For example, the effect of functional reuse in maintenance was evaluated by Abran and Desharnais in 1995 [129]. They developed an approach for the identification and measurement of reuse in the enhancement projects by considering Function Point Analysis Method. Their approach depended on two key concepts: reuse indicator and predictor ratio. The study depicted how an alternative size measure could be obtained by combining predictive ratio and reuse indicator.

FS has been subject to one of the common FSM methods, COSMIC. It defines the FS concept in its Guideline for Sizing Business Applications document [130]. It is stated that developers might avoid duplications by realizing the functional reuse opportunities among functional processes, however, the user point of view ignores the functional similarities since the FURs are measured independently instead of grouping similar functional processes.

Abran and Maya [131] are one of the researchers evaluating similarities within a software product from a FS perspective. They refined and extended the functional similarity measures to create a more precise basis for the cost estimation and productivity models.

In addition to above, there are considerable numbers of research studies evaluating the software reuse performing at the source code level. However, few of these studies focus on developing methods to identify the functional similarity in the early phases of the software life cycle [3] [132] Ho, Abran and Olingy [133]emphasize the importance of measuring the functional reuse in the early phases of the software life cycle rather than coding phase to improve the performance of the software engineering processes. Their work bases on extending the method of Abran et al [129] by using the COSMIC FFP method. The approach proposed in their paper considers only the reuses without modification and is called black box approach. The approach utilizes the functional relationships among functional layers.

The development effort and the functional size correlation have been subject to research studies as well. In 2000, Meli [81]discussed the effort and size correlation problem. He stated that in some situations, it is possible to aggregate and assemble much different logical functionality which leads to rapid and economic implementations with a small amount of working effort. As a result of this, the effort needed to realize the overall system will decrease, and will not be proportional at all to the logical functionalities required.

In their study Santillo and Abran [79] proposed the approach called "functional similarity" to identify the software reuse from a functional perspective. The technique is based on uncovering the functional similarities among functional processes from a data set that comprises functional processes, data movements and data manipulations which are evaluated by using the COSMIC FSM method. Although their study comprises a method sorting out functional similarities, it does not provide an approach for the relation of size and effort.

Lastly, entity abstraction methods are also valid approaches for eliminating the measurement variances, grouping similar functional processes, providing abstract data and as a result eliminating the replication of the same functions. A research study considering this approach was conducted by Turetken *et.al [126]*. They depicted the utilization of entity generalization concept in COSMIC and IFPUG FPA Methods and evaluated the effect of different interpretations on the measurement results.

2.3.2.2. Variable Effects of Components

Component effects were considered in some size measurement models like MKII and IFPUG and were included by different multipliers to functional size components. In others such as Cosmic, components were not treated separately. In Mark II and IFPUG the purpose of the multipliers as well as their origin were undefined. Multiplier values for MKII and IFPUG were fixed and not arranged according to the environment, conditions or selections of the project.

We realized that, using subcomponents of size to improve the accuracy of the effort estimation was a concern in 2000s. Kitchenham and Kansala [82] suggested that Function Points were not well-formed measures because there was a correlation between their constituent elements. They claimed that since function point elements were not independent, there might be better measures for effort prediction than the sum of the elements. Using stepwise multivariate regression to derive the best fitting equation between effort and the five function point elements, they concluded that, for their dataset, the best fitting equation included only two elements: input and output function points. So according to this study, all the function point elements were not related to effort.

Abran et al. [134] investigated whether there was a relationship between the individual function types or profiles with project effort. In this study the concept of a software functional profile is defined as the distribution of function types within the software. They have used FPA's 5 function types. In their results of comparison language was the main consideration. Depending on the language, functional profile was found different. They found that External Input and Output function types made important contribution to total effort while others had a weak relationship with effort.

Abran and Panteliuc [135] investigated the sub-components' effect on application types, based on regression models. They used the ISBSG data set projects measured using Cosmic FFP and built linear regression models with and without considering the functional subcomponents. The types of the projects were divided into three groups; enhancement project; development type with single layer and development type with multilayer. In that study, a term, "functional profile" was used to define subcomponent's effect on effort value. They concluded that identification of the functional profile of a project and its comparison with the profiles of their own group of development type can help in selecting the best estimation models.

Gencel and Buglioni [83][84], compared the effort estimation based on BFC types, with the ones based on a single total functional size value. They performed multiple regression analysis for investigating the strength of the relationship between the functional sizes of BFC Types and development effort. In both studies they found significant improvement in size-effort relationship. Tunalilar and Demiors applied both multi-variate regression analysis and ANN method for creation of effort model using BFC types of COSMIC [160]

2.3.2.3. Application Domain Specific Effects

FSM methods are developed mainly for MIS type of software. Therefore for other application domains, examples about the use of rules in standards and problems encountered during that experiment is important. Some companies provide their results to software community for information sharing purposes [27][28][76]. For example, even Cosmic method is accepted as applicable to real-time & embedded type of software, empirical research results of companies using this method will be beneficial for other measurers in the world.

Companies need some guidelines in order to validate size measurement values and to achieve common consensus. It is well known fact that even certified measurers may encounter conflicting situations. In these cases, a commonly accepted procedure among measurers is very useful that identifies the rules for size measurement [29]. Although standards explicate the concepts, assumptions for measurement should have been written for future use. This is recommended by other companies to satisfy reliability. For example The Company, Renault [27][28], who managed its automotive embedded software development cost by using Cosmic method wrote rules for measurement. As its software development costs were increasing, Renault decided that some measures and an estimation process was needed in order to be able to predict the software costs early. They experimented Cocomo and IFPUG previously, however for larger sized software applications, these methods were not found useful for cost estimation. They found COSMIC as more repeatable and faster than the IFPUG method and claimed that COSMIC could be automated in a further step. They produced several productivity models for the same software: one for interfaces modules and one for algorithmic modules. Besides Embedded Software group in Renault [27][28] wrote rules to map all the COSMIC concepts to share it with their suppliers. As the software functional size measurements were performed by several measurers, they claimed that it was mandatory to write textual guidelines. The reason for this was to define very clearly and without any ambiguity, the way of measuring functional sizes. Then they also experimented to check the reproducibility of functional size measurements that were performed independently by different people. They announced that manual measurements were very reproducible with at the most 1% of difference with this way.

If we are developing a military avionics software that is a kind of embedded software with safety and reliability concerns, it is highly probable that extra tasks are necessary for standard software development. Because some specific standards like DO-178[136], DOD-STD-2167A [137] (Military standard defense system software development) are needed to be complied. These specific standards defines a process that requires specific documentation with the structural information of the related military avionics software components. These systems are generally integrated systems therefore test phase may require a number of extra sub-phases that is different than many other application domains. Song et al. [140] mentioned a similar approach in their research. Their main consideration in their research is to investigate maintenance phase. They claimed that other than the size of software, the number of CSU and Configuration Software Components should be input parameters for the effort analysis. A representing decomposition in parallel to their study is given in Figure 28. Therefore a CSU level software is needed to be tested at unit, component, item, system and platform levels. Due to the requirements of high reliability in safety critical domain, the testing effort has a considerable impact upon the effort. Besides in terms of safety considerations, the critical level of software guides the amount of required software test effort. For this domain they categorized the test effort as Ground test, Flight test and found that Flight test needed more effort. Finally the experience of people was found as an effecting factor for the safety critical software domain. They proposed a new forecasting method for effort estimation that was based on software size but using domain specific information.

2.3.3. Need for Database Reliability and Effect of Datasets

The term 'Reliable data' is used to mean that both collected effort data and size measurements are consistent and repeatable across the organization, in terms of personnel, projects and processes. In a reliable dataset, data is extracted from a controlled repository which has been established using the results of projects that are managed via similar processes and tools. Furthermore, the procedures for the construction of dataset such as data collection, measurement and analysis are well defined and they are performed and monitored by the same personnel [143][28][74].

2.3.3.1. Need for large and Consistent Data-set

With the aid of repositories like ISBSG [142] and the ESA [141] datasets, companies can compare themselves with respect to their productivity with others or may use the results of these world-wide data-sets for their estimation. However, the advantage of the multi-company projects database is still under debate. Even assuming that there is enormous number of data for similar projects, there is insufficient data to construct and to test an effort estimation model. Besides, data collection methods are not uniform across different companies and differences in processes and practices are not reflected well in these datasets. Using single company data will minimize the differences. Furthermore, to ensure the reliability of the data an organization should have well-defined requirements specification documents.

Mendes and Lokan [68] focused on all previous similar studies and analyzed their results and specific data-set problems. They claimed that a minimum parameter set was more applicable if collected completely and appropriately. For example; ISBSG database included 88 different variables to be collected. They proposed that only 21 could potentially have an impact on effort. Other problem related to data-sets was that most of the variables in these sets had more than 40% of their values missing, therefore excluded from the analysis. In order to use these incomplete data sets, imputation methods were proposed [144].

Dery and Abran [9] made a research about the consistency of the information contained in databases by using ISBSG database.They claimed that, for estimation models two prerequisites had to be conformed:

- There must be enough historical data to assure statistical validity,
- The data must be homogeneous enough to provide meaningful interpretations.

In their analysis they controlled effort parameter and found that, one system included effort from initial planning to full deployment, while another only reported effort for the programming and testing phases. So, total project effort reported covers a variety of combinations of phases. Besides, on the dataset, there were missing effort values on some phases. The missing data in many of those complementary explanatory fields led to much smaller usable samples with less statistical scope for analysis. Even worse, for some projects, the sum of each individual effort phase might not be equal to the total project work effort. With more than one field to designate specific information, fields may contradict one another, leading to inconsistencies. In these cases, data analysts must either make an assumption on which field is the correct one or dismiss the projects that has conflicting information.

In another study performed by Abran, Symons and Oligny [73], a similar situation was noted among the list of problems that created difficulty in building the size & effort relationship. They claimed that, in cross-company datasets, there was a non-homogeneity in the standards across organizations for data collection of the effort variable, therefore this situation directly effected creation of database. Since each organization had their own practices and abilities in defining project phases, even though the reporting was based in their best effort to match the ISBSG definition of project phase, it might not be possible to note all recorded information accurately in cross-company database.

2.3.3.2. Need for Measurement Quality

There can be significant subjectivity in measurement. Although the availability of standards for size measurement would help to improve the quality of measurements, since different measurers may obtain different results, the validation of size measurement and removal of the inconsistencies between measurers are necessary. To ensure the reliability of the size data, all project measurements should be performed by an expert measurer who had also been trained or certified in size measurement and the results should be verified by other measurers and measurement reference guides are necessary [28][29][74].

2.3.3.3. Tools and Structure of Data

Database: In some researches, groups or firms explicate their best practices for data collection. For example Springsteen et al, prepared a report for the Department of Defence in USA [138] and described the use of data collection and analysis tools and data repositories among the industry organizations contracted. According to their report, tools are needed to collect basic data, to calculate measures, and to generate charts, graphs, and other forms of metrics reports. Although they have specific recommendations as a tool, they do not insist on any of them, since the pricing issue depends on the company. They pointed some problematic areas for metrics program to be researched further;

- Identification of standard reports for the corporate level,
- Creation and reporting of abstract information in a valid manner across different application domains,
- Investigation of the impact of reuse and COTS software on metrics data and analysis,
- Decision of the beneficial metrics,
- Creation of tools and procedures to make measurement non-intrusive.

Besides, they also classified collected size and effort metrics at different levels of company and claimed that while at the project level, metrics describe work status, expenditures like effort, and other factors that are necessary in daily decision making and planning, at a division or domain level, metrics are needed to synthesize trends and improvement issues.

Similarly, Braungarten et al. from Canadian Software Productivity Center Inc. [139] disposed a list of general criteria to satisfy the best when defining a software measurement repository facility. So this facility should:

- be *easy to use* so that team members can update and report their data with a minimum effort,
- be *flexible* so that one can change its structure if new data is collected,
- *interface to other tools* such as configuration management and project management systems, ideally to simplify data collection and reduce repetition of data within the company,
- be *large enough* to contain substantial historical corporate data.

Data Collection: Galorath which is the supplier of commercial data collection tool gives some recommendations about data collection method. The first consideration is motivating potential data providers to participate in data collection process. For doing this it is necessary to provide data collection forms and instructions beforehand. If providers give definitions themselves, this will increase their participation in the process. Another consideration will be making a face-to-face interview to confirm that data is realistic and valid.

Phases and Activities : Briand and Wieczorek [61] in their resource management study , mentioned that to monitor and control the project, we need process model for the development and data about effort actually spent on major activities. With this way it would be easy to understand relationships among project attributes and to capture them through resource models. They recommend the data collection based on a process model that describes the relevant activities, artifacts, and resources on an appropriate level of granularity. Besides they recommend the characterization of effort expenditures aims at finding out where, when, and how resources are spent on a project. The reason for this is if enough data from various projects is available, staff and effort planning of new projects is easy to estimate. It is then possible to estimate which proportion of the overall estimated effort will be spent in each phase, for which activity, and how much effort has to be spent for completing a product. Under the best industry practices subject, they commended that regardless of the specific estimation methods to be used, software organizations need to define and integrate an explicit, repeatable estimation process into their development practices by defining procedures for data collection and analysis, and selecting appropriate estimation methods, and applying appropriate system-sizing mechanisms.

Yang, Bohem et al., [75] performed a study on phase effort distribution data of 75 industry projects, by using the China Software Benchmarking Standard Group's database [75]. They claimed that poor effort estimation and allocation cames from lack of recognition to process variations and lack of understanding to process effort distribution patterns. They found different distribution pattern for Chinese software than that was found in another study based on Cocomo. They included the activities taken into account that is listed in Table 33. They observed FP-based software size and application development type were two major factors to be considered when adjusting effort allocation.

In study about "lessons learned in work effort metrics" Wiegers [76] suggestedtracking trends in the distribution of work activities to improve the understanding of how to develop software. He used these information to set quantitative improvement goals, to identify opportunities that could increase productivity, and to build heuristics to serve for estimating new projects. He made a classification of work effort by using six development and four maintenance phases in his research. Development phases were: preliminaries and project planning, specification, design, implementation, testing, and writing documentation. The maintenance categories he applied were: adaptive,fixing bugs, adding enhancements, and user support. Other then the development categories, one more category was included for capturing time spent for routine execution of activities. On this category he collected effort information for certain activities like generating monthly management reports from an information system. His conclusion on his study was that; collection and analysis of software work effort metrics provides multiple benefits to software group. Many of the team members deduct information for project planning, for the estimation of tasks from this database. His categorization is given in Table 33 at Appendix. Another conclusion of his study was these type of datasets provide a quantitative understanding of the group's development process. Besides it was possible to supervise evolution of the process during life-cycle. Even more, this method allowed team members to compare where they think they spend their time with where they actually spend their time. He also explicated their data collection philosophy and process.

Application Type : If structure of database established based on the information of application types, companies should define their classification of application types. There exist different classifications for application types. For example : Morris categorised based on the type of end-user and the services provided by the software [161]. His classification is given in Figure 12.

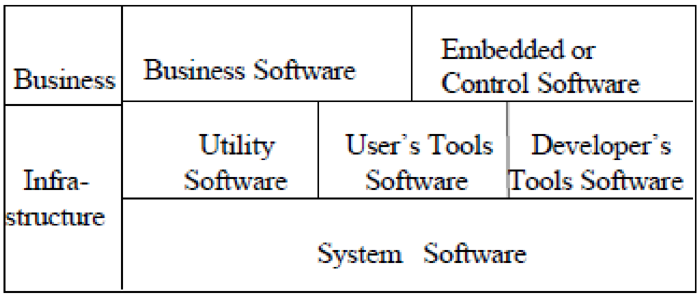


Figure 12 Categories of Software according to Service provided by Morris

2.3.4. Effect of Data Analysis Method

The majority of the effort estimation models proposed in the literature are based on statistical methods, in particular, the regression analysis method, and use total size as the driving parameter. Regression analysis is one of these parametric approaches that use historical data for curve-fitting. Reliability of fit of the proposed curve can be measured using different parameters like mean square error. An alternative approach that uses historical data for the generation of a specific estimation model is the ANN. The models based on ANN capture the underlying relations among input variables by machine learning.

In literature many studies compared the results of these analysis methods. Tronto and Silva [49] investigated and compared the Artificial Neural Network and regression based Models. The neural network was implemented with 1 input, 9 units in the first hidden layer, 4 units in the second layer, and 1 output neuron. They compared the accuracy of models base on Mean Magnitude Relative Error (MMRE) and coefficient of determination (R^2) values. On R^2 dimension they found ANN models useful. However in terms of average error, MMRE, there was no much difference. They first selected four independent variables to compose their models by using a statistical analysis package. Among them only “thousands of delivered source instructions” is found to be the most important variable and used to build the ANN and regression models. In another similar work, Tronto and Silva tried several [49] neural network models with a number of 23 neurons and developed several regression models by minimum 3 multiple inputs. These inputs were effort driver variables. They concluded that neural network based models were able to capture the parameters that had an influence in the development effort. Therefore ANN results in better accuracy than those $^{\text{obtained}}$ with multiple regressions, and simple linear regression.

Berlin and Raz [46] examined the linear regression and ANN techniques in IT networks. They tried size, complexity and productivity values or a combination of these as an input to neural networks with one or two hidden layers and compared the resulted accuracy. For ANN models best result was obtained using combination of these three as an input with two-hidden layer. For regression technique exponential transmission was selected best for that data set. Besides, they found regression models better compared to ANNs. Therefore they claimed their results were opposite of Tronto and Silva [49], i.e ANN was not outperforming to regression techniques. They asserted that, using the number of files as product complexity improved the linear regression model such that it performed strongly better than ANN model.

Finnie and Wittig [14] compared CBR with different regression models using FP and artificial neural networks on a large database. They reported a better performance of CBR when compared with different regression models based on function points. Artificial neural networks, on the other hand, outperformed the CBR approach. The neural network inputs they recommended were the system size, several system characteristics and the programming environment.

2.3.4.1. Regression Analysis

For the construction of the single input models, regression techniques were implemented using total functional size as the input. Regression models are represented in a defined form. For example, for a linear regression model the effort has the general form:

$$E = b_0 + b_1X_1 + b_2X_2 + \ldots + b_kX_k \qquad (1)$$

Where, for a set of observations, the variable E is a linear combination of an offset b_0, a set of predictor variables X with matching b_k coefficients. If more than one X parameter is used, then multi-variate linear regression analysis is assumed, otherwise it is assumed to be a single input. These b_k values are derived using statistical techniques and tools. *Linear regression* undertakes to find linear relationship between one or more of these X predictor variables and a dependent variable, minimizing the mean square of the error across the range of observations in the data set. Many existing effort estimation models based on this approach. However, the main disadvantage of this technique is that it is vulnerable to extreme outlier values.

Linear regression models were the first ones that was recommended for effort estimation models. In literature a variety of different models have been proposed that were generally formed by using Ordinary least-square regression (OLS) technique. This technique presumes a functional form interrelating one dependent variable, with one or more independent variables.One has first to specify a functional representation of relationship between dependent and independent variables. The least squares regression method then fits the data to the specified representation by trying to minimise the overall sum of squared errors. To carry out an OLS Regression, a confidence level must be set according to the precision requirements of the research. In the majority of previous surveys in this domain, the confidence level of the OLS Regression analysis is set to 95%.

Although regression is a standard method for building a relationship among dependent and independent variables, it faces major challenges. The model is frequently overfitted. This occurs when unnecessary predictive variables remain in the model. If some of the independent variables are highly correlated, this situation may cause high variances and covariances in coefficients and result in poor predictive performance when user tries new data. In order to get rid of such situations, some techniques that reduce the number of predictor variables are proposed [85].

2.3.4.2. Artificial Neural Network

For ANN based models, observations in the database are not considered to be independent. This analysis method is different from model based regression analysis techniques in that there is no need for the model to be specified beforehand. In the learning phase, with a number of iterations, dependencies between input variables are arranged according to both output variables and all other observations.

ANN is a system that uses the ideas about the operation of biological neural networks. ANN is an application of artificial intelligence principles and is formed by using the information about how biological brains operate. A brain is composed of many neurons that receive a stimulus and trigger a response to another neuron. The response from the other neuron can trigger other neurons and so forth. At the end, this chain of neural activation results in body response like the remembrance of a memory, movement of a muscle.

Similar to the brain, where an input (stimuli) can be related to an output (action), an ANN can model the relations between inputs and outputs by using neurons inside. These neurons are called as nodes. ANN is composed of a number of layers at which a number of nodes exist: input layer, hidden layer and output layer. The final equation at the output can be represented by considering the activity of each neuron. A typical mathematical representation of each neuron is summarized in Figure 13. This representation is from study of Rios [145]. A final network model can be formulated by taking into account the mathematical representations of all the neurons in the network.

Input layer gets the signal and delivers it to the inside of the neural network, Hidden layer deals with the signal and calculates the weight by using neurons. Generally, only one hidden layer is used. The reason is that multiple hidden layers make the neural network quite complex. The number of hidden layers of ANNs implies a classification: The single-layer networks and the multi-layer networks. The "hidden" concept comes from the fact that the performance of the network is not open to user. Output layer is responsible for receiving the signal from the hidden layer and generates the appropriate result. Each node works independent from each other. The architecture of each ANN is based on very similar process elements, that are neurons, which work in parallel and perform the processing.

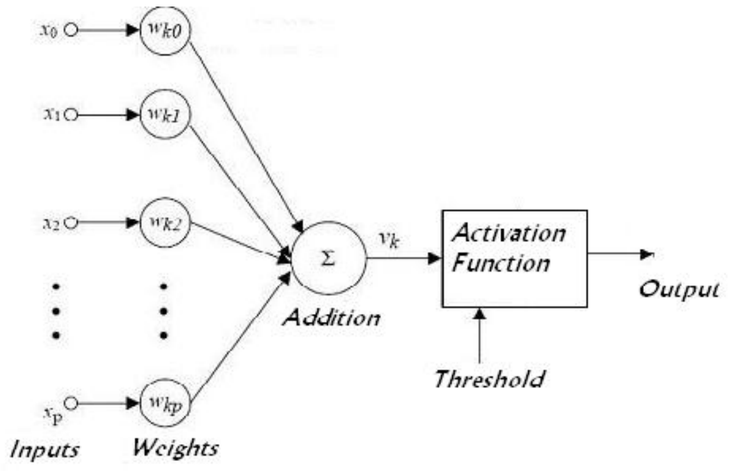


Figure 13 Mathematical Representation of Neuron "Adapted from Rios Daniel Study

The neuron calculates a weighted sum of its inputs and generates an output. This process is performed in all the neurons. The resulting architectures solve problems by learning the characteristics of the available data. For the output computation in neurons an "*activation function"* is used. There are a number of different types of activation function. The most common ones are step, ramp, sigmoid and Gaussian function. However proper selection of this function is still under investigation [146].

The performance of an ANN depends on its architecture and certain parameters such as the number of layers, the number of nodes in each layer, the transfer function in each node, learning algorithm parameters and the weights which settle the connectivity between neurons. So inappropriate selection of these parameters may cause serious difficulties in network performance and training.

2.3.4.2.1. The Number of Hidden Layers and Nodes

In order to build a neural network, one of the biggest issues is the determination of the appropriate number of layers and nodes. For the best network performance, optimal number of hidden-units must be selected for the generalization of results [147] To decide the appropriate ANN model, two approaches can be applied: one is the trial/error approach and the other one is the application of the Genetic Algorithm [148][149]. The second approach is feasible for larger data sets.

If very few neurons are used, then network can't approximate the desired output function well. However, if too many neurons are used, then over fitting can occur. In other words, the system only works on the few test samples that have been provided. Small error is observed but network cannot converge and become useless for any other input.

In many studies, selection is recommended by considering the number of training samples [150]. Therefore, more training samples require the usage of more neurons to represent the complexity of the output function. Complex output patterns cannot be defined by a small number of hidden layer neurons. For the decision concerning the number of neurons, a rule-off thumb was applied. According to Heaton "The number of hidden neurons should be less than twice the size of the input layer" [147].

2.3.4.2.2. Backpropagation

The fundamental concept using neural networks is to achieve the best learning algorithm to adjust the weights in multilayer. Back-propagation learning algorithm is re-invented in 1982, by Rumelhart [151]. This technique originally was discovered by Werbos in 1974 [153]. Back-propagation is an algorithm to adjust the weights from layer to layer backwards to reduce the errors at the output during the training process. After a number of iterations, proximate representation of the effort was formed based on the each network training input.

The tool user selects "Learning rate" parameter, that directly defines the convergence speed of the back-propagation algorithm[152]. If learning rate is selected as higher, this will cause an over-correction and network may not converge to a state. However, a small learning rate will increase the running time and create problems in converging. Training phase aims to finalize the values of the weights and biases that minimize the output error. Training time may be arranged with some parameters such as iteration number, or specific error value.

A major limitation in the usage of a neural network as an analysis tool is that it requires a large dataset for training. For small data sets, it is necessary to make the training more efficient. One way of doing this is to repeat the training process several times with a randomly selected training set, then the model is tested with the validation data [154].

2.3.4.3. Effort Estimation Accuracy

In order to determine that one effort estimation model leads to better estimation than another. A large number of different prediction accuracy statistics have been used in the literature.

The Mean Magnitude Relative Error (MMRE) is an average error used to indicate the relative amount by which the predictions over or underestimate the real value for the model. The magnitude of relative error (MRE) is the value calculated by dividing the difference between the actual and the estimated values to the actual value. The mean MRE (MMRE) is therefore the mean value for this indicator over all observations in the dataset. A lower value for MMRE generally indicates a more accurate model.
The coefficient of multiple determinations, R^2, on the other hand, gives the percentage of the variation that can be explained by the independent parameters. If R^2 approaches 1, it can be said that a strong relationship exists between the independent and the dependent variables.

The prediction level parameter PRED(k) represents the quality of the predictions. It defines the percentage of estimated values within k% of the actual measured values. So unlike MMRE values, we need higher PRED values in our estimation models. Contemporary expectation of a good model is the achievement of PRED (30) = 60% [50].

CHAPTER 3

3. EFES: EFFORT ESTIMATION METHODOLOGY

For an approach to achieve its benefits and be useful, it is essential to provide a systematic way to implement it in an organization. The defined methodology provides a disciplined guidance for organizations to perform all tasks in relation to effort estimation.

This chapter presents the EFES method we proposed for organizations. First section of the chapter discusses the requirements of the methodology. Sections 3.2 presents the operating principles of processes, Section 3.3 provides Database structure, 3.4 defines Roles and Actors in methodology. Finally 3.5 presents the procedures for processes and artifacts.

3.1. Requirements Of The Methodology

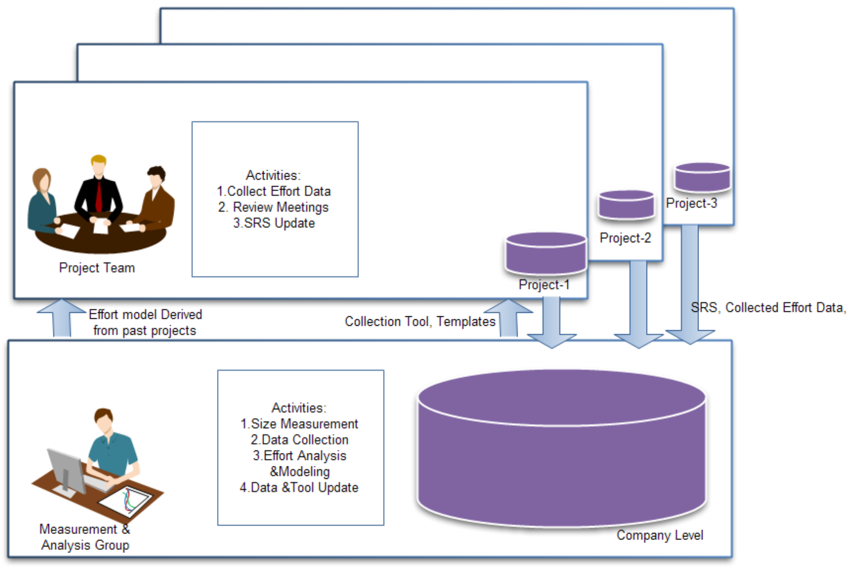


Figure 14 Top-View of Effort Estimation Framework

The aim of the methodology is to define all necessary steps, related artifacts, templates, roles and responsibilities for the processes for a company. It is executed in two levels:The project level and the company level. At project level, project managers and project teams are responsible for performing the actions. At company level, Measurement and Analysis Group is the coordinator,controller and performer of whole procedures.

The methodology aims to build a reliable database. To do this first, information needs should be defined. Then, how and who will collect, analyze and publish will be decided. The selection of technology level available for collection, analyzing, storing and publishing depends on the company. The information obtained from projects, analysis results and all the other related documentation can be kept on either at a standart server with basicly arranged folders or at an information system with the selection of user-friendly interfaces. Both methods will enable to protect the information obtained from previous projects and to use them in the future.

Similarly, the specific subjects that need to be analyzed depends on the company.The company may select to investigate the amount of requirement change effort, effort needed per specific task, wasted effort, unplanned effort, unexpected events, demos, etc.

Building historical data requires time. To have a sufficiently comprehensive set of data, it takes many projects to be finished. Then performance and productivity measures from completed projects are used to make reasonable assessments for new projects. Different needs require different amounts of data to be collected. Simple project tracking can actually be performed without much data. It is just enough to accurately track and monitor the progress versus deadlines. For the estimation of future work, an agreement on standard measures and setting up a repository for the ongoing projects is necessary. After significant number of finished projects, we will obtain the first set of historical data to use for the second step. Naturally, the confidence and accuracy of the estimates depends on the number of projects and data details. If only project size and effort are collected, we can not expect to estimate well. The problems concerning the operationalization of effort estimation methodology in practice are to capture the status and background information of company, to represent and to store it in a reusable form, and reuse efficiently and effectively in future software projects. Gathered information have to be continuously acquired and integrated into the available knowledge.

In this study, we defined the formation of this historical database for a specific company, moreoever an approach for data measurement, collection and analysis is introduced. This approach mainly is constructed by considering feedbacks of developers in the company and investigation of available data. Since processes are defined as generic, it can be used for other companies by tailoring. To favor this opinion and to guide other companies, a number of templates for the execution of processes are given in the Appendix. So companies may create a comprehensive processes to produce their specific executions. For example for Effort Collection process uses a predefined “General List” that defines phases and activities. For our specific company by using that information a WBS structure is formed for data collection. However, another company that mainly deals with maintenance activities may form another WBS structure to collect effort, like in Table 33 .

The main operation and activities can be summarized as follows:

- The Project Team uses previously published company WBS templates for their new project analysis and plans. These templates draw the essential groupings for phases and recommended activities under that grouping.
- The Project Team collects effort data periodically during the development. The recording structure also allows to track the current status of the project. To support these goals, the collected data is never used to evaluate the members of the project team.
- Measurement and Analysis group organizes and controls the data formation during the execution of the project. By using the available data from all the projects in repository, they create an effort model. To do this they use both project related and unrelated information.
- Calibration of the model and tools are performed periodically, to announce the current situation.

3.1.1.Detailed Requirements:

After a literature review, and the results of the exploratory case study, the requirements of this framework are defined as follows:

3.1.1.1. Data Requirements:

- Up-to-date information should exist in database[74][119]:

All the information about the project should be located in a common repository and be accessible by the responsible personnel. Measures and historical data should be available when needed therefore they should be periodically put into repository.

- For a company based analysis, all projects should provide the required information with valid and complete data[9][13][60][168].

Validity requirement should be satisfied to ensure that everyone understands the terms in data collection, the definition of each variable actions, phases, tasks, the necessity of data collection, the real aim behind the collection so that a common consensus is created. Information comprises all the projects and the sources.

- Collected data should be checked to prevent missing data problem or other errors [13][9][169][170][171][29][172].

Group of experts should be responsible in the company to control the problems in database. They should give feedback to data providers to make the information up-to-date, consistent etc. and should check to prevent missing data problem or any other errors. Recorded values conform to actual values. To improve data quality and ensure that measures usage is evolving, it is recommended that the contents of the history database should be frequently reviewed. This should be done before each major reporting cycle.

3.1.1.2. Effort Collection Requirements

- Effort data should be collected, based on the defined phases, and the activities [61][76].

If there is a non-homogeneity across the projects for data collection of the effort variable, total project effort reported covers a variety of combinations of phases, that cause analysis to be incorrect. Planning at a more detailed level that is defining shorter activities and smaller tasks, increases the accuracy of the estimates and reduces the size of the estimation errors. Besides unrelated activities performed during the execution of project should be moved-out of analysis to prevent noisy information in effort value. Planning at a more detailed level that is defining shorter activities and smaller tasks, increases the accuracy of the estimates and reduces the size of the estimation errors.

- To compare the expended real effort, project related and not related efforts should be noted in detail [24][26][60][156].

Team may attend to some presentations, other trainings, supportive activities etc. that is not a part of the project. The time expended for other projects, and unrelated tasks should not be inserted into effort collection tool. Some part of effort may be correlated directly tothe size of the functionality delivered. But for some other type of efforts, like documentation, meetings, demos it is not possible to create a relation with functionality. Overtime work is recorded. The time expended for other projects, and unrelated tasks should be differentiated.

- Overlooked tasks or unexpected events, should be recorded [23][24][26][60][156].

Though, it is named as “unexpected events”, these situations may be commonly experienced in the company. For example ,waiting for the hardware to operate, lack of SW developer, unavailability of a tool, requirement changes etc. that prevents to proceed or that requires rework.

- Efforts collection review should be performed with the periodicity of a week [76].

If it is collected more than a week, results may not be reliable. Team participation improves data quality. Using the estimates for project monitoring and control increases the commitment of the estimators (i.e project team) and reduces the occurrence of estimation errors' Size Measurement Requirements.

3.1.1.3. Size Measurement Requirements

- Repeatability of Measurements is necessary [28][60][77][173][174].

To make sure on repeatability on measurements, other than standards and manuals, a written procedure will be useful.

- Validity of Measurements [74][119].

A group of people may be assigned for this purpose for checking the correctness of the measurements. Besides, certified measurers' existence is beneficial.

- Requirement Specification document should be well defined[29][76].

Incorrect or missed information results in inaccurate size estimation. Following a checklist to make a review can be a good suggestion. Besides a standard and tailoring method is published according to the size measurement rules: layer definition, boundary definition etc.

- Measurement Results should be documented [29][174][175].

Documented measurements allow this process to be controlled and more repeatable.

3.1.1.4. Analysis Requirements

- Tool should allow multi-variate data analysis [82][83].

Since we aimed to improve the correlation by using size components, tool should allow the analysis of multiple input case to correlate the effort output.

- Easy user interface will be beneficial.

Since this is a recurring process, after years a large dataset is used for analysis purposes. Therefore, tool shall perform its operation on a large database with minimum number of effort.

- Allow to calculate effects of unexpected events on effort in terms of percentage [23][119][156]

This will be performed by any tool selected by Measurement & Analysis Group.

- Outliers should be investigated [176][177]

This analysis results in either pointing a problem or finding a possibility for improvement of the project.

- Effort model should be formed for each application domain type [75][178]

Each type of application domain type should be arranged according to the development tools, some development processes ,the language and the platform.

3.2. Processes of Framework

Framework consists of five processes. These processes can be performed at company level or project level.

For the requirements of the methodology we decided the related processes and responsible people for the execution. In Table 6, we indicated this traceability. Measurement and Analysis group (M&A Group) is the owner of the overall methodology that controls the execution of processes, performs the measurements, create effort models. Managers also take part in execution of the processes at decision. points: for company level decisions Software Managers and for project level decisions: Project Managers are responsible.

Figure 15 provides a top-view of all processes. Details of processes are given in part 3.5 with specific flowcharts. All the steps of Data Analysis and Calibration are executed at company level. Data Analysis process uses the effort collection results and BFC size measurements of all the projects for analysis. Then it produces “Effort Models” for each application type. Besides, other important datacollected during project are analyzed for the final effort estimation of the new project. Calibration process is executed at company level periodically. Two different calibrations are performed by using this process. One of them is the calibration of effort models by including last finished software projects’ data in a definite time. The other calibration is carried out to update the templates for the lists of data collection. These lists are predefined WBS structures for defining phases, activities, unexpected events, unplanned events etc. The period for the calibrations mainly is decided by the Measurement and Analysis Group, considering the arrival rate of the available information.

Table 6 Traceability of Requirements to Processes of Methodology and responsible people

Req. No	Related Process	Responsible People
001	Effort Collection Process, Calibration Process	Project Team, M&A Group
002	Effort Collection Process	Project Team, M&A Group
003	Effort Collection Process	M&A Group
004	Effort Collection Process, Calibration Process	Project Team, M&A Group, Software Manager
005	Effort Collection Process, Calibration Process	Project Team, M&A Group, Software Manager
006	Effort Collection Process, Calibration Process	Project Team, M&A Group, Software Manager
007	Effort Collection Process,	Project Team, Project Manager, M&A Group
008	Size Measurement Process, Effort Estimation Process	M&A Group
009	Size Measurement Process, Effort Estimaton Process	M&A Group
010	Size Measurement Process, Effort Estimation Process	M&A Group, , Project manager
011	Size Measurement Process	M&A Group
012	Data Analysis Process	M&A Group, Project managers
013	Data Analysis Process	M&A Group
014	Data Analysis Process, Calibration Process, Effort Estimation Process	M&A Group, Software Manager, Project Manager
015	Data Analysis Process	M&A Group, Project Manager
016	Data Analysis Process, Effort Estimation Process	M&A Group, Project Manager

Effort Collection, Size Measurement and Effort Estimation processes are implemented for each project. Effort Collection process is used for two purposes. During the project the collected data is used for project tracking, monitoring and control. But the main aim is using the final values obtained at the end of the project. Size Measurement process is used to measure the BFCs of the project before estimation. However ,it can be repeated during the development at specific milestones for tracking. FS effect is included in size measurement before applying the effort model. BFC components of sizing method are grouped for the analysis phase. Effort estimation process applied only once and combines the results of other processes to generate an estimation for a new project.

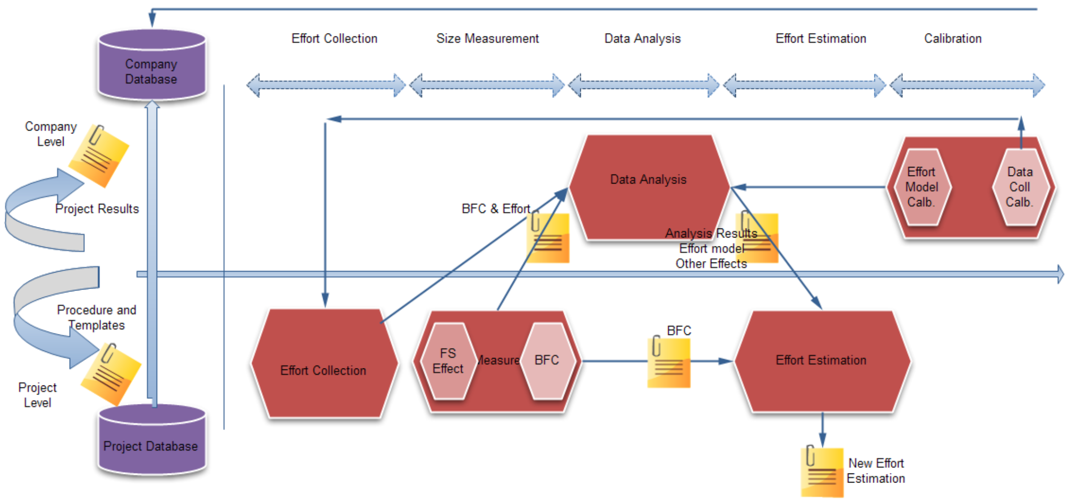


Figure 15 Effort Estimation Process Framework

The methodology is a combination of formal and expert estimation methods. Data collection structure is based on expert knowledge obtained from previous projects, but effort model is formed by using modeling functions.

3.2.1.Effort Collection:

Effort collection is performed for each software configuration item (SCI) in project. It is continually performed during the project life-cycle. It is performed for eachSCI. System breakdown for the decomposition of system to define SCIs is given in Figure 16. This figure is presented in DOD-2167 [137]. System is decomposed into Segments first, where a Segment consists of one or more Hardware Configuration Items and one or more Computer SCIs. An example decomposition in parallel to this classification is given in Figure 28 at Appendix 1. Some military standards [137] also append

Interface Requirements Specification as a part of this disintegration. Typically, a Hardware Configuration Item is a printed circuit board or microcontroller. A Computer SCI is further decomposed into Computer Software Components (source code files) and finally Configuration Software Units (CSUs) (functions or modules).

For effort data collection differentiation of phases is determined at company level and published as “General Lists”. A general list suggested in parallel with the processes of IEEE EIA 12207 [155] is given in Table 7. Some example lists are given in Table 31-34 at Appendix.

Sub-tasks/activities needed are defined for each SCI at project level by the project team, and controlled by Measurement &Analysis group. For this step a General List is formed into a predefined WBS structure for data collection by considering the project information and by detailing the activities. Then this structure is used to track the status of the project by each member. Every member in the project record his/her process for the last week.

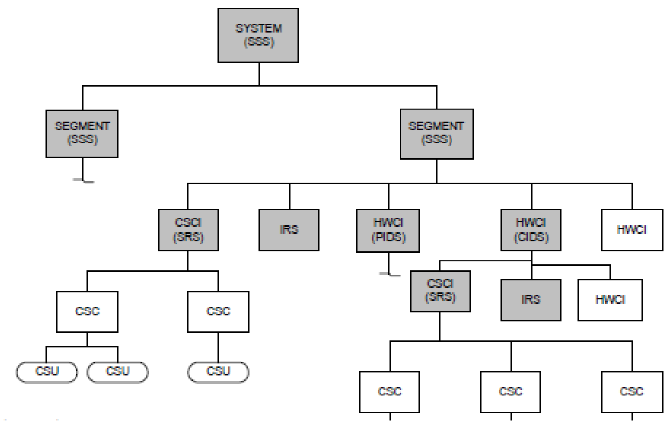


Figure 16 Decomposition of Project Items (From DOD-2167)

In order to reach effort value related to functional size, it would be beneficial to differentiate the activities. For this reason, the effort spent that effects the project’s total effort, but not productive for the project should be separately recorded. These factors may belong to different categories: It will be directly related with the operation of the organization, may depend on other stakeholders, may depend on environmental conditions or people. For example if company makes a decision like compiler and utility changes, hardware upgrades, media conversions where no new or changed functionality is required, this change may require extra amount of work, that is not a concern for customer. Final effort reached after such changes will never be correlated with size.

Similarly, change requests from customer may cause an alteration in the design that also increase the planned effort. Another example will be related to hardware problems. If the developer waits a hardware to start an activity, then this information should not be noted under expended real-effort value. Recording the amount of such time-lag that prevents to proceed will be beneficial for the new projects' estimation. When estimating the effort required for a new project, these unplanned activities are included in calculations as an extra contingency. Therefore, during the development, all similar situations in each project should be recorded for future use.

Table 7 A Part of Recommended General List Structure based on IEEE EIA 12207

Phases &Tasks	***Activities***
Software requirements analysis	-Preparation of prototypes to specify the requirements of the system -Working with System Engineers to understand functionalities
Software coding and unit testing	-Coding/Updating the Program -Writing and executing unit tests - Interface testing
Software integration	-Integration at SW level -Integration at HW Level
Software qualification testing	-Coverage Testing -Qualification of Software Component - *Software Safety Assessment Testing*
System qualification testing	-System Level testing -Test According to a standard -Platform level testing -Defect Removal During Qualification -*Safety Validation at System Level*

This process starts at the beginning of the project and continues until the product is delivered to the customer. To prevent errors, reviews are included in the process.

3.2.2. Size Measurement:

This process is used to define and count the BFCs of the measurement method. Measurement manuals define the stages of the methods in detail and in a very explanatory way. Final measurements are reviewed by other members of the measurement and analysis group, to satisfy validity. To satisfy repeatability, besides standard manuals, company measurement procedures should be published. These templates define rarely observed cases or assumptions to be applied in size measurements to support measurers. If during the measurements there occurs a discussion about a case, then the final decision should directly be included in these procedures.

The measurement process applies the rules of selected measurement method and tailoring. Process is implemented for each SCI. For size measurement, all methods require to define application boundary. COSMIC method also requires to define the layers. Our recommendation is to arrange SCIs such that each SCI should include only one layer. Otherwise, the number of SCIs should be increased.

Tailoring is applied according to the notes for Application Domain for each SCI. To exemplify this concept we can give an example situation for the size measurement of Embedded Device Drivers software: if an embedded application requires a register to be updated in order to light a panel LED, then a decision needs to be made as to whether writing that register should be considered as outputting a value, named 'exit (X)', or should be considered as recording a value, named 'write (W)'. Some measurers may accept this situation as "Write". However, since writing that register is only a part of hardware-board design, and the aim is to display the information via the LED, then we named this measurement as 'exit (X)'.

Since the size measurement depends on the available information, SRS should be complete enough. Even with all the information available, the accuracy of estimates will rely on the effectiveness of the estimation processes and models employed. However, the more detailed the information, the more accurate the estimation can be. For this reason a template should be fixed. IEEE-830 is found suitable for FSM methods[127][29]. SRS document should be reviewed for the correction of requirements defects, to remove ambiguities and inconsistencies.

However, SRS template may include specific parts inside to improve the measurement method. For example boundaries, layers, external devices or users may be defined in some parts. If modification is not desired, then this information will be combined with quantitative measurement results that allow measurement &analysis group to have his measurement results documented in a consistent manner. Documented measurements allow this process to be controlled and more repeatable.

The initial measurement was performed by an experienced measurer, than another group of experts review the initial measurement results; most preferably, these measurement experts should be certified for the specific measurement method.

The FS consideration on effort is included by re-evaluating all the measurements such that; for each functional process measurement, the similarity level of each functional component is compared with their respective components on other functional processes. At the end of this process, we have two size measurement results. The one with the FS related is the suggested one for data analysis process.

After a number of steps of validation, preparation for data analysis part is performed. Since we use BFCs of measurement methods, grouping of size components and inclusion of this information to measurement results are necessary. Validation of these basic addition is also beneficial to prevent the errors.

Table 8 A Part of Supporting & Extra effort List

Supporting & Extra effort	***Explanation***	***Effect Level***
Requirement Changes	Include extra expended effort not planned for each SCI	Application DomainType
Support to Other Projects	Record support time not wasted for current project	Application Domain Type
Demo	Include this unplanned Effort	Project
12207/SupportingProcess -Documentation	Writing and inspecting the software requirements specification. Writing and inspecting the software design specification. Writing and inspecting the software test specification. Writing technical note or giving presentations	Application Domain Type
Hardware problem,	Include the wasted time in hours for each SCI, if HW not available or HW has problem	Application Domain Type
12207/SupportingProcess-Configuration Management	-CM Activities of CM Personnel -CM Activities of SW Development Personnel	Project

3.2.3. Data Analysis:

This step is performed to find a suitable and representing “Base Effort Estimation Equation” between BFCs and Development Effort. Different tools can be used for the creation of effort model. Tools and analysis methods require specific parameters to be decided. Measurement & Analysis Group must be trained on these subjects to apply them successfully. Process steps have variations depending on the selected tool. The estimation equations should be formed for each application domain. So, the number of equations, i.e. models, should be equal to the number of application types developed by the organization.

Other than analyzing the required data for creation of base effort model, the historical database is useful for reaching some valuable data. So, the percentages of Supporting & Extra efforts; like HW problems, requirement changes are included as a multiplier to reach final effort value. A part of such list is suggested in Table 8.

3.2.4. Effort Estimation

In this process estimation for a new project is defined in steps. It uses the results of other processes. For a new project, Effort Collection’s first sub-process is applied to find SCI identification. Then for each SCI, Size Measurement process is executed. At that point previously brought out model of effort at company level for the application domain that SCI belongs to is used to calculate the base effort value. Effort model and other valuable information are imported from Data Analysis Process to attain Final Effort value.

3.2.5. Calibration

Calibration of effort models should be performed at an acceptable periodicity. Period will be determined by the Measurement and Analysis Group, according to the delivery rate of SCI’s. If during the development of a project, a need for an update in the templates is encountered, this will be discussed and included in the process at company level then will be ready for publishing. For example, if company starts to develop safety related systems that requires specialized processes, techniques, skills and experience, then new activities are included in this list and are published for the company.

3.3. Data-base Formation

Data-base is formed by considering two virtual repository: Company and Project repositories Physically these databases can be settled on the same server. Project Database contains project related documentation as shown in Figure 17.

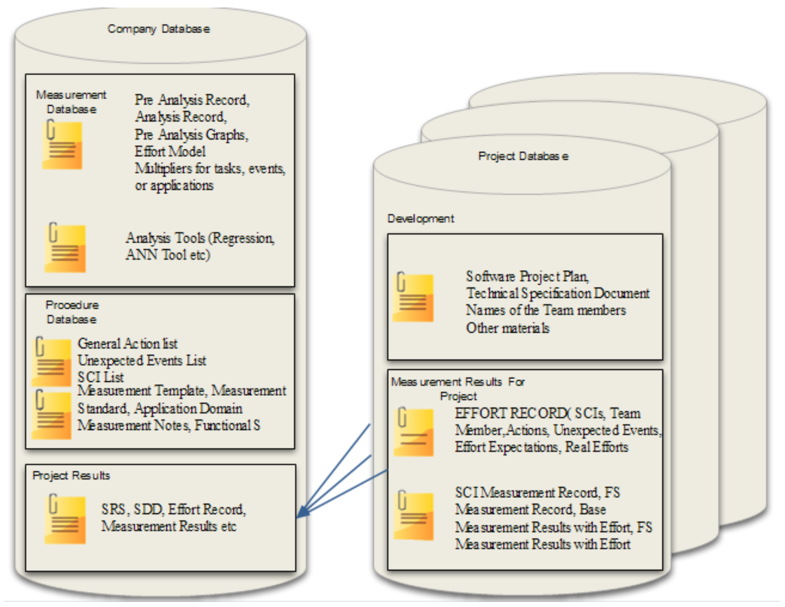

Company Database
Measurement Database
Pre Analysis Record, Analysis Record, Pre Analysis Graphs, Effort Model Multipliers for tasks, events, or applications
Analysis Tools (Regression, ANN Tool etc)
Procedure Database
General Action list Unexpected Events List SCI List Measurement Template, Measurement Standard, Application Domain Measurement Notes, Functional S
Project Results
SRS, SDD, Effort Record, Measurement Results etc
Project Database
Development
Software Project Plan, Technical Specification Document Names of the Team members Other materials
Measurement Results For Project
EFFORT RECORD(SCIs, Team Member,Actions, Unexpected Events, Effort Expectations, Real Efforts
SCI Measurement Record, FS Measurement Record, Base Measurement Results with Effort, FS Measurement Results with Effort

Figure 17 Database Organization

Project related documentation includes: Development Documentation, Measurement Results of SCIs. Company Database contains final versions of measurement results of the projects, tools, standards, last effort models created for each SCI, Final versions of templates etc.

3.4. Roll & Actors

Process definitions require to define a responsible person or group for each execution step. In flowcharts of the processes the responsible people are indicated. In Table 9, the actors of the methodology are given. For each step of the processes “Actors” are indicated in related flowchart.

Measurement and Analysis group leads the overall methodology. On the other hand the owner of the processes is Software Manager. Project Managers not only provides the required reliable information for their project, but also use the final results for the control of project and use effort models outputs for future decisions. Most of the decisions are made by review meetings. These meetings and related actors that will attend are also indicated in flowcharts of processes. Delivery lists for meeting notes are pointed.

Table 9 Responsibilities of the Methodology

Definition of Roles	**Responsibility**
Measurement& Analysis Group	Responsible for Size measurement, its validity, creation of repository, selection of data analysis tools, performing data analysis, effort collection, improving correctness and validity of data, calibration of General List, Supporting & Extra effort list,
Team Lead	Responsible for the management of Software activities for a specific project,
Project Team	Responsible for the development of software, giving exact values for effort they expended,
Project Manager	Uses the collected effort and other information to make estimation, to track the project,
SW Manager	Responsible for the measurement and development activities to be executed, owner of the processes,
Experts	People who has knowledge and experience about the subjects of the project.

3.5. Process Definitions

In following parts "process definition" of each process is given by referencing the related templates, flowcharts and metrics. The metrics defined in these process definitions are not used in EFES Methodology. However they are included as recommendation for company and are aimed to be utilized to improve the methodology in future.

3.5.1. Effort Collection

3.5.1.1. Purpose and Scope

To establish a consistent process for data collection of effort within the company, this procedure guides the effort collection method for a software project. It defines the performing steps with the related responsibilities.

It consists of four sub-processes for effort collection. The first three of them are applied at the beginning of the project. The last sub-process is periodically executed until the final delivery of the project. The procedure is applicable to all of the software projects from all the application domains, for all the development environments. Discrepancies should be noted and discussed with Measurement and Analysis Team.

3.5.1.2. Process Flow Chart

Given in Figure 18.

3.5.1.3. Inputs &Outputs

Inputs:

Software Project Plan,

Technical Specification Document of Product,

Software Requirements Specifications,

Other materials exist or prepared during the feasibility phase of the project or prepared by customers.

Names of the Team members

General List (Table 7)(Includes all project development related& unrelated activities)

Supporting & Extra effort List (Table 8) (Unplanned activities or uexpected events)

Application Domain Categorization List

3.5.1.4. Outputs

Effort Collection Records and Updated Lists are outputs of the processes and recorded in project repository as given in Table 10.

Table 10 Deliveries for Effort Collection Process

Document No	**Document Title**	**Created By**	**DeliveredTo**
PrjNo_EF_CollV1	PrjNamePre.EffortCollection Record	Team Leader	Prj\SWMR
PrjNo_EF_CollV2	PrjNameUpd.EffortCollection Record	Team Leader	Prj\SWMR
PrjNo_EF_CollV3	PrjNameFinalEffortCollection Record	M&A Group	Prj\SWMR
PrjNo_Upd.Gen.List PrjNo_OtherList ReviewLog-21.01.11	GeneralList Supporting & Extra effort List ReviewLog	M&A Group M&A Group Team Leader	Prj\SWMR Prj\SWMR Prj\SWMR

3.5.1.5. Process Execution

3.5.1.5.1. Preparation of Preliminary Effort Record

The Team Leader of the project prepares an action item list for the whole project and for each individual member. The aim is to measure the time that will be spent both in the project development (real effort) and in different work activities and phases. At the beginning of the project, this sub-process can also be implemented by supporting the team leader with experts on the project subject.

a. Identify the team members (if not identified in the project plan);

b. Identify the Software Configuration Items.(SCI)

- *Example Configuration of Software is given at Figure 28(Appendix 1)*

c. Identify action items for each SCI in accordance with the General list given in Appendix. This list should contain both project development related activities, and unrelated activities.

- *Applied Configuration is given in Table 31(Appendix2)*
- *Examples for other General lists suggested in literature are given in Table 33-34 Appendix 2*

d. Identify Supporting & Extra effort items.

- *Applied Configuration is given in Table 35(Appendix 3)*
- *Examples for Supporting & Extra effort List suggested in literature are given in Table 36 Appendix 3.*

e. Assign roles and responsibilities for each action item so that each action item is separately recorded for the individual member.

- *An example template for the effort data collection is given in Table 37 Appendix 4*

f. According to the level of each team member propose an initial effort in terms of days (8 hours) required to finish that item.

- *Values inserted in Table 37 Appendix 4*

g. Distribute and ensure the Preliminary Effort Record to Team and arrange the review meeting. The attendants of this meeting should be arranged such that, all participants of the project that has knowledge about the action items should be present. (Project managers, project team including hardware or system engineers etc.)

3.5.1.5.2. Review of Effort Plans

Each team member examines the work items and planned effort on the Effort Record, especially assigned items for himself before meeting.

a. The team leader presents the lists in the record.

b. The Team leader gathers the information from team. Team leader may reschedule the meeting, if s/he concludes that team members are not adequately prepared, or required members are absent.

c. During the meeting, proposals for additions or removals to action items and Supporting & Extra efforts are discussed and noted on the list. If there is disagreement, discussed item is always included and remained to further discussions of Measurement& Analysis Group and Software Manager.

d. Undefined action-items that are not explicitly given in Appendix General List, will be included in the Effort record and noted in ReviewLog.

e. Team leader sends this list and ReviewLog to Measurement &Analysis Group.

3.5.1.5.3. Review of Effort Records

Measurement and Analysis Team :

a. Checks the suitability of the recorded action items and unexpected event items.

b. Checks whether whole information is prepared and recorded.

c. Decides the group for undefined action-items in the General List. If necessary, holds a meeting with SW Manager for final decision. Decides whether an unexpected event will be included in the record. If necessary, holds a meeting with SW Manager for final decision.

   - *Ex: When a need for Safety Requirements appears, they are included as shown in Table 32, Table 31(in italics)* .

d. Sends the Final version of Effort Record to Project Manager, SW Manager, Team Leader, Team Members, delivers it to the project folder .

e. If projects started then continue with effort collection, otherwise send this table to “Effort Estimation Process” to provide data about SCI List, and Supporting & Extra effort list. Besides, send this table to “Size Measurement Process” for the Prj SCI List reviews.

3.5.1.5.4. Data Collection

The objective here is to collect the real time spent on project development, and to record time for other types of efforts. Therefore, extra working hours spent for the projects during that week and time spent for unexpected items that prevents to proceed are captured as well.

For every finished week; Team Leader and Each team member together:

a. Keep daily records of the time he/she expended on each project activity, to half-hour resolution.
b. Record events that prevents from proceeding in day basis.
   - *Values updated in Table 37 Appendix 4(Real Effort)*

c. Check whether there is a deviation from initial guesses. If guess is lower than the required one note it whether it is a planning error, or an unexpected event occurred. Note the unexpected event effort. (Like an HW error solution, a change in requirement, etc.)
   - *Values updated in Table 37 Appendix 4 (Ex: HW Problem)*

d. Team Leader sends this information to Project Manager, SW Manager, Measurement & Analysis Group.

3.5.1.6. Metrics

The following metrics are applicable to this procedure:

1. Updated Effort (on daily-basis) during review meeting: Shows the effectiveness of the review meeting.

2. Number of undefined action-items that are newly included in the General List.

3. Amount of effort spent for other type of efforts.

3.5.2. Size Measurement

3.5.2.1. Purpose and Scope

To establish a consistent process for functional size measurement of software projects within the company, this procedure is used to design a size measurement procedure that allows the application of the COSMIC measurement method. It uses tailoring approaches according to the application types and application notes for FS reflection. It will be used both at the beginning of the project for size measurement of each SCI's of the project, and during the project development for tracking and monitoring the state.

3.5.2.2. Flowchart

Given in Figure 19 .

3.5.2.3. Inputs

Technical Specification Document

Software Requirements Specification for each SCI

Checklist for SRS Review

Measurement Results Document Template (include Detailed Measurement Results Table)

Size Measurement Manual (Standard for the measurement method)

Tailoring Notes

Functional Similarity Application Notes

Final Effort Collection Record(for each SCI)

3.5.2.4. Outputs

Measurement Results Documents for each SCI and BFC Groupings of the measurements are recorded in Project repository as given in Table 11.

Table 11 Deliveries for Size Estimation Process

Document No	Document Title	Created By	Delivered To
PrjNo_ReviewLog	SRS_Review	Team Leader	Prj\SWMR
PrjNo_SMRD	MeasurementResultsDocument	M&A Team	Prj\SWMR
PrjNo_SCIDMNoFS	SCIName_DetailedMeasurementResultsNoFS	M&A Team	Prj\SWMR
PrjNo_SCIDM_FS	SCIName_DetailedMeasurementResults_FS	M&A Team	Prj\SWMR
PrjNo_BFCGrpNoFS	BFCGroupingNoFS	M&A Team	Prj\SWMR
PrjNo_BFCGrpFS	BFCGroupingFS	M&A Team	Prj\SWMR
PrjNo_NewSMR	SCIs_New MeasurementResults	M&A Team	Prj\SWMR

3.5.2.5. Process Execution

For the functional size measurement COSMIC_FFP method will be applied. The measurement process is conducted according to the guidelines defined below.

3.5.2.5.1. Improvement of Measurement Quality

This step requires Application types to be used as input information.The Company should differentiate and group the software development projects according to their development environment, development procedures, development language or based on an expertise.

Team leader and Measurement &Analysis Group performs the following steps:

a. Check the Software Requirements Documents for eachSCI. A suitable way would be using a Requirements Review Checklist.

   - *Table of Contents for SRS Example is given in Table 40 Appendix 6*
   - *Example Review checklist format is given in Table 41 Appendix 6*

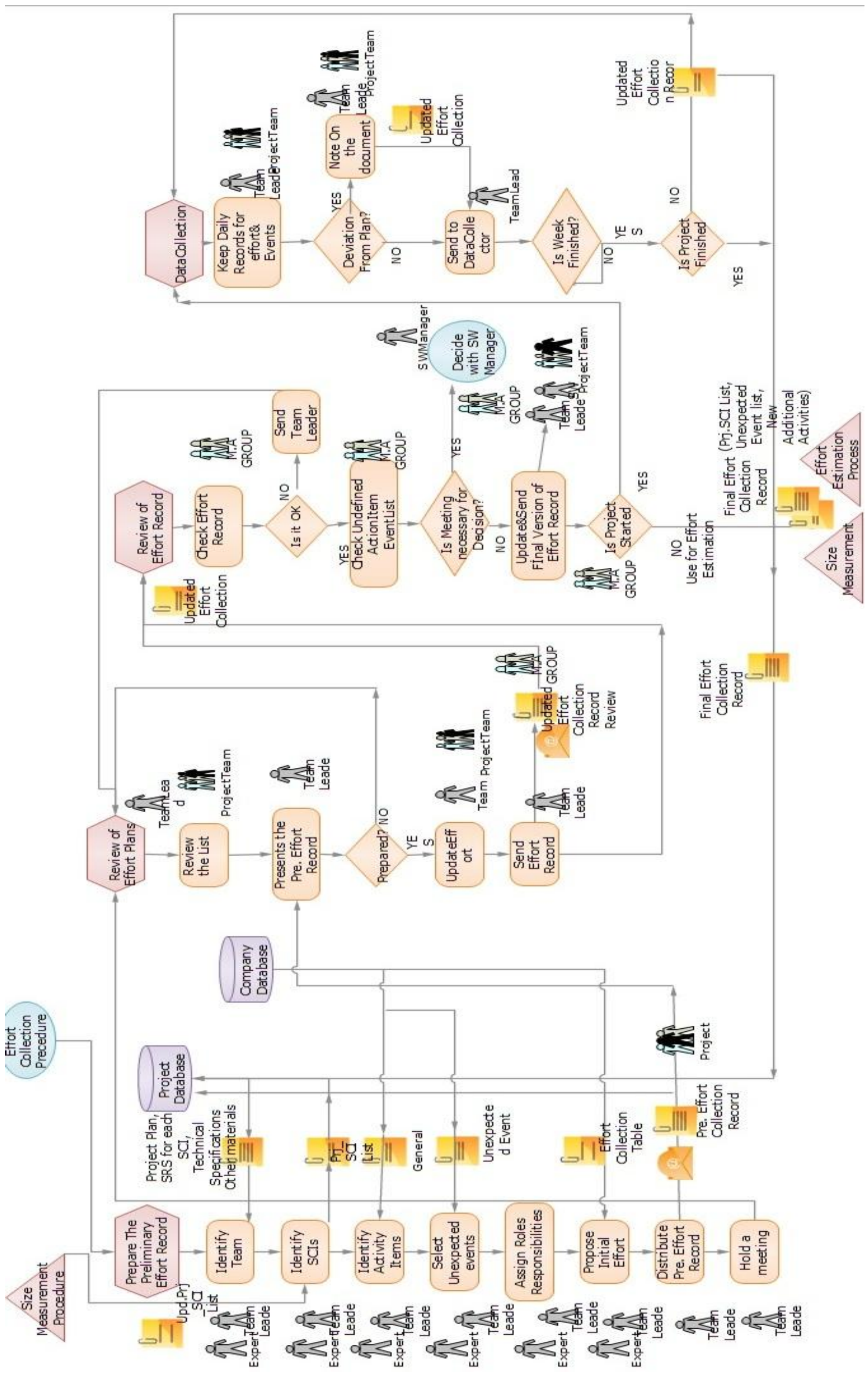
   1. Is Standard Textual format applied?
   2. Verify the comprehension of all software functions,
   3. Are triggering events, output locations reviewed? If necessary sequence diagrams are prepared or discussion sessions will be held.
   4. Missing data, ambiguities are found with reviews and document is updated etc.

b. Record the Application Domain Type of each SCI

   - *Applied categorization is given in Table 38 Appendix 5*
   - *Example categorizations are given in Table 39 Appendix5.*

c. Prepare the Measurement document. This document may include all measurement steps.

   - *Example Measurement template is given in Table 42 Appendix 6*



Size Measurement Procedure
Effort Collection Precedure
Prepare The Preliminary Effort Record
Upd.Prj SCI List
Identify Team
Identify SCIs
Identify Activity Items
Select Unexpected events
Assign Roles Responsibilities
Propose Initial Effort
Distribute Pre. Effort Record
Hold a meeting
Expert
Team Leade
Project Plan, SRS for each SCI, Technical Specifications Other materials
Prj. SCI List
General
Unexpected Event
Effort Collection Table
Pre. Effort Collection Record
Project Database
Company Database
Project
Review of Effort Plans
Review the List
Presents the Pre. Effort Record
Prepared?
NO
YES
UpdateEffort
Send Effort Record
TeamLead
ProjectTeam
Team Leade
Updated Effort Collection Record Review
M.A GROUP
Final Effort Collection Record
Review of Effort Record
Updated Effort Collection
Check Effort Record
Is it OK
Send Team Leader
Check Undefined ActionItem EventList
Is Meeting necessary for Decision?
Decide with SW Manager
SWManager
Update&Send FInal Version of Effort Record
Is Project Started
Use for Effort Estimation
Final Effort Collection Record
(Prj.SCI List, Unexpected Event list, New Additional Activities)
Size Measurement
Effort Estimation Process
DataCollection
Keep Daily Records for effort& Events
Deviation From Plan?
Note On the document
Updated Effort Collection
Send to DataCollector
Is Week Finished?
Is Project Finished
Updated Effort Collection Recor

Figure 18 Effort Collection Process

3.5.2.5.2. Identification of the Measurement Definitions

This step will be performed according to the manual of Cosmic Functional Size measurement ver3.0 [186]. So, the measurement steps are arranged according to the measurement manual. If the measurement method changes each step should be reviewed and updated.

Expert Measurer and Team Leader:

a. Identify the Layers for each SCI: Check SCI differentiation is performed accurately for the SW that will be measured.

b. Identify the System Boundary for each SCI : A conceptual interface between the functional user and the piece of software that will be measured.

c. Identify the functional users for each SCI: The types of users that send (or receive) data to (from) the functional processes of a piece of software.

d. For each SCI Identify the triggering events: An event that causes a functional user of the piece of software to initiate one or more functional processes.

- *Example Measurement template is given in Table 42 Appendix 6*

3.5.2.5.3. Mapping of the Concepts

This step will be performed according to the manual of Cosmic Functional Size measurement ver3.0. If the measurement method changes, each step should be updated.

Measurement and Analysis Group and Team Leader :

a. Identify the functional processes for each SCI: Every functional process is triggered by a data movement from the functional user, and the functional process is completed when it has executed all the data movements required for the triggering event.

b. Identify data groups and attributes for each SCI: A set of data attributes that are distinct, nonempty, nonordered, nonredundant, and that participates in a functional process.

- *Example Size Measurement is given for Cosmic inTable 43 Appendix 6*

3.5.2.5.4. Measuring the Base Functionality

This step will be performed according to the manual of Cosmic Functional Size measurement ver3.0. According to application domain tailoring is applied.

Team Leader and Expert measurer:

.1. Identify the data movements: BFCs (Entry, Exit, Read and Write) for every functional process must be identified.

.2. Tailoring is done according to the Application Domain.

.3. Record the Cosmic measurements by using the template “Design Template for Cosmic Measurement”, name the file as “SCI_Measurement Record1”, where SCI is the actual name of the software.

- *An example size measurement is given in Table 43 Appendix 6.*

3.5.2.5.5. Application of Functional Similarity

In this step functional similarity approach is applied for each functional process.

M&A Group and Team leader :

a. Check the similarity of BFCs of each functional process with other processes.

b. Apply an appropriate FS reflection method.

  - *Zero Effort method is given as example is given in Appendix 7.*

c. Update the measurements and create Measurement Results that includes Functional similarity.

  - *An example size measurement is given in Table 43 Appendix 6.*
  - *Measurement Results Template is in Table 42 Appendix 6.*

3.5.2.5.6. Validation of Measurements

In this step, another group of experts is selected to review the initial measurement results. The measurement experts may be internationally recognized or certified.

Some other people from Measurement & Analysis Group:

a. Review all the initial measurement results,

b. Check the correctness by controlling all the numerical values and computations,

c. Update and aggregate the BFCs measurements, FS Measurements for all SCIs into Measurement Results Document.

  - *Example size measurement template is given in Table 45.*

3.5.2.5.7. Recording of Measurements

In this step, M&A Group:

a. Group the measurements of BFCs for all SCIs.
   - *Grouping is given in Table 44 Appendix 6.*

b. If the project is new, that is, measurement process is used for estimation purposes of a new project, then send the BFC Grouping Results to "Effort Estimation Process" with no Effort value included,

c. If the project is not new, that is, the size measurement is performed to track the project status, then include effort collected value until that point and Publish, Record and Send Measurement Results to Project manager, SW Manager, Team Leader. These BFC grouping Measurements are used for "Data Analysis Process" .

3.5.2.6. Metrics

The following metrics are applicable to this procedure:

- Number of Updated Measurements (on daily basis) during "Validation of Measurements"
- Number of ambiquities and missing information in SRS documents found during the review.

3.5.3. Data Analysis

3.5.3.1. Purpose and Scope

To establish a process for data analysis in order to define the effort estimation model,in this procedure application of Neural network tool is defined. Therefore, the procedure explains the necessary steps for this analysis method in details with the related responsibilities. The process will be applied either to overall projects of the company, or to a specific Application Type. The recommended way is grouping the projects according to the application-domain types. This procedure also defines the application and steps of the Neural network model.

3.5.3.2. Flow Chart

Given in Figure 20.

3.5.3.3. Inputs

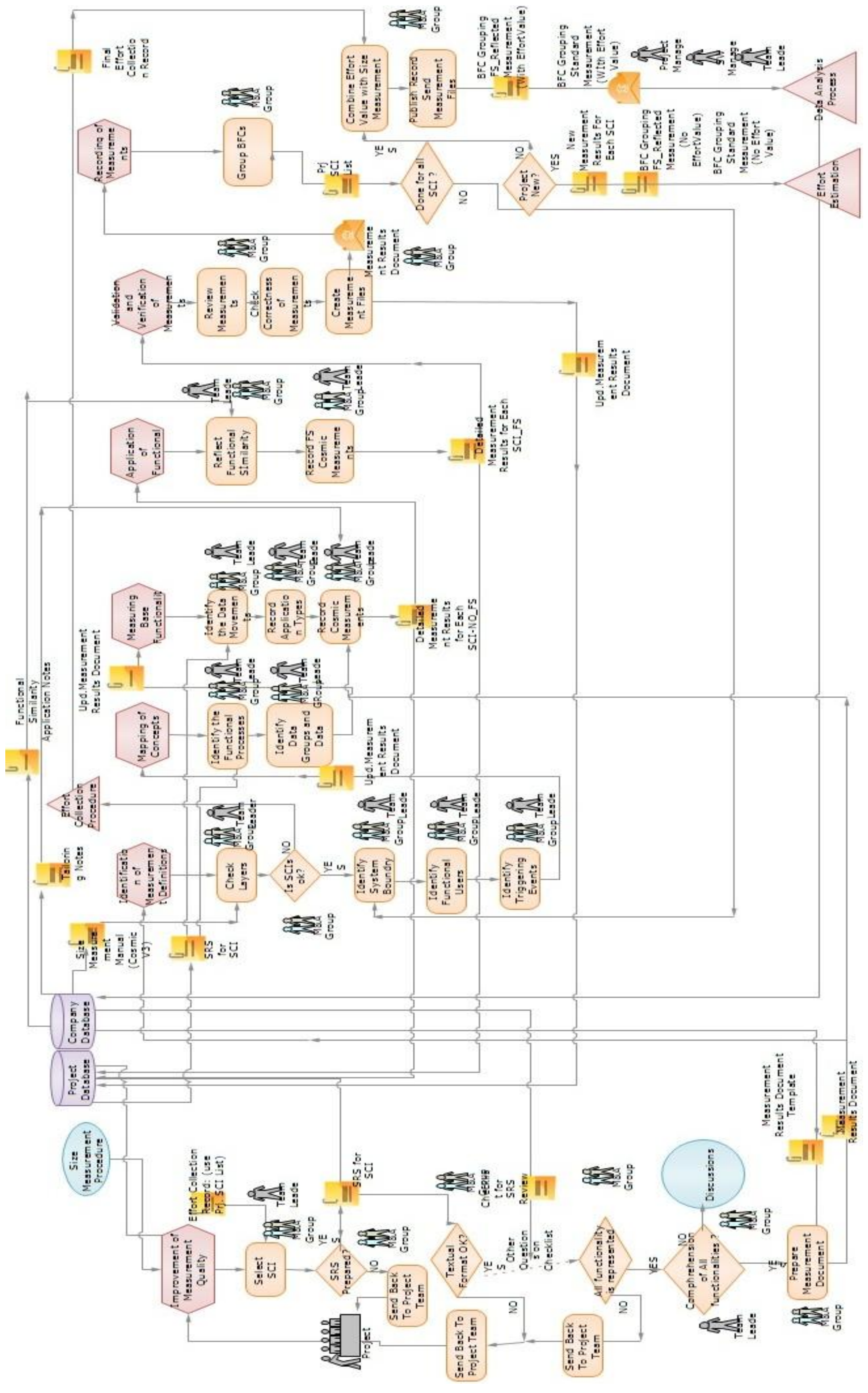
BFC Grouping Measurements with Effort Value for all projects,
Application Domain Types,
Data Analysis Tool (Ex: Neural Network Tool, Multi-Variate Regression Analysis)

3.5.3.4. Outputs

New Effort Models and Analysis results for each SCI is recorded in Company repository as given in Table 12.

Table 12 Deliveries for Data Analysis Process

Document No	Document Title	Created By	Delivered To
Effort_Model_List_vYear_X	Effort Model List	M&A Team	CompanyFolder\Model
Analysis_Results_vYear_X	Analysis Results	M&A Team	CompanyFolder\Analysis



Size Measurement Procedure
Improvement of Measurement Quality
Select SCI
SRS Prepared?
Send Back To Project Team
Textual Format OK?
All functionality is represented
Comprehension of All functionalities ?
Prepare Measurement Document
Discussions
Send Back To Project Team
Project
Project Database
Company Database
Size Measurement Manual (Cosmic V3)
SRS for SCI
Identification of Measurement Definitions
Check Layers
Is SCIs ok?
Identify System Boundary
Identify Functional Users
Identify Triggering Events
Effort Collection Procedure
Tailoring Notes
Functional Similarity Application Notes
Mapping of Concepts
Identify the Functional Processes
Identify Data Groups and Data
Upd.Measurement Results Document
Measuring Base Functionality
Identify the Data Movements
Record Application Types
Record Cosmic Measurements
Detailed Measurement Results for Each SCI-NO_FS
Application of Functional
Reflect Functional Similarity
Record FS Cosmic Measurements
Detailed Measurement Results for Each SCI_FS
Validation and Verification of Measurements
Review Measurements
Check Correctness of Measurements
Create Measurement Files
Measurement Results Document
Recording of Measurements
Group BFCs
Prj SCI List
Done for all SCI ?
Project New?
Combine Effort Value with Size Measurement
Publish Record Send Measurement Files
Final Effort Collection Record
New Measurement Results For Each SCI
BFC Grouping FS_Reflected Measurement (No EffortValue)
BFC Grouping Standard Measurement (No Effort Value)
BFC Grouping FS_Reflected Measurement (With EffortValue)
BFC Grouping Standard Measurement (With Effort Value)
Effort Estimation
Data Analysis Process
Measurement Results Document Template
Measurement Results Document
Effort Collection Record (use Prj SCI List)
Checklist for SRS Review
Other Questions on Checklist
MBA Group
Team Leader
Project Manager
YES
NO

Figure 19 Size Estimation Process

3.5.3.5. Process Execution

Data analysis is applied to overall data obtained from all software projects in the company. The process is conducted according to the guidelines defined below.

3.5.3.5.1. Preliminary Data Analysis

The M&A Team of the Company:

a. Group BFC Groupings data for all SCIs according to the Application Domains.

- *Example table is given in Table 46 Appendix 8.*

b. Calculate the Total Size for Each SCI and prepare a list with three variables; Total Size, Effort in hours, Application Type for all projects. Made a simple regression analysis to investigate outliers.

c. Arrange a meeting for the discussion of anomalies. Decide about the problematic projects: include the data into data-set, discard it for specific reason, or create a new application type.

d. Review and Update the Measurement Results File.

3.5.3.5.2. Defining Neural Network (NN) I/O

Neural Network uses a list of Input/ Output Relationship that was recorded as Measurements File. This list contains information of all projects developed in the company.

a. Identify the number of inputs for the NN model: Each type of BFCs is used as one input for the network analysis. Additional inputs may be used such as Application types, Team Size etc.

b. Identify the Output: Effort for each SCI.

3.5.3.5.3. Create The Neural Network Structure:

M&A Group should select a number of hidden neurons and layers by applying “Forward Selection Method” whose details are summarized below. The “Measurement Results” file should be presented to each neural network since it contains the input data and the desired output.

1. Start with 2 hidden neurons,
2. Train the network,

3. Evaluate the performance by calculating Mean Magnitude Relative Error (MMRE),
4. Repeat items 2-3 for a number of hidden neurons and 1 and 2 layers,
5. Compare MMRE values of the network and select the one that shows better performance.

Rule of thumbs:

- The number of hidden neurons should be in a range between the size of the input layer and the size of the output layer.
- The number of hidden neurons should be less than twice the input layer size.

3.5.3.5.4. Learning

M& A Group use Measurement Results file as an input to learning phase. Learning algorithm (such as back -propagation) is used to ‘train’ the neural network by adjusting its weights to minimize the difference between the current neural network output and the desired output.

M&A Group use Neural Network Tool and;

a. Adjust Learning Algorithm Parameters,

b. Start training the network,

c. Halt the process to check the level of learning at a defined number of iterations,

d. Test the network performance and if necessary increase the number of iterations and return step 2. Evaluate the network performance until an acceptable accuracy is achieved,

e. When an acceptable level of accuracy is obtained, the neural network is then deemed to have been trained and is ready to be utilized. (Number of İterations between 500- 1000 will generally be enough.)

   Accuracy parameters will be as follows:
   - Mean Magnitude Relative Error (MMRE)
   - Predictive Quality (PRED)
   - The coefficient of Multiple Determination (R^2)

f. Record the accuracy parameters of each iteration in “Accuracy Record” file.
   - *Example accuracy record is given in Table 47Appendix 8.*

g. Record the weighs of each neurons in “Effort Model List” file to be used for the effort model application.

- *Example weight distribution is given in Table 48 Appendix8.*

3.5.3.5.5. Analyze Effort Multipliers and Supporting & Extra Tasks

M&A Group uses a statistical tool and ;

a. Uses “Final Effort Collection Records” of all projects’ data to analyze the unexpected items effecting factors, and calculates the multiplier values in terms of percentage.

b. Uses “Final Effort Collection Records” data to analyze the effect of Action-items that are not related to development and calculates the multiplier values.

c. Prepares Analysis Results Document that includes the list of these multiplier factors and the calculated effects.

- *Example Analysis Results Template is given in Table 49Appendix8.*

3.5.3.6. Metrics

The following metrics are applicable to this procedure:

- Number of hidden neurons
- Number of outliers

3.5.4.Effort Estimation

3.5.4.1. Purpose and Scope

To establish the process for the effort estimation of a new project, it uses the results of other processes to estimate the effort.

This procedure is applied to all size of projects.

3.5.4.2. Flow Chart

Given in Figure 21.

3.5.4.3. Inputs

Final Effort Collection Record (This Record includes: Prj.SCI List, Other List,
BFC Grouping Results (Only Size Measurements, no effort data),
Analysis Results (Effort Models for each application domain)

3.5.4.4. Outputs

Estimations for new projects are recorded in company folder as given in Table 13.

Table 13 Deliveries for Effort Estimation Process

Document No	Document Title	Created By	Delivered To
Effort_Estimations_v1	Effort Estimations	Project Manager	Company Folder

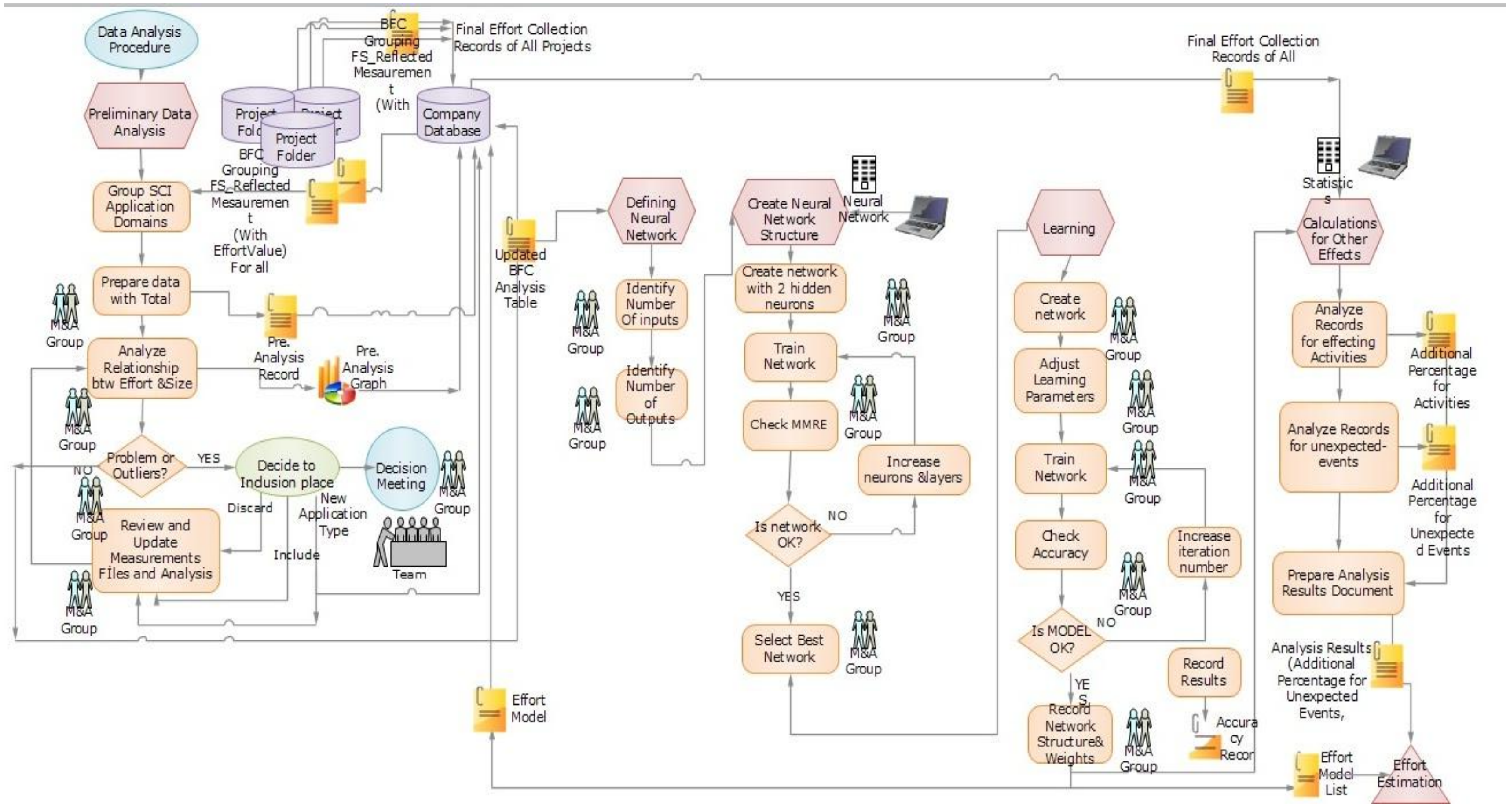

Data Analysis Procedure
Preliminary Data Analysis
Group SCI Application Domains
Prepare data with Total
Analyze Relationship btw Effort &Size
Problem or Outliers?
YES
NO
Decide to Inclusion place
Discard
Include
New Application Type
Decision Meeting
Team
Review and Update Measurements Files and Analysis
M&A Group
Project Folder
BFC Grouping FS_Reflected Mesaurement (With EffortValue) For all
BFC Grouping FS_Reflected Mesaurement (With
Final Effort Collection Records of All Projects
Company Database
Pre. Analysis Record
Pre. Analysis Graph
Updated BFC Analysis Table
Effort Model
Defining Neural Network
Identify Number Of inputs
Identify Number of Outputs
Create Neural Network Structure
Neural Network
Create network with 2 hidden neurons
Train Network
Check MMRE
Is network OK?
Increase neurons &layers
Select Best Network
Learning
Create network
Adjust Learning Parameters
Check Accuracy
Is MODEL OK?
Increase iteration number
Record Results
Accuracy Recor
Record Network Structure& Weights
Final Effort Collection Records of All
Statistics
Calculations for Other Effects
Analyze Records for effecting Activities
Additional Percentage for Activities
Analyze Records for unexpected-events
Additional Percentage for Unexpected Events
Prepare Analysis Results Document
Analysis Results (Additional Percentage for Unexpected Events,
Effort Model List
Effort Estimation

Figure 20 Data Analysis Process

3.5.4.5. Process Execution

3.5.4.5.1. Prepare the Base measurement

Project Manager and M&A Team for each SCI;

a. Use “Effort Collection Procedure” to decide Prj SCI list, Supporting & Extra effort list, for the new project.

b. Use “Size Measurement Procedure” to measure size of each SCI of the current project.

c. Get the result of “Data Analysis Procedure” i.e effort models and analysis results of Supporting & Extra efforts.

d. Use the related Effort Model generated for different Application domain types, to calculate Base effort Value for each SCI.

3.5.4.5.2. Calculate Total Effort

Project Manager and M&A Team for each SCI;

a. Use the related Analysis Results (unexpected events effect etc.) for each SCI, for the adjusted calculation of effort value for each SCI.

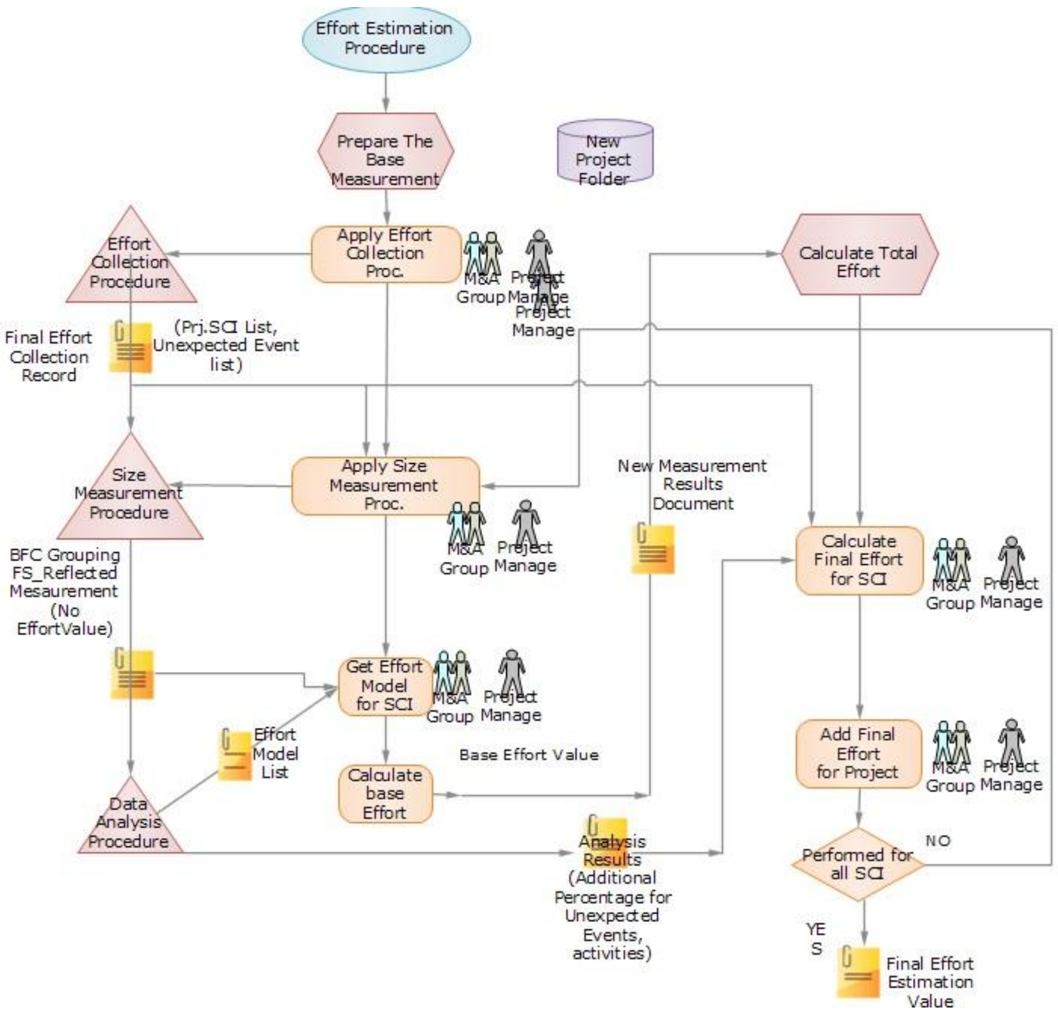
b. Perform the Effort Estimation for all SCIs to reach Final Project Effort Estimation Value.

  - *Example Table for Total effort calculation is given in Table 50 Appendix 8.*

3.5.4.6. Metrics

The following metrics are applicable to this procedure:

- Number of Multiplier Factors
- Effected Percentage of Effort



Effort Estimation Procedure
Prepare The Base Measurement
New Project Folder
Effort Collection Procedure
Apply Effort Collection Proc.
M&A Group
Project Manage
Calculate Total Effort
Final Effort Collection Record
(Prj.SCI List, Unexpected Event list)
Size Measurement Procedure
Apply Size Measurement Proc.
New Measurement Results Document
Calculate Final Effort for SCI
BFC Grouping FS_Reflected Mesaurement (No EffortValue)
Get Effort Model for SCI
Effort Model List
Calculate base Effort
Base Effort Value
Add Final Effort for Project
Data Analysis Procedure
Analysis Results (Additional Percentage for Unexpected Events, activities)
Performed for all SCI
NO
YES
Final Effort Estimation Value

Figure 21 Effort Estimation Process

3.5.5. Calibration

3.5.5.1. Purpose and Scope

To update the Effort models periodically by including new projects' information and to update the commonly used templates (General List, Supporting & Extra effort list etc.) by considering the information obtained from continuing projects, updated versions of effort models, templates and documents are put in a repository by using version control.

3.5.5.2. Flow Chart

Given in Figure 22 .

3.5.5.3. Inputs

General List, Supporting & Extra effort list,

BFC Grouping Measurements with Effort Value for all projects including new ones.

3.5.5.4. Outputs

Results of Periodic Calibration is recorded in Company folder as indicated in Table 14.

Table 14 Deliveries for Calibration Process

Document No	Document Title	Created By	Delivered To
Effort_Model_List_vYear_X	Effort Model List	M&A Team	CompanyFolder\Model
General_List_vX	General List	M&A Team	CompanyFolder\Lists
Unexpected_Event_List_vX	UnexpectedEvent list	M&A Team	CompanyFolder\Lists

3.5.5.5. Process Execution

3.5.5.5.1. Review The Lists

M&A Group :

a. Reviews each project recordings every 2 months with SW Manager and publishes updates for general list and unexpected event list to all Team Leaders in the company.

3.5.5.5.2. Calibration of Effort Models

M&A Group :

a. Apply the Data Analysis procedure with the frequency of 3 months. Calibration of the model is necessary to include the last projects' data.

3.5.5.6. Metrics

The following metrics are applicable to this procedure:

- Number of Supporting & Extra effort types encountered.

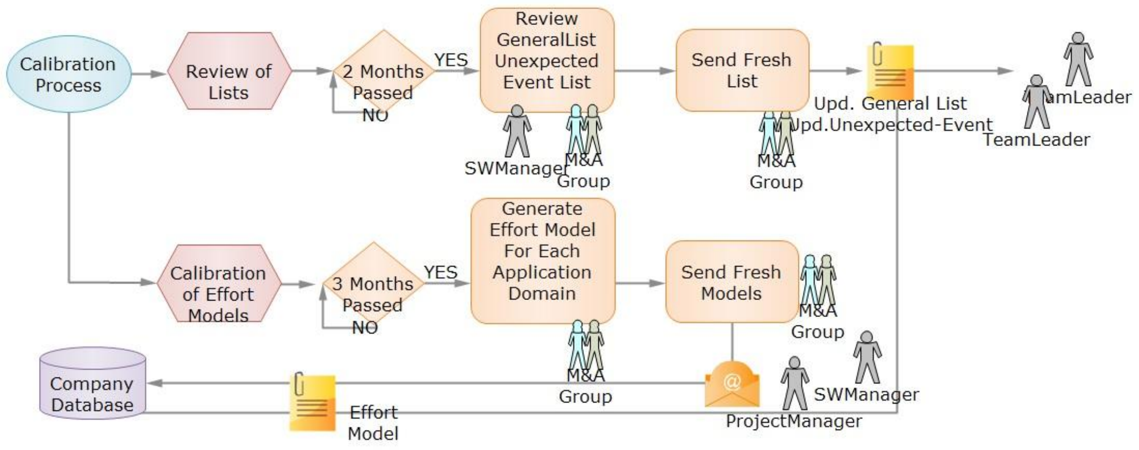


Figure 22 Calibration Process

CHAPTER 4

4. EXPLORATION and APPLICATION OF THE METHOD

This chapter describes utilization of a multiple-case study to develop and validate the Effort Estimation Methodology. Two multiple case studies were conducted. In the first multiple case study we aimed to identify the requirements to develop a methodology for effort estimation and investigated improvement opportunities in Functional Size Measurement and Data Analysis methods. In the second case study our aim was to validate the methodology. Section 4.1 explains the research strategy followed and Section 4.2 describes the plans for case studies. In sub sections of 4.3 case studies are explained. Lessons learned and results will be given in 4.5.

4.1. Research Methodology

In this thesis study, we investigated how we can build an improved size and effort relationship model and how we can apply it in organizations.

For the investigation of this phenomenon, we used a variety of evidences from different sources, such as documents, artifacts and observations. However, during this examination, we had no control over the behavioral events. Thus, we used the case study as our research method.

We performed two case-studies. The first case study was an Embedded type unit, because we aimed to analyze multiple improvement opportunities in building a relationship and in data collection method. It was also a multiple case study, after a first pilot investigation in improvements that was applied on a very small dataset, we checked these opportunities at a different time and for another larger set of projects.

The second case study was a single case study to confirm the methodology. It represented a unique case. As Yin suggested [162], it is holistic, because it doesn't contain multiple analysis units. Next section expands the design of our multiple-case study. It draws the questions of the study, the propositions, and data collection and analysis strategies.

4.2. Multiple Case Study Design

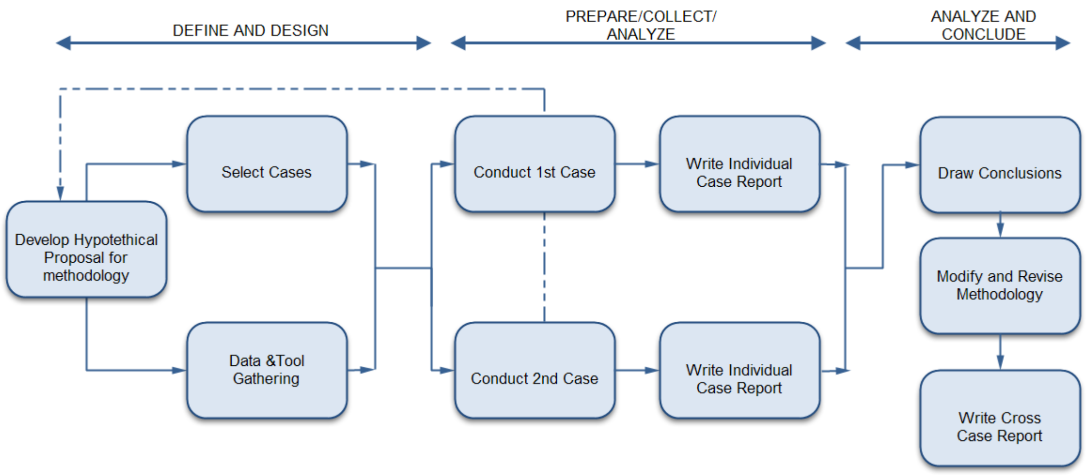


Figure 23 Case Study Flow

4.2.1. Define and Design of Case Study

4.2.1.1. Hypothetical propositions for methodology

We investigated the following research questions :

RQ1 How can we build an effective size based effort estimation methodology using the available data?

RQ2 What are the factors that might improve the effectiveness of methodology?

RQ3 Can we utilize the methodology to build a size and effort relationship?

4.2.1.2. Case Study Plan

In order to answer our hypothetical propositions above we, as researchers, conducted two case studies in a middle sized company. Detailed plan of the research is given in Table 15.

Table 15 Case Study Plan

	Action	Approach	Resource&Tools	Role
Preliminary Work	Selection of Projects	Investigate documentation maturity of projects. Evaluate the contents of the available collected effort data. Select projects with detailed collected information for each task.	SRS , SDD Documents, Other technical documents	Researchers, Team Leaders
	Data & Tool Selection	Select Analysis Tools	Tools for Single Regression Analysis Multivarite Regression Analysis, ANN Analysis,Excell	Researchers
C A S E 1	Conduction of case1-Part1	Search literature on FS Consideration Measure Cosmic Sizes of projects Check structure and recordings on Effort values Check FS effect on relationship of size and effort. Write Report for this sub-part	Cosmic Standard v3.0 Effort recordings	Researchers, Reviewers
	Conduction of case-1-Part2	Evaluate FS Consideration on different Application domains Evaluate FS effect on Project Phases and Tasks Write Report for this sub part		Researchers, Reviewers
	Conduction of case-1-Part 3	Generation of General Models Generation of Specific Models (Single, Multivariate, ANN) Comparison of All models by using Accuracy parameters. Evaluate collection processes, its problems, designate solutions Develop methodology including processes and related assets Write Report for this sub-part	Tools for Single Regression Analysis Multivarite Regression Analysis, ANN Analysis,Excell	Researchers,
Case Study -2	Conduction of Case study 2	Apply Effort Collection Process Apply Size Measurement Process Apply Data Analysis Process Evaluate the benefits of methodology	Tools for Single Regression Analysis MultivariteRegressionAnalysis, ANN Analysis,Excell	Researchers,

In this company after justifying the rationale for some improvements by conducting the Case-1, in Case-2, we validated the "EFES methodology".

The steps of our study are summarized and reports on our findings are presented in Table 15. These studies are applied for the decisions of improvements in our methodology empirically. In following paragraphs all case studies are performed by researchers. The required information is provided by team leaders and managers when necessary. If company applies EFES methodology, it can directly use FS and BFC concepts. So there is no need for company to repeat these steps. In real execution of methodology only the Actors given in Table 9, Part 3.4 will take part.

Roles and Responsibilities :

Researchers : Researchers and Data Analyst of the methodology who are experts on the COSMIC method measurement.

Team Leaders - Software Design Leaders of the Projects from the company.

Data Collectors - Responsible People for the Data Collection in the company.

SW Manager - Owner of the Collected Data and processes.

Measurement Group: Certified on the COSMIC method measurement and expert on Functional Size Measurements.

Data Collectors and Measurement Group will form a "Measurement and Analysis group" for controlling all processes.

4.2.1.3. Case Selection:

Selection of Projects: For the selection of the projects in this company we identified three criteria to be fulfilled.

1. In order to measure functional sizes of the cases, we required a well defined software requirements specification documents, and software design documents for the project.

2. Each project should have its effort data in detail. By “detailed” we mean, both project related development activities and other types of effort such as efforts for demonstrations, meetings, hardware problems, etc exist in effort record.

3. Projects from several application domains should be included in Case-Study provided repository.

For the 3rd criteria the_application categorization of the company is investigated which is given in Table 37. Their interrelation as layers are shown in Figure 24 . Each block in that figure is a SCI.

GUI applications for the Data Driven Control Systems, (GUI): These SCIs are simulators of some existing products. They are used to generate and send an artificial data in order to test another system or are used to show the results of an externally connected system to check the accuracy of the data. They have one or more data-interfaces to be connected to other systems and GUIs for user to change or create a new data, or presents the externally generated results.

Real-time Embedded Application Software(ES) : These SCIs are developed using RTOS development environments and also include the communication and control software, algorithm processing software etc. In our case study algorithm processing software SCIs are not included

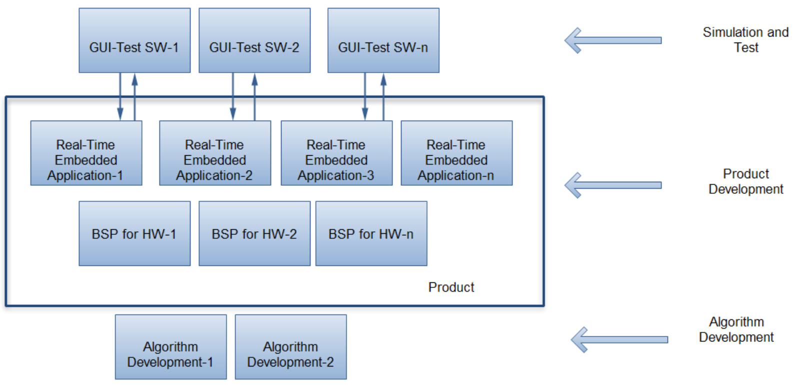


Figure 24 Categories of software in company in layered approach

Board Support Package applications for Hardware Support, (BSP): These SCIs are embedded device drivers developed for specifically designed hardware. All existing subcomponents of hardware are controlled with the aid of this type of software.

Algorithm Development: These applications are not put in the product. They are developed externally by using MATLAB tools and coded by using “ES” type Applications. Neither the algorithm related development nor the related ES are considered in our study. Because, there are still discussions for the functional size evaluation of these type of applications.

The evaluation of these projects is performed by applying the above items. By considering the item 3 and investigating the projects, it was decided that, company has finished projects grouped under three application domains: GUI, ES, BSP.

The evaluation of cases was performed in two cycles. In the first run, observations showed that there were many GUI projects completed, but most of the ES projects that had recorded effort values were still under development. Therefore for the first case study, to evaluate the methodology requirements, only finished projects were applied. In the second run, for the case study-2, newly finished projects were included in the overall verification of the methodology.

At the beginning of the research there were: 14 ES, 16 GUI projects and 8 BSP projects.

For case study-1:

According to item 1 above, projects are required to be completed. Researchers ended up with 6 ES projects, 13 GUI projects and 6 BSP projects. The remaining were still under development. Among the finished projects 1 GUI and 1 ES project didn’t have much detail in the effort recordings so it was eliminated . Researchers learned that 2 ES project has changed the development environment during design phase and started the development from the scratch. For these projects, effort recordings were including the previous effort, so Researchers ignored these projects. 1 GUI and 2 BSP projects did not involve other types of effort spent such as HW problems, maintenance activities of other projects etc. The responsible team leaders commented that, “such information was considered as a part of development and not differentiated”. Therefore these projects were also eliminated.

For the case-study 2,

Recently finished projects for each application domain were: 8 new GUI, 12 new ES and 8 new BSP project. One of the BSP projects has been integrated to two different HW environments which requires some parts of the functionalities of the software to be updated or renewed. Efforts for such modifications were recorded in a combined way. So Researchers excluded this project from our database. 1 ES and 1 GUI applications SRS and SDD documents were not mature enough so they were removed from our analysis. 1 GUI application was restarted by using another environment with the request of customer that prevented to obtain real effort values in recordings.

So final project numbers are as follows:

Case Study 1: 11 GUI , 3 ES and 4 BSP projects

Case Study 2: (6 GUI, 9 ES and 7 BSP)

As a total: 17 GUI, 12 ES and 11 BSP

4.3. Case Studies

4.3.1. Case Study-1.

Researchers have conducted the first case study to find answers for the following questions connected with our hypothesis:

For RQ 3: We, as researchers investigated the following questions in Part 1 and Part 2:

- “Is functional similarity inclusion improves the correlation between size and effort? Whether its effect varies based on the application domain, or for some specific phases.

- "Does considering efforts for specific tasks or events (i.e effort wasted for HW problems, unexpected effort, effort necessary for safety processes etc.) enable better size effort relation ?

For RQ 1 and RQ 3: We investigated the following question in Part 3:

- "Does considering functional components (BFCs) in effort model enable better effort estimation accuracy?"

For RQ 2: We investigated the following question in Part 2:

- What kind of problems may occur in collecting and using the effort data? Can we improve effort collection process and related data such that it would be useful for future project estimations?

4.3.1.1. Case Study 1-Part-1 Prepare and Collect Data

Aim: To observe if functional similarity would improve the correlation between size and effort.

The three GUI projects include the development of different simulators which have been nominated to the embedded platforms were selected from organizational database. They all have a Mil-std 1553 communication interface which enables user to connect to a terminal side where data are taken from and send to. Simulators have a user friendly GUI which shows and controls all sent and received messages. With respect to CHAR Method defined in [21], the functional domain of this three simulation products is accepted as "Complex Data Driven Control System". This application domain has an apparent visually similar component that enables us to investigate the similarity reflection in effort model.

Two separate teams each consisting of one Researcher who was an expert on the COSMIC method, measured the case products independently using software requirements specification (SRS) documents. SRS documents of each case were conformant with the IEEE Standard 830-1998 [127] . The measurement results were verified by a different team members, that was Reviewer in measurement group, who was not involved in the measurement process of the verified project.

4.3.1.2. Case Study 1-Part-1: Conduction

4.3.1.2.1. Implementation of the method defined by Santillo and Abran:

For the evaluation of the FS the method proposed by Santillo and Abran [79] is used. This method has two stages. The first one which is called as "the first order evaluation" compares the functional processes only from data movements' point of view. Similarity among functional processes are determined by comparing the data group and data movement relationships; in addition to this, in some cases where the comparison technique does not suffice , it is suggested that the analyst makes her best judgments in order to identify the functional similarities. The second one "second order evaluation" determines the functional similarities by considering both data movement and data manipulation action types.

Researchers applied the second order evaluation to our three projects. For all these projects, the first step was to determine the Cosmic function point size (CFP) of the products by using the COSMIC method. The results of the measurements in terms of BFC components are given in Table 16.

After this step, detailed size measurement data sets were arranged to uncover the amount of functional similarities. This step was aimed to compare the data group and data movement couples within different functional processes, which required comparing all the couples with each other. As Santillo and Abran [79] emphasized, the average of the functional similarities can be calculated in order to be able to make a judgment about the reuse capacity of the product.

Table 16 Measurement Results of the Projects

Case Project	No of Functional Processes	No of Entries	No of Exits	No of Reads	No of Writes	Cosmic Function Point Size (CFP)
GUI-Project1	53	107	98	39	33	277
GUI-Project2	44	73	33	24	63	193
GUI-Project3	11	82	16	4	17	129

So, both the first order and the second order evaluation approaches have been applied to GUI Projects. Evaluation results are given on Table 17. It seems that GUI projects contains multiple similar processes, especially if we only consider first order evaluation. During the second order evaluation we came to the conclusion that there may be other kinds of data manipulations apart from the ones given in Santillo and Abran's research. These other type of manipulations may depend on the application types to provide appropriate granularity. For instance, it was realized that even if two of the calculation data manipulations within the IS-Project were different, they were considered similar because of the current data manipulation types. However, for this situation the complexities behind the calculation processes should be taken into account.

Actually a closer study of the FURs provided a clear insight to the analyst about the similarities among functional processes.

4.3.1.2.2. Productivity comparisons:

Researchers investigated if the FS results could be used for the correlation of the functional size and the development effort. Since the GUI projects were developed under the same conditions by the same team, it was assumed that the productivity values were the same and could therefore be compared. For the evaluation of productivities Researchers compared the effort values for the whole development lifecycle. In Table 17 , initial productivity values represent the productivities in which the functional similarities were not taken into account.

Table 17 The results of first order and second order evaluations

Case Study	**# of Functional Processes**	**Size of the Product (CFP)**	**Avg. Functional Reuse Percentage (First Order)**	**Avg. Functional Reuse Percentage (Second Order)**
GUI-Project1	53	277	79,0%	52,41 %
GUI-Project2	44	193	57,75 %	34,37 %
GUI-Project3	11	129	25,8%	22.36 %

Table 18 Productivity Rates of Simulator Projects (Productivity= Size Cfsu/ effort (man-day))

Project	Functional Size (CFP)	Initial Productivity	Productivity Results After Similarity Considered
GUI-Project1	277	9,8913	4,24
GUI-Project2	193	6,89	3,39
GUI-Project3	129	4,44	3,19

4.3.1.2.3. Functional Similarity Evaluation

When we, as researchers, evaluated the initial productivity results of the projects, we observed that the effort required to complete these projects did not depend on the sizes of the projects since productivity values were so different. Moreover, although the size of the GUI-Project1 was greater than the other simulator projects, it had been finished in less time, with a higher productivity rate.

When we measured the size only from the user's point of view and try to predict the effort by looking at the historical measure we attained three different effort values. In a sense even a new product would have been required to be developed by the same team, it would be very difficult to estimate the effort required based on the historical data.

To attain more reliable productivity values, we counted GUI Projects' size again by considering functional similarities. For two functional processes, if they were found similar, we counted one instance, and assumed the other one as an enhancement of the first one. So we assumed that the effort needed to develop for the second instance was negligible. Therefore during the size measurement we ignored the effort required for the modified part and reached new productivity results given in 3rd column on Table 18. Results for the evaluation of Functional similarity is given in following part.

4.3.1.3. Case Study 1 Part 1: Results

When the functional similarities were not considered , distinct productivity values had been obtained for the same team. On the other hand when the functional similarities were taken into account productivity values were closer to each other. This revealed that for representative productivity values, FS should be reflected in size measurements. These new productivity results reflected the real productivity of the team better since the values were closer to each other than the values in the 3rd column.

4.3.1.4. Case Study-1 Part-2: Prepare and Collect Data:

Aim: We, as researchers, have two aims in this part: The first aim is to check whether effort and size relationship is affected by functional similarity, by expanding the application domains.

Our second aim is : To investigate effort collection strategies, problems and improvements for developing effort estimation methodology.

We also looked into the effect of FS on different phases or tasks of life-cycle, and investigated the improvement opportunities in Data Collection Method.

We selected large numbers of projects from organizational database for this part. Since we also aimed at the inquirement of this information for different application domains, we expanded our domain selection compared to first part, and included both BSP projects and ES projects to case study. Total number of projects under analysis became 18.

The first step for this part was to determine the functional size of the projects by using the COSMIC method without considering similarity issues. Two software engineers performed the size measurement for all the projects. One engineer was an author of this paper and the other was responsible for the measurement processes in the organization and leads the development of projects. For the validity of results another team, certified to apply the COSMIC method, sampled seven of the projects to check the measurement results.

4.3.1.5. Case Study -1 Part-2 : Conduct

By using a basic formula (Productivity= Size / Effort) and without using any regression techniques we found that productivity values of the projects for similar teams have large variances. Productivity value changed between 30,6 and 0,35. The results can be seen in Table 19.Then in order to determine the reuse percentage level of the projects Santillo and Abran’s Method first order evaluation was used as it was evaluated in part 1. The reuse percentages of the projects are also given in Table 19.

Table 19 Productivity Rates of Projects (Productivity= Cfsu/ man-day)

	Size(CFP)	**Effort(day)**	**Initial Prod.**	**% Reuse**	**Size-After FS Reflected**	**After FS Prod.**
GUI-1	588	127,5	4,61	33	364	2,85
GUI-2	1384	49,5	27,9	88	208	4,2
GUI-3	412	59,5	6,92	11	374	6,28
GUI-4	1438	47	30,6	81	213	4,53
GUI-5	129	29	4,44	25	93	3,2
GUI-6	76	31	2,45	14	59	1,93
GUI-7	986	64,5	15,29	56	423	6,55
GUI-8	1347	104	12,95	83	396	3,8
GUI-9	136	12	11,3	86	13	1,08
GUI-10	277	28	9,89	79	119	4,24
GUI-11	193	28	6,89	57	95	3,39
BSP-1	419	98	4,27	10	402	4,1
BSP-2	1740	471	3,69	6	1620	3,43
BSP-3	660	253	2,6	0	660	2,6
BSP-4	421	133	3,16	0	421	3,16
ES-1	2040	336	6,07	16	1634	4,86
ES-2	246	696	0,35	6	194	0,27
ES-3	362	139	2,6	4	311	2,23
AVE.			8,66			3,48
AVE. Deviation			6,22			1,19
VAR			87,01			2,69

For the FS calculations, we thought that if a newly developed functionality was similar to the previous one, the effort required for this modification would not be so high and could be assumed negligible. Therefore for the sake of simplicity in our calculations, we assumed “zero” effort for these new functional units. We calculated new Cosmic Functional Size Units based on this condition and recalculated productivity values. These new FS reflected size is given in Table 19. Basic Productivity calculations using the FS reflected size is included in the same Table for comparison.

Since our second aim was to observe the effect of FS on different application domains, we analyzed the results of 3 applications domains. In Table 19 average productivity rates and average deviation of productivity for different application domains are given.

As a third question we investigated the effect of FS on different project tasks. When we compared the reuse percentages of the application domains in Table 19, we found that BSP and ES projects had lower values of similarity. Therefore, to observe how increased percentage of similarity value impact effort in any specific phases, we selected only GUI applications for further analysis.

In Figure 25 , we presented how effort per unit functional size changes with functional similarity percentage of the project. Functional similarities are decided using requirement phases’ results, therefore efforts on man/-day basis of other phases are considered for this observation. For calculation of the unit effort initial functionality size is used, since similarity issue is considered in other axes of the graph.

Table 20 Average productivity rates and average deviation of productivity

	No Func.similarity			**Func. Sim. Included**		
App.Type	Avg. Prod.	Avg.Dev. Prod.	Variance	Avg. Prod.	Avg.Dev. Prod.	Variance
GUI	12,11	6,96	87,01	3,82	1,21	2,69
BSP	3,43	0,55	0,51	3,32	0,44	0,39
ES	3,00	2,04	8,30	2,45	1,60	5,30

4.3.1.5.1. Evaluation of Effort Data Collection Process of Company:

For the methodology development we investigated the problems and the improvement opportunities in Data Collection Mechanism. Therefore we looked into the database of different projects and effort collection method of the company.

In this company, at the beginning of each project, the Team Leader lists the required activities on a standard excel sheet with his/her specific estimates of these items. Then, this table is reviewed by the software team in the same project to include the required unseen items and for better predictions. The software action items are listed such that, the minimum amount of time for performing an item will be 1 day (eight hours) with the maximum allowable time being two to three weeks. Every day, each developer inserts his/her effort and related work package into this table. Every week, these values are checked by the software team then are sent to the data collectors. However, in order to collect the real effort, unexpected situations are also noted with this tool, such as 'unavailability of developer', 'hardware problems', 'specific procedure appliance' and 'unexpected meetings'. Any event that prevents the developer from proceeding can be added to the list. Although the main aim is to collect the real effort, this tool allows the developer to plan and monitor his/her work packages.

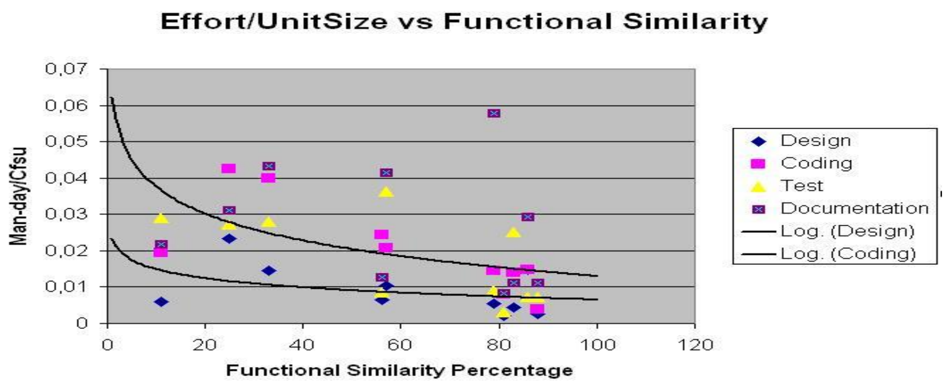


Figure 25 Effort/cfsu variation w.r.t. FS in different phases

It was noted that despite the resolution of the effort was defined as eight hours, the developers counted their effort on the basis of four hours. We realized that some type of efforts requires considerable

Table 21 Findings and Requirements for Methodology

Status	Finding	Requirement
Positive	Each Development Team define their activities and phases themselves, and collects data rigorously,	Teams should define their own activities by taking responsibility.
Positive	Most of the teams prepare very detailed activity lists,	Activity definitions should be reviewed by Project Team together to check whether all items are included.
Positive	Each member take responsibility and control their own activities,	Inlusion of remaining effort will enable the methodology to be used for monitoring.
Positive	Each development team performed SCI decomposition well enough.	SCI decomposition Review meeting should be included.
Negative	Activity definitions and related phases are not very interpretable. Some discussions were needed to understand the activities performed.	A predefined activity and Task list is necessary.
Negative	It is not easy for some activities to be put under certain phase of the project. Even some definitions include a new SCI development.	SCIs should be reviewed. Phases should be defined in parallel to a known standard.(i.e 12207)
Negative	While some projects add information for items (Like demonstrations) as an activity to be performed during development, some other projects assume that the mentioned effort was wasted and prevented to proceed.	Collect Supporting and extra work separately, Create a template for data collection for those items.
Negative	While some projects counts time spent for requirement changes separately, some don't care this situation and includes this extra effort as a part of design effort.	Template should include defined works. If new item appears "expert review" to include, then distribute.
Negative	A recording for "infrastructure work" item exist. However some team leaders create a new very detailed activity list instead of using it. In this list they also include algorithm development works, hw problems (even there exist an item for this), a kind of simulator developments (which is another SCI) etc.	Template should include defined works. Review points should exist for the decision of contradictory opinions
Negative	A few projects collect data montly, while most of them weekly.	Weekly collection is recommended in literature .Data collectors should review data availability.
From Empirical Study	Requirements, Design and Coding and Testing phases are effected by Functional Similarity. Documentation task is not a function of functional Similarity	For Development Effort: Include the effected phases Percentage of effort for Documentation should be evaluated. It is not included in base effort modeling.

amount of extra effort. For example, by looking at the Table 19 below we can say that for HW and ES Applications for this company, a large amount of effort is required for safety related issues. GUI applications on the other hand, utilize major effort for requirements changes.

Development of Artifacts for Data Collection during the investigation of database, unavailability of some records dropped our number of SCIs. Besides, we encountered some positive and negative issues. These issues also form the requirements of our methodology. The findings, and our approaches are given in the next column:

Controversial assumptions create problems in data analysis. Therefore, for methodology we need a pre-specified effort data structure, that will be arranged according to a world known standard. For this reason, based on the literature findings and above results we developed data collection templates, effort collection lists for this company. The following structures have been formed:

Data Collection:

- Standard Effort Collection Record List (Table 31, Table 32): To prevent ambiguity on concepts, tasks, activities.
- Standard Categorization list of Software (Table 38) : After discussing with team leaders formed for the company.
- Effort Collection Table Template (Phase, Activity, Developer name, Real Effort, Completeness of task) (Table 37) : Both to record the past efforts, and also to track the remaining effort.

Size Measurement:

- SRS Document review template (Table 41): To evaluate the maturity of the SRS documents.
- Size Measurement Results Template (Table 43, Table 44): To record the information on measurement problems, assumptions etc. to record the BFC measurements.
- Size Measurement Groupings based on BFC for all projects: To combine the available data of all projects for further analysis. (Table 46)

Data Analysis:

- Effort Estimation Model Weights template (Table 48): To record the developed effort model.
- Supporting &Extra Effort analysis template (Table 49): To record the percentage of effort affecting factors, tasks, activities like demonstrations, trainings etc.
- Analysis Table Templates (Table 46,Table 47)
- Analysis Results (Table 49)

Effort Estimation:

- Estimation Table Template for new project: (Table 50)

4.3.1.6. Case Study-1 Part 2 : Results

In our case study we focused on the structure related problems of software sizing namely functional similarity. As it can be seen from Table 19 and Table 20, for GUI projects, by considering FS we can achieve a better size and effort correlation. In other words we were able to obtain acceptable average productivity values with this approach. Variance of the productivity value of GUI projects became reasonable. However for the BSP and ES type of applications there is not much change. Their similarity percentages are lower. Although for ES applications variance has decreased, we still cannot reach an agreeable productivity value. For these types of applications it seems that the problem cannot be solved with the Functional reusability approach we have utilized. We hypothesized that for the effort estimation of these types of applications, size should not be the only driving factor. We saw that; although the number of projects was limited to decide on, HW related projects had comparable number of average productivity values with each other and whenever a new project started, a rate might be agreed on for a better estimation.

Table 22 Percentages of special effort/total project effort.

Application Type	%Req Change Effort	% HW Problems	%Safety Related Effort
GUI	19,6	1,2	0
BSP	0,7	16,07	21
ES	0,35	1,4	19,1

Table 22 presents that, safety related effort is found high for ES and BSP applications and requirement changes require extra effort for GUI applications before the delivery of the products. Therefore specialized part of the effort seems to be expended for unplanned situations. For more accurate effort estimation these parts should be investigated.

We found that for GUI applications, design and coding phases were affected from the similar functionalities of the project. Even for a small number of similarities there exists an effort reduction. However it seemed that, for the test phase, effort per unit of software size did not decrease much, but it is still affected. Moreover, we did not find any relationship between similarity percentage and documentation phase's effort. Although all application types were affected by similarity, FS was a major concern for GUI type of applications. However in terms of effort data recordings, we saw that there was no standardized way to perform further analysis.

Even the data collection process was executed appropriately and every effort wasted was recorded, the definitions of phases and activities were contradictory in some cases.

Methodology should include a predefined phase and activities terminology to prevent misunderstandings, contradictions. Besides to control the available data, a mechanism should be included such that, if no activities inserted.

4.3.1.7. Case Study-1- Part 3

Aim: To Investigate the effect of using BFC components in effort modeling.

To guide software designers and organizations in their endeavor to identify the best fitting estimation models,

As researchers, we performed an exploratory study that investigates some of the concepts of the functional size based effort estimation models. First, we examined the estimation accuracies of previously proposed functional size based estimation models for a specific company. Then, we investigated the usage of ANN and multi-input regression analysis to build an effort model that uses the COSMIC method, and finally we explored the contribution of functional similarity concept to effort estimation models.

4.3.1.8. Case Study-1 Part 3: Prepare and Collect Data

The same functional size of 18 projects in Case-1-Part2 was used.

4.3.1.9. Case Study-1 Part 3: Conduction

Conduction of this part consisted of two steps:

[1] Selection of functional size based effort models,
[2] Application of generic and specific models of effort estimation.

4.3.1.9.1. Selection of functional size based effort models

In our study we categorized the effort estimation models that are based on functional size as being generic and specific. We used the term ‘generic models’ terminology for models that are proposed in the literature or are used in software community. Specific models have been retrospectively established for the company where we conducted our case study.

We searched the literature to find suitable functional size based models. The selected models, the reason for selection, number of samples used for creation, and applied FP method are given in Table 23. The first three models are the most well-known effort estimation models that are used in comparison studies [3][65][66].

Generic models commonly in use are defined in linear or power forms. In another comparison study, Abran [129] developed various non-linear estimation models for the selection of an appropriate single input regression model. While constructing our models Abran’s equations were taken into account.

The selected single input regression models for specific effort models are given in rows 1-4 in Table 23. For regression analysis the Least Squares Regression was applied in order to provide the best fit for all the models.

The measured size of projects consists of several 'component types' which have a different effect on development effort. In many functional sizing methods that have been defined, the types of the components of functionality are essentially similar, but are given a different name. For example, the COSMIC Functional size has four types of BFCs; ("Entry", "Exit", "Read", "Write"). In our study the method of aggregating the COSMIC components was investigated and compared in order to find the best fit model equation. To generate multi-variate models, components were used as independent parameters of the models and the ANN and multi-variate Regression methods were applied. The multi variate models that were selected from literature are given in rows 6 and 7 in Table 24.

In order to construct the ANN models, a number of BFCs were used as inputs for the model and a number of neurons were selected. Therefore, the independent variables of the ANN model were: # of E, # of X, # of R and # of W. The dependent variable of the model was the calculated effort. Two different ANN models were built to predict the required output. The first model used one hidden layer, and the second one had two hidden layers. In order to find functionally similar parts the 'Requirements Definition Documents' of the company and the FS definition from Santillo and Abran [79] were used.

4.3.1.9.2. Application of Generic and Specific Models of Effort Estimation

4.3.1.9.3. Size and effort relationship

The first investigation for the relationship between total functional size and effort is given in Figure 26 . One outlier was found out of the confidence interval. After analysis it was discovered that this was the first sub-project that had been developed using the safety critical approach. Therefore, the simulation, verification and test effort were compared very high to the other projects under the 'Real-time Embedded System Application Software' classification. Furthermore, it was found that extra tailoring procedures had been prepared for this type of application. As a result, we excluded this data from the data set on the basis that safety critical applications should be grouped under another classification, even if they were developed by the same team using the same tools.

4.3.1.9.4. Utilization of generic models

The original generic effort models utilized the IFPUG/FPA methods as shown in Table 23. However, in this current research it was planned to produce the effort models using the COSMIC measurement method. Therefore, in order to create a good comparison base, the COSMIC Size was converted to utilize size before applying the estimation model using the models of Fetcke [22], and Vogezelang [23].

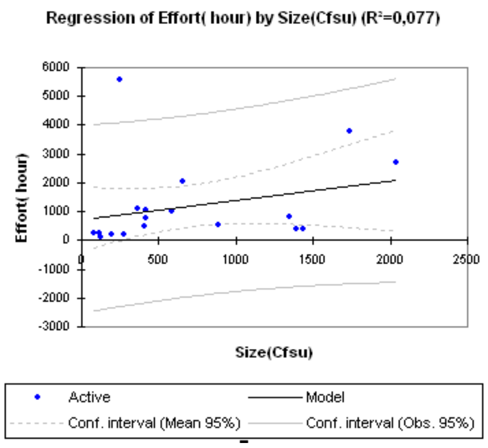


Figure 26 Size and Effort Relationship for the Dataset

In this study, all the generic models have fixed multipliers except the Cobb-Douglass model [67] which allows the calculation of each parameter in the model according to the dataset and measurement method. From this point of view, it can also be assumed to be a specific model. In our research, the FS concept was not applicable to generic models since the functional processes of the projects were not provided for analysis in the previous studies.

4.3.1.9.5. Building specific regression models

For the construction of regression models, Ordinary Least Squares Regression (OLS) analysis was selected. This is the most commonly used method for developing software estimation models [24][25][26] since it is simple and available in most statistical software packages. This analysis allows data to be fitted into a pre-specified model. In order to provide the best fitting equation the overall sum of squared estimation errors is minimized using the least square regression equation (LSR).

Table 23 Selected generic effort models

Model	***Explanation***	***Reason for Selection***	***Definition***	***# of Smp.***	***FP Method***
Albrecht-Gaffney [3]	Suggested by the founders of FP method.	Accepted as first model based on software functionality	E(man-mnt) =0.0545*FP–13.39	24	FPA
Kemerer [65]	Argued that Albrecht model is not appropriate.	Compares their results with Albrecht -Gaffney Model, suggested a curved one.	E(man-months) $=60.62*7.728*10^{-8}*FPA^{3}$	15	FPA
Matson [66]	Argued that the two models above have limitations.	Follow up study, compares two models above, and his new model	E(work-hours) =585.7+15.12*FPA	104	FPA
Cobb-Douglass [67]	A production function that takes into account Team Size, suggested by Pendharkar et al.	Ability to explore an effort effecting parameters other than Total Size. Note: Previous studies show that a scale of economies exists between team size and software effort []	E(work-hours) $=Ax^{b}Z^{c}$	over 500	No Info
MendesLokan–SingleCompany [68]	A regression model of a single company based on large dataset.	Projects in this large dataset use standard procedures of the same -single- company	E(work-hours) $= 64.2 \times ufp^{0.635}$	184	IFPUG
MendesLokan–CrossCompany[68]	A regression model based on Large ISBSG data set.	To evaluate whether a cross-company model could be feasibly implemented	E(work-hours) $= 17.27 \times ufp^{0.897}$	672	IFPUG
Tronto-Silva-Anna ANN [49]	A formulated neural network model	An ANN based model, built using large Cocomo dataset.	E(work-hours) =-1,68+1,676*x	63	FPA

One of the drawbacks of LSR is that the fitting function is directly affected by outliers. Since we eliminated our single outlier, it was possible to use this method for both the single input and multivariate regression analysis. With the aid of a regression analysis tool the regression models were formed. For multivariate models, the effect of each BFC was included with different multipliers.

Table 24 Specific Regression models constructed on the basis of the company dataset

	Model Name	**Standard COSMIC FP**	**FS Reflected**
1	Linear	190,8+1,018FP	82,1+1,98FP
2	Power	$4{,}12FP^{0{,}82}$	$4{,}12FP^{0{,}82}$
3	Exponential	$343{,}97\exp^{1{,}037FP}$	$466{,}18\exp^{1{,}2FP}$
4	Logarithmic	-2700,4 +587,34 lnFP	-2679,54 +654,7 lnFP
6	Multi-variate Linear	2,35E-0,7X+5,88R-5,58W+842,3	64,7+3,16E+3,44X-0,91R+3,3W
7	Multi-variate Exp.	EXP(-0,005E+0,002X+0,003R-0,005W +6,74)	EXP(0,005X-0,005R+0,003W+6,195)

To perform an OLS analysis, a confidence interval should be set. The majority of earlier studies in this domain selected this value as 95% [24][25][27][28]. By using this value, a balance between the complexity of model and prediction performances is created.

The results of the specific regression models formed by the datasets in the current study are shown in Table 24 with two different representations for each type of model. These were the standard calculation of the COSMIC and the calculation when the FS list was applied. The quality of these specific models in terms of commonly used comparison parameters is given in Table 25 in the rows indicated as ‘specific’.

For the generation of all models, all projects from 3 different application domains are used. For specific evaluation of different application domains a larger dataset is needed.

4.3.1.9.6. Building specific ANN Models

In literature, the technique used in a neural network for predicting software estimations is a combination of two ANN architectures: 'back propagation' trained 'multilayered feed forward' networks. As an activation function the 'Gaussian activation function' is generally used. We also selected the same structure of network since it is the simplest and comprehensive neural model. These types of ANNs have two different phases, namely, training and execution. In the training phase the ANN is trained to return a specific output when given a specific input, this is carried out by the back-propagation algorithm. In the execution phase ANN returns outputs on the basis of inputs. Our input numbers for this network were four, the number of BFC types in the COSMIC method, and the output was one for the effort.

The performance of an ANN depends on its architecture and certain parameters such as the number of layers, the number of nodes in each layer, the transfer function in each node, learning algorithm parameters and the weights which settle the connectivity between neurons. Thus, inappropriate selection of these parameters may result in serious difficulties in network performance and training. To decide on the most appropriate ANN model, two approaches can be applied: one is the trial/error approach and the other is the application of the Genetic Algorithm. The second approach is feasible to be used in larger data sets.

In our research the steps in the trial/error approach were as follows: Divide the network randomly into two, for example; 75% for training set and 25% for validation. Select an optimal network model, and then train it using the training set. Check its validation using a validation set and calculate the prediction errors. Apply the same process to the other networks then chose the networks which have a lower prediction error for the validation set compared to other networks. Optimal network selection depends on the number of neurons, levels and datasets. A major limitation in the use of a neural network as an analysis tool is that it requires a large dataset for training. For small data sets, it is necessary to make the training more efficient. One way of doing this is to repeat the training process several times with a randomly selected training set, and then the model is tested with the validation data. For the selection of the number of levels, the size of the dataset will give a clue and since our data set was not very large, one hidden layer was enough to build an ANN model. However, to observe the differences, we also created a two level network. For the decision concerning the number of neurons a rule of thumb was applied. According to Heaton the 'The number of hidden neurons should be less than twice the size of the input layer' [147]. Training was implemented for the one and two hidden level networks where the hidden neurons were less than eight. To build an optimal neural network, several training parameters must be arranged. Training was performed by a back propagation algorithm that passes errors back through the network to update the weights at the neurons. There is a need to set some parameters for this algorithm such as learning rate and momentum. These selections either improve the speed at which the network reaches the minimum error or causes the network to diverge completely. Therefore, for inexperienced users the selections should be the defaults of the analyzing tool. In the current research the default value for learning rate is applied as 0,5.

Similar to the OLS regression analysis, to complete the training a stopping criterion exists for ANN models; in the current study we selected this as the Mean Square Error (MSE). After the application of the trial and error approach we decided to use two network topologies: '4:5:1' and '4:3:4:1' since they produced the minimum prediction error. Then these networks were created and applied to our data set for the execution of a number of iterations and calculated the network predictions. It was found that after a number of iterations the network converges to a model that represents dataset well. So the network evaluation can be stopped at around 500 iterations. At this point the final weights at the neurons can be used to formulate the ANN effort model. To investigate the effect of functional similarity the same process was applied to the FS reflected sizes.

4.3.1.10. Comparing the accuracy measures of the models

For the comparison of the goodness of fit of effort estimation models some parameters are utilized. The Mean Magnitude Relative Error (MMRE) is an average error used to indicate the relative amount by which the predictions over or underestimate the real value for the model. The coefficient of multiple determinations, R^2, on the other hand, gives the percentage of the variation that can be explained by the independent parameters. If R^2 approaches 1, it can be said that a strong relationship exists between the independent and dependent variables. The prediction level parameter PRED (k) represents the quality of the predictions. It defines the percentage of estimated values within k% of the actual measured values. We calculated these values for each effort model. Table 25 gives the accuracy measures of all the effort models. In order to compare the accuracy of the multi-variate regression and ANN models, we used the same inputs as in the multi-variate regression models. We presented accuracy results of all models in same table to decide several issues at the same time: usage of single size-or BFC component, usage of generic or specific model for effort, and reflection of Functional similarity in size measurement.

Table 25 Comparison of effort models based on Total Functional Size (n=17)

Type	**Model Name**	**No FS**			**FS Reflected**		
		R^2	PRED (0.30)	MMRE	R^2	PRED (0.30)	MMRE
Generic	Albrecht-Gaffney	0.39	0.06	511	Not Applicable		
Generic	Kemerer	0.46	0.00	73249	Not Applicable		
Generic	Matson	0.39	0.00	1769	Not Applicable		
Generic	Cobb-Douglass	0.41	0.30	83	Not Applicable		
Generic	Mendes&Lokan Single Company	0.38	0.00	563	Not Applicable		
Generic	Mendes Lokan Cross Company	0.39	0.00	819	Not Applicable		
Generic	Tronto-Silva-Anna ANN Model	0.39	0.35	114	Not Applicable		
Specific	Power	0.39	0.30	88	0.90	0.59	31
Specific	Exponential	0.46	0.12	97	0.83	0.35	77
Specific	Logarithmic	0.34	0.18	97	0.60	0.35	127
Specific	Hyperbolic	0.21	0.18	123	0.12	0.24	145
Specific	Linear	0.40	0.30	95	0.90	0.65	28
Specific	Multi-variate Linear	0.85	0.41	61	0.91	0.59	30
Specific	Multi-variate Exponential	0.82	0.41	97	0.86	0.35	76
Specific	ANN Model (1 hidden layer 500 iteration)	0.98	0.76	33	0.97	0.64	32
Specific	ANN Model (2 hidden layer 500 iteration)	0.97	0.64	38	0.94	0.76	33

Table 26 summarizes the results in terms of evaluation criteria for two ANN models. Two PRED ranges are given in this table thus, the prediction quality and overall fit of our models was investigated. Even convergence is satisfied at around 1000 iteration, we increased the number of iterations to observe the changes in goodness of fit of all the models. Table provides results for a number of iterations.

4.3.1.11. Results and Conclusions Case-1-Part3

According to the results given in Table 25 for generic models where total functional size was used as the only driving parameter, an accurate effort prediction model with an acceptable level of MMRE and PRED values could not be created. Therefore, none of the generic models were suitable for our data set. For the specific models that have single driving parameter, the situation was the same. As shown in Table 25, even the Linear and Power models built by our dataset did not produce acceptable results.

Utilizing the FS reflected size in single input regression models, the R^2, PRED and MMRE values were significantly improved, which means the FS improved reflected size increased the effort estimation accuracy. However, the MMRE and PRED values were still outside the acceptable limits for an accurate effort model. We concluded that building regression models utilizing total functional size as a single input might result in inaccurate effort prediction.

The Multivariate Regression models gave a better accuracy compared to the single input models in terms of all the evaluation criteria that were used. However, the PRED values indicated that they were not sufficiently accurate. For the multiple regression models the FS provided better results in terms of the R^2 and MMRE values. According to the PRED values, there was no improvement in the quality of the predictions.

When the results of the ANN and multi-variate regression models were compared, it was seen that the ANN models even for a low number of iterations provided much better results. Even when both the effort models used BFC components as inputs, the ANN approach led to a representation that explained the complexity of the overall data set and provided a better relationship among the BFCs for the overall data set.

Utilizing FS reflected size with the ANN models, caused the network to converge to a final effort model with a lower number of iterations. For both of the ANN topologies, which had a different number of layers, the maximum achievable PRED values were better if similarity was considered, thus demonstrating the importance of using FS concept when creating effort models. Whichever model was used the similarity consideration should be directly added to the effort calculations.

Table 26 Accuracy measures of the ANN models

	No FS				FS Reflected			
ANN Model	R^2	MMRE	PRED (0.30)	PRED (0.20)	R^2	MMRE	PRED (0.30)	PRED (0.20)
One-Hidden (4:5:1) - (100 iteration)N=17	0.90	64	0.47	0.41	0.94	37	0.53	0.41
-(5000 iteration)N=17	0.99	17	0.70	0.64	0.98	15	0.82	0.70
-(20000 iteration)N=17	0.98	8	0.88	0.88	0.99	10	0.94	0.94
Two-Hidden (4:3:4:1) - (100 iteration)N=17	0.86	66	0.53	0.41	0.91	64	0.47	0.29
-(5000 iteration)N=17	0.98	11	0.83	0.76	0.99	10	0.83	0.64
-(20000 iteration)N=17	0.99	10	0.88	0.88	0.99	9	1	0.94

Generic models produce significantly poor estimates for individual companies. Since this is the case, organizations should build their own prediction models that reflect the company's specific characteristics. According to the results we have obtained, this is only possible when the BFCs are treated separately in the models. However, the generation of specific multipliers for the effort prediction model that could be applied to all the projects in the company mainly depended on a reliable dataset. Therefore, to improve this process companies should work together with expert measurers or incorporate them into their organization.

We concluded that ANN and FS reflected component sizes produce excellent effort estimation models. Although we applied our empirical research to the COSMIC measurement method to discover the best approximation to represent the data set, the method is suitable for other component based functional sizing methods. By using the ANN method, companies will not rely on the fixed multipliers of the methods, they will be able to create specific multipliers that appropriately represent the characteristics of their company.

4.3.2. Case Study-2

Aim: To investigate the validity of effort estimation methodology. For this purpose, two research questions are inquired:

c. "Is methodology applicable?"

d. "Is it effective in building relation between size and effort?"

To evaluate the applicability of methodology, we assessed:

- Whether effort groupings are suitable for the company.
- Whether it is appropriate for different application types of the company.

To evaluate the effectiveness of this methodology on size and effort relationship, we performed analysis to see how the estimated effort compares with the actual effort if this methodology is used. For this reason, accuracy comparison parameters were used: MMRE, PRED and R^2 and following issues were investigated.

- Whether it provides more control over execution of projects by providing feedbacks and project elimination to produce better estimations.
- Using Development Effort instead of Total effort in building relationship.
- Deriving a new relationship equation for each application type.

4.3.2.1. Prepare and Collection of Data

We assessed the EFES methodology by using the projects of the same company that we conducted our previous cases.

The methodology we proposed requires all processes, templates antecedently established. Then applying it precisely and deriving benefit takes at least one or two years depending on the company.

Therefore instead of building that database from the scratch, we took the way of using existing data in company. So it is a retrospective study. SCIs for this case study was selected from projects whose project team collected their effort at least for 2 years and SCIs are officially released. We removed the following SCIs from our analysis.

- If detailed phase activities are not recorded,
- If SRS document does not exist or is not updated,
- If time spent in other works is not recorded,
- If requirement change decision is done by the company during the implementation to build another infrastructure or to create a new environment,
- If extra integration of SCI module is done for R&D or maintenance purposes.

We expanded our database in Case-Study1 by including results of finished projects in the last two years. After the elimination phase as given in 4.2.3.1, selected projects are as follows: we have 12 GUI, 5 BSP and 9 ES project SCIs. So, the total number of projects under analysis became 26.

SCIs are belong to 3 application domains as given in Figure 24. We ignored the Algorithm Development part in that figure for this assessment because very low number of projects collected algorithm development effort in detail.

4.3.2.2. Case Study-2: Conduction

This case study has been performed by applying the five processes of methodology. Effort collection, Size Measurement and Data Analysis processes are applied fully. Effort Estimation Process aimed for the estimation of a new project requires a drift in time. We applied this process for one of the GUI projects since we had no previous effort model of other type of applications. Similarly for calibration process: we developed a new effort model for GUI projects. Details of the findings for the company are as follows:

4.3.2.2.1. *Effort Collection Process*

For the application of this process we evaluated the Effort Collection Records. We selected this approach, because applying the newly defined item list for projects and collecting new effort data will take two-three years. Therefore, instead of directly applying the process with the new Pre-Defined Development Effort items and Supporting and Extra effort item lists, we used them as guidelines to review, and evaluated the project records of finished projects. The steps were as follows:

[1] Standardization of Records: We looked for which effort items in Table 31 and Table 35 exist in effort records in common.

- It was found that even all items exist in records, only a part of these list included nearly all projects that are as follows:

  For Table 31: “Software Requirements, Design, Coding, Code Test, Integration and Qualification”

  For Table 35: “Documentation, Requirement Changes, HW Problems”

[2] Elimination of effort records: Even System requirements and System Architectural part were performed for all the projects, only a few projects recorded these items separately. We learned that most of the team leaders only recorded software related development. So, even their project team spent effort for this part, they ignored effort recordings on these items. So, we removed system related effort recordings from effort datasheets to satisfy commonality.

[3] Groupings of effort items for specific domain: If an item is only applicable for specific application types, then we included them in Effort calculations.

- For example: A standard procedure application for safety requirements is included for most of the ES projects.

Table 27 Effort Distribution in Phases

Project	*Req.*	*Des.*	*Code & Test*	*Test (Integ.&Qual)*	*Doc*	*Req Chg.*	*HW Prob.*	*SafetyTest*
Project1-GUI_Keyb	13,5	10	30	15	18	9	32	0
Project1-ES_Target	5	3	17	41	16	10	6	4

As we mentioned in part 4.2.3.1 Case Selection part, for this case study we had newly finished projects for each application domain: 3 new GUI, 8 new ES and 2 new BSP project.

A Part of Table data showing the effort distribution of a project is given in Table 27.

4.3.2.2.2. *Size Measurement Process:*

This process was applied for 6 large size projects that totally consist of 26 SCIs. Measurements and BFC groupings were performed by using COSMIC method. During size measurement ofES, we encountered that, measurement experts had different cerebrations on some specific situations and discussions were needed to finalize measurement. We reviewed and updated first measurements according to final decisions. The measurements were validated by expert and cosmic certified measurers by sampling method. Part of size measurement results are given in TABLE 28. These measurements were recorded in CUBIT database [180] and they were controlled by using the inspection checklists.

Table 28 Size Measurement Results for a Specific Application Domain

GUI Projects	*# of Entry*	*# of Exit*	*#of Read*	*#of Write*	*Effort In Hours*
Project1-GUI_Keyb	57	73	236	222	1020
Project2-GUI_DLoad	226	745	174	247	396
Project6-GUI_Screen	58	41	149	164	476

4.3.2.2.3. *Data Analysis Process*

Since data set was not large, and ANN and multivariate regression models produced approximate results, we applied this process by using Multivariate Regression analysis method and investigated the following cases on size and effort relationship. The results are provided for Case 2, Case 1 and all projects (Case-1 + Case 2 Projects)

- Effect of Application Domain on relationship,
- Development Effort-Total Effort Difference on relationship,
- Supporting and Extra Effort percentage on total effort,

Effect of Application Domain

To see whether effort models based on Application Domains improved estimation accuracy we grouped the projects based on their application type and created effort models for each of them. We also merged different Application Types and tried every combination of application types. Accuracy results of these models are given in Table 29. In this table all combinations are shown. We found that separation of application domains is necessary in effort estimation modeling. Merging of application domains generally resulted inaccurate estimations in terms of accuracy parameters.

Development Effort &Total Effort:

In order to check whether using Development Effort instead of Total Effort improved effort estimation accuracy, we created effort models for both of them, for each application type and groups of application types. The results are given in Table 29. For development effort we included "Requirements, Design, Code & Unit Test, Integration & Qualification" activities. For total effort documentation task, supporting and extra effort values were included. Based on the accuracy results, it is found that differentiation of variable types of effort is necessary. Size components, i.e Base functional components have better relationship with the development effort compared to total effort.

Analysis of Supporting & Extra effort values:

To see the effect of "other effort" factors we calculated the percentages of spend time for those activities in total effort spend. HW problems, requirements changes, documentation tasks and safety tests are the ones that we found in common. Results of our analysis are given in Table 30. Based on the findings, it can be said that %20 of effort is wasted for requirement changes in GUI projects. In terms of documentation tasks, very large percentage of effort is expended for the documentation of BSP projects. On the same table, for the first set of ES projects it seems that HW problems are not encountered much. However for the second set of same application domains, around %11 percentage of effort is wasted for such problems.

SCI Based Activity Analysis

Other than the aims of our study we investigated how effort distributions changed in years. According to the completion time of 9 ES projects the changes in "percentage of activities in total effort" are given in Figure 27 . We used average percentages of projects. This figure gave a top view of the efforts in terms of phases and tasks in company.

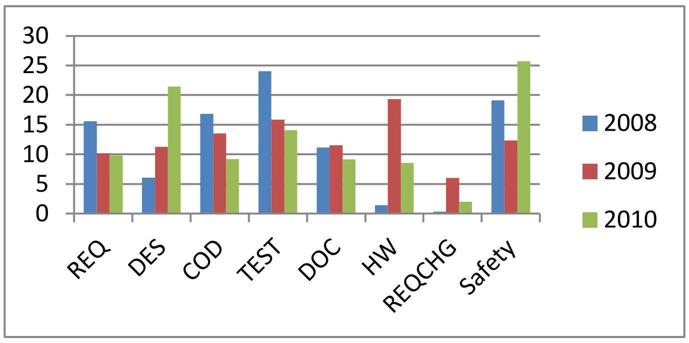


Figure 27 Average percentages of work distribution in years (for ES)

Table 29 Accuracy Comparison of Models Development Effort & Total Effort

	Case 2				**Case 1**				**Total Projects**			
Proj.(s)	**Dev.Effort**		**Total Effort**		**Dev.Effort**		**Total Effort**		**Dev. Effort**		**Total Effort**	
	R^2	PRED (30)	R^2	PRED (30)	R^2	PRED (30)	R^2	PRED (30)	R^2	PRED (30)	R^2	PRED (30)
GUI	0,86	0,83	0,34	0,83	0,81	0,83	0,90	0,64	0,76	0,59	0,32	0,41
BSP	0,62	0,85	0,55	0,57	Only 4 projects –Not enough to analyze				0,69	0,82	0,49	0,64
ES	0,71	0,91	0,60	0,73	Only 3 projects –Not enough to analyze				0,62	0,57	0,24	0,36
GUI, BSP	0,29	0,69	0,44	0,77	0,93	0,62	0,94	0,62	0,44	0,36	0,34	0,39
GUI, ES	0,65	0,71	0,57	0,65	0,56	0,21	0,17	0,36	0,27	0,35	0,08	0,35
ES, BSP	0,27	0,61	0,51	0,61	0,54	0,43	0,14	0,43	0,23	0,50	0,19	0,50
GUI,BSP,ES	0,24	0,63	0,41	0,48	0,60	0,39	0,32	0,33	0,15	0,27	0,19	0,35

Table 30 Percentages of Supporting & Extra effort Values

	Dataset version 1				Dataset version 2			
	Doc.	**%Req. chg**	**%HW Problem**	**%Safety**	**Doc.**	**%Req. Chg**	**%HW Problem**	**%Safety**
GUI	20,75	19,6	1,2	0	20,34	18,92	2,81	0
BSP	28,12	0,7	16,07	21	24,98	1,39	13,6	14,87
ES	11,17	0,35	1,4	19,1	10,63	3,16	10,81	15,43

.

4.3.2.2.4. Effort Estimation Process

To apply this process we needed historical data to be analyzed and an effort estimation model for specific SCI is formed previously. The only SCI type applicable for this process is GUI projects since in case study 2 we had 11 GUI projects completed two years ago. Other type of application domains did not have enough data to form a specific effort model. If we directly use total effort values of finished GUI projects for creation of the model:

Total Effort Equation (GUI) (man-hour) =6,67E -0,31 X+5R-1.85W +171,1

By applying this total effort equation for a new project whose values are E=23, X=22, R= 300, W= 14 we calculate total effort as: " 1639".

In our methodology our recommendation is to use development efforts for modeling. The effort model is:

Development Effort(GUI) (man-hour) = 0,76E -0,5X +2,11R – 0,46W +69,85

By using this model development effort is found as "703". At this point, we add "Supporting & Extra effort values for GUI projects: Based on available data of 11 GUI projects, this value was %20,75 extra effort for documentation, % 1,2 extra effort for HW problems and 19,6 for requirement changes. Totally %41,55 of total effort is expended for activities other than development. Therefore the development effort "703 man-hour" is %58,45 of the total effort.

So according to our methodology the total effort we need is:

703/58,45*100= 1202 man-hour.i.e 150 day

Actually expended effort for this project is “904”. i.e 113 day.

4.3.2.2.5. Calibration Process:

With the addition of new projects, GUI projects have a new model:

Development Effort (GUI)(man-hour)=138,5+0,16E- 0,46X+1,56R+ 0,1W

4.3.2.3. Results and Conclusions of Case Study-2:

In this case study- 2 , we investigated the validation of effort estimation methodology empirically.

Table 29 gives accuracy results of produced effort models in several situations. In case study-1, since the number of ES and BSP projects are low, producible effort models were either pure GUI applications or a combination of application domains.

As seen in TABLE 29, producing effort models for only one application type gives better results. Models for each type of application model produced higher values for both PRED(30) and R^2 compared to other groupings. Especially PRED values that represent estimation quality is found very high.

Results of (GUI-BSP) and (GUI-ES) groupings show that if we had cluttered the projects on a 5 dimensional environment (i.e. Entry, Exit, Read, Write, Effort) GUI and ES projects may be placed close to each other, while BSP projects will be located on a completely different area. For GUI-ES grouping in case study-2 both R^2 and, PRED (30) value is high for development effort compared to other grouped projects. However when all projects in these domains considered accuracy parameters drop directly. So, even they are found close to each other in methodology evaluation, the overall model produced for these two domains are not very representative. According to the obtained results, when different types of applications are grouped, any of the effort models will produce a good representation.

When we compared the Development Effort and Total Effort results, in general, Development Effort models produces better estimations. In other words the relationship between the development effort and BFC components of size is better than the total effort. Observations show that application type differentiation increases the accuracy in development effort models.

We performed a Data Analysis on ES projects to investigate how effort distribution has changed in three years time. It seems that requirement phase effort is decreasing while design effort is increasing. Even number of projects is not much, some impulsive cases are seen apparently. For example HW related issues had a peak in 2009. We enquired the reason for this. We learned that instead of using COTS products that has been tested before development, company had started to use their proprietary boards. Following such trends on activities obtained by using the methodology enable company to improve the quality of development and also estimations. There may be a relation between hardware correction activities mentioned above and requirement changes for ES systems.

The analysis results on “Supporting & Extra efforts” show that for some type of projects, when effort is estimated we certainly need to add extra effort values for specific activities or unexpected situations. Even the average has dropped in second analysis; safety related activities require %15 of total effort for ES and BSP type applications.

Effort estimation process was only applied for informative purposes in this case study. We estimated only one GUI project. Estimating development effort and adding up other types of efforts produced better results than estimating total effort.

One of our research questions was whether the methodology was applicable or not: We found this methodology applicable. According the results we obtained, the utilized lists of effort in our methodology seems suitable for the company. It resulted better analysis of projects and accurate estimations. In our case study we had the possibility of applying a part of these effort lists. This part is nearly 80-90% of whole list. Application of full list will provide more valuable results.

Since all three application types gave encouraging results in terms of accuracy of effort estimations, methodology is applicable to any type of applications. Actually using the methodology separately for each application domain is more appropriate to reach reliable results. To do this as methodology suggested concerning deviances in size measurement and analyzing other type of efforts are necessary.

In terms of efficiency, one of our questions was related to whether methodology provides control over execution of projects. During the effort collection and size measurement process, it provided feedback to the developers by evaluating the SRS documents. It was also used for the elimination of the projects to produce better results. In execution of the methodology, the problems found in eliminated projects are fixed by developers, and then these projects will be used as input to methodology again to reach more accurate effort models for the company. Methodology is also effective in analyzing the company situation by using data analysis process. These results also control some other processes in company. For example at least 10% of effort is wasted for HW problems which mean either HW development or COTS selection process has some problems.

In terms of efficiency, the accuracy improvement provided by methodology is another question. Based on our findings we can say that application domain type affects many activities and related efforts. It allows us to create a better relationship between size and development effort. The methodology is effective in obtaining real development effort that has better relationship compared to the total effort.

The quantitative findings in our study validate the application of our methodology. We aimed this methodology to be used by middle or large sized companies. But a similar methodology can be defined for information collection of cross-company datasets in future again by using the distribution of activities based on the standard IEEE 12207 [155]. Even, there are some guides for data providers of cross-company datasets, it seems that current guidance is not enough to normalize and match available information of companies at final phase.

4.4. Validity Threats

The main threat to validity of the study is that we performed this study in only one organization. In order to minimize this threat and to generalize our results we utilized the same methodology cycle for three different application domains.

The selection of subjects in our study was special. We chose the projects having necessary records that mostly match our proposed lists since we didn't have time to collect data using our methodology. To improve trustworthiness of data quality, we went through strict data cleaning and exclusion of projects based on the discussions with developers of the project. We had a number of projects to make this elimination for the evaluation of the data collection process of methodology. We performed a retrospective study.

In another company there will not be so many projects for such elimination to be applied. Therefore the required and proposed way is of course to use data collection process as defined in this study and utilizing pre-defined given lists.

We selected projects having team leaders who displayed willingness to participate. So, projects do not reflect all the projects of the company. The projects that were controlled by other team leaders may include important items that require larger effort to be expended and that effect the final estimations.

A kind of bias selection existed in this study: in case study 1, in order to see the FS effect, GUI projects were selected that had apparent similar modules. In the same study projects from different domains were also included in this data-set for comparison. But data-set was still small, that might threaten internal validity. Besides, many data in our dataset belongs to lower functional size values. Therefore, this study if carried out with a larger population from different application domains, with larger functional sizes, might yield different results.

In this company application domain developers were teams having expertise on a specific subject. All teams are using common procedures that were already settled previously. So, developers had been using them for years. In another company that is not mature enough in software development processes the results may be different.

All projects in this study were developed in the same environment. In other words, projects from third parties were not included in the case selection. In such a case, another categorization under specific application domain type will be necessary.

The instruments used for the analysis of data and produced models and their parameters were verified by using other commercially similar tools, like "SPSS Statistics Tool" [182], DataFit [181]

To prevent subjectivity of size measurements and to prevent errors on this issue, the measurements were reviewed by other certified experts and periodic discussions were performed.

CHAPTER 5

5. CONCLUSIONS

Majority of the effort estimation studies focus on one part of the estimation process. Many researchers have been addressing the new size estimation techniques that result in better effort correlations. Similarly, several ways of data analysis like ANN, Fuzzy, Bayesian networks are evaluated and compared to find best handling method [7][41][9][10] [11]. Results of these studies are valuable and have considerably positive effect on effort estimation process. However, each recommended solution is constructed by the assumption that, the historical data is very reliable. But without a reliable data set, these solutions are neither repeatable, nor usable even for different teams of the same company. Unfortunately, obtaining reliable data is very difficult due to some other effects in the environment. These effects mainly the conflicting information in repository stems from the differences in processes, definitions or grasp of practices by people.

The term ‘Reliable data’ is used to represent a number of issues: First, it means that collected effort data is accurate, consistent, and informative for analysis purposes. Secondly, size measurements are valid and repeatable. All the information should be homogenously created and developed across the organization, in terms of personnel, projects and processes. In a reliable dataset, analyzed data is extracted from a controlled repository, which has been established using the results of projects that are constructed and managed via homogeneous processes and tools. Besides no conflicting or missing information exists. Using single company data will minimize some differences in processes and tools. However, if companies don’t have well-established standardized processes, and common interpretation is not constituted among software teams the same problems in cross-company cases will be observed.

In the last years, one of the research questions was whether different components of functional size measurements be included in effort models or not. Some measurement methods assign some values for these components during size measurement period that represent their respective contribution to total effort like IFPUG and MKII. However, the multipliers of these methods are fixed. The better estimation accuracy will be achieved by considering the environment, conditions of the project on the effect of these multipliers.

In building the specific effort models one of the improvement opportunities in measuring functional size was Functional Similarity. This is related to the fact that the structural aspect of the software and its influence on the development effort has not been well reflected in existing estimation models. The re-utilization of existent data structures and functionalities to build new structures directly affect the required development effort. Most of the time the development and test effort will be lower than that required to create a new functionality.

Therefore, we developed a methodology that includes core processes of effort estimation approach, that can be modifiable based on new improvements and that can be automated in the future.

Even previous studies analyze the reasons for estimation errors, the ways of improving accuracy in estimations, and the dataset-related problems, there is no defined estimation methodology that guides the companies in their estimation journey by combining all these findings.

Many standards related to software development as if CMMI recommends doing effort estimation. But up to now, a fully defined methodology that defines “how to do this” still does not exist. Software companies need a guide that promotes consistency among their development phases, terminology, collected information etc.

For this reason we developed a methodical way that would guide the organizations. Our extensive survey in the literature revealed the requirements for the processes. In parallel, we investigated the reasons for effort estimation errors. By performing two case studies, we focused on following issues:

In case study-1 we assessed the existence of some concepts in our methodology by using three steps. In the first step the basic concept of Functional Similarity was investigated, in the second step the same concept and its effect on project phases were investigated. Besides the percentages of some tasks for specific application types were evaluated. In the third step, application of different data analysis methods" and "Effects of BFC components on building effort models", were examined. Besides the applicability of EFES methodology were applied by analyzing "Application type importance in effort estimation accuracy" and "Work effort distribution on effort estimation models". We also searched for how unexpected or unplanned efforts affect estimation accuracy.

Then we integrated the results of these findings with the real-time operations of the company we selected and formed generic process models.

In our methodology by using the ANN method and Multivariate regression method, effort estimation models were created using the four components of COSMIC method and for which specific multipliers were generated that were calibrated according to the parameters from the company. Although, previously, neural network approaches were suggested as a way of creating effort estimation models, our methodology is the first one that uses the components of functional sizing methods as inputs to the ANN models.

In our methodology we also included the FS concept in creation of effort models and validated its usage in effort estimation quality.

This thesis has addressed a range of issues arising from the relationship problem between size and effort. This chapter examines the contributions in more detail, summarizes the benefits and suggests future research directions based on the findings discovered during the thesis study.

5.1. Contributions

In case study 1; our motivation is to observe how functional similarity has an impact on different application domains and phases of project. Regardless of the aimed logical functionality, if functional processes of modules are similar to any other process, effort needed to develop new ones will decrease, and total effort will not be proportional to the functional size. Few studies examined distinguishing similarities and quantified reuse potential of projects. However validation of these results for different application domains and how to reflect these results into effort models has not been studied. We found that especially for specific phases and tasks similarity has a considerable effect.

In case study-1 we also compared several functional size based effort models in terms of accuracy using a reliable company dataset. These models comprised not only the generic models proposed in the literature or currently in use, but also specific models that we generated using our dataset with a single and multivariate regression analysis and the ANN method. The important element in this research was the comparison of several functional size based effort models using a reliable dataset. In general, data collection methods are not uniform across different companies. Differences in processes and practices are not reflected well in the cross-company datasets. By using a single company data we minimized these differences. Furthermore, to ensure the reliability of the effort data an organization was selected where well-defined requirements specification documents exist and effort data is gathered on a daily basis with the aid of standardized time-sheets. Since the data creation and collection is performed in a controlled manner, by qualified staff, there were no missing values in the set. Besides, in our research to ensure the reliability of the size data, all project measurements were performed by expert measurers and a software design leader in the company who had also been trained in size measurement and the results were verified by a further two different expert measurers.(Measurement group) One contribution of our study is that it is the first time that the COSMIC has been used for ANN models. Companies may use either multivariate regression models or ANN for component based model creation. Even though we applied COSMIC method for our data set, other methods, which have different functional size components, may apply these methods in their effort model creation. For ANN structure creation we applied trial and error methods by using the recommendations of the studies [147] [154]. Optimal ANN structure must also be investigated.

The main contribution of this thesis is that, it is the first time that an effort estimation methodology is defined with processes, their artifacts, assets, templates. It is valuable because it embraces the literature results on effort estimation, the best efforts suggested and applies the method practically. Even some studies mentioned or suggested some ways to collect data, to investigate effort trends but we didn’t encounter any quantitative validation for the overall estimation processes. Our study is an empirical validation of the related processes on a reliable dataset. Contributions are also made by developing the procedures in detail, and set of flow diagrams.

The results discovered in case studies and investigations during the research study revealed that the proposed method can be successfully utilized for effort estimation. The EFES methodology enables companies to reach meaningful and reliable information for effort estimation while removing inconsistencies and problems in related processes. Moreover, the artifacts of this methodology are aimed to be used for project and company monitoring.

With the aid of data collection process, team members can take the responsibility on their work, make plans on their individual assignments. Team leaders and project managers can observe quantitatively the development state of the product and can create plans evolving from a latent state. They can also identify problematic areas. From the company's point of view several data analysis can be done. With the data obtained from processes it is possible to supervise the status of company on some subjects, to observe improvement opportunities or problematic areas.

In other areas of software development, processes are nearly standardized by using international standards and assessed by specific experts (ex: CMMI). Processes became mature by measuring parameters and controlling them for years. Similarly in effort estimation processes we need to control the work by using well defined processes. It can be assumed as a sub-discipline of project management. In this thesis work effort distribution is established by matching the processes of IEEE 12207 and activities of the projects. So, if other companies use the same structure in collecting their effort data, this predefined WBS structure may provide useful results for cross-company data sets.

5.2. Limitations

The case studies showed that the expected benefits from the methodology are not fully realized if the defined effort estimation processes and SW development processes are not performed or not effectively established in the organization. Especially, for the effort data collection process, all teams must follow the predefined WBS structure. Otherwise, the aim disappears, and for analysis sufficient amount of data may not be available again. Besides, inconsistencies and incompleteness in data sets may occur. A common understanding is necessary to be formed. The Measurement and Analysis group (M&A Group) and Software Manager is responsible for this issue, they should give necessary training for the execution of process.

Another limitation is related with the analysis tools used. If an analysis method is chosen, not only the usage of that tool but the algorithm behind it should be known by the responsible M&A personnel. Besides some tools, like ANN, requires larger data set to build a reliable model.

We have created our effort models from a limited number of projects that has started from scratch and released. Besides, process implementations and development processes are very well known by teams. Therefore the generality of the conclusions will benefit if this study to be performed on other companies with varying number of projects. Even the methodology is applied in a mature organization, that has knowledge about processes & procedures, small companies can also use it. We found company mature enough in terms of process definitions, documentation and execution. However the applicability based on process maturity of company must also be investigated.

We have currently no data for larger or smaller organizations. Similarly we have no implementation for the maintenance projects.

The methodology we proposed does not consider the life-cycle models. However all the projects in our dataset had been developed using Waterfall Life-cycle model. If project is generated using several builds as in agile development and iterative life-cycles, the estimation will be calculated by considering specified tasks for each successive build. In this case evaluation of SRS is performed for only the items that had finalized for that specific build.

5.3. Future Research

In future, we will expand our data set by including the new projects not only from this company, but also from other ones. Such study will provide new kinds of problems and related efforts to be included for specific application domains. As a result of this a new list will be formed for common use of companies. As a next step by using a larger data set, we aimed to build a relationship between BFC components and each task and phase of effort. Besides, since all the steps of processes are defined with required templates and responsible people, an automated information system that can be developed.

REFERENCES

1. Farr, L., Nanus, B., (1964) “Factors that affect the cost of computer programming v.1.” United states air force Electronic Systems Division., from http://www.dtic.mil/cgi-bin/GetTRDoc?Location=U2&doc=GetTRDoc.pdf&AD=AD0603707

2. Jørgensen, M., Shepperd, M., (2007) “A systematic review of software cost estimation studies. IEEE Transactions on software engineering

3. Albrecht, A.J., Gaffney, J.E, (1983) “Software function, source lines of codes and development effort prediction”: A software science validation. IEEE Trans. Software Eng., SE-9: 639-648

4. Nelson, E. A. (1966) “Management Handbook for the Estimation of Computer Programming Costs”. AD-A648750, Systems Development Corp.

5. Helmer, O., (1966) “Social Technology”, Basic Books.

6. Morgenshtern, O.,, Raz T., Dvir, D., (2007) “Factors Affecting Duration and Effort Estimation Errors in Software Development Projects”. Information & Software Technology

7. Boehm, B.,(1981) “Software Engineering Economics”. Prentice Hall,

8. Mendes, E., Kitchenham, B.A., (2004) “Further Comparison of Cross-Company and Within Company Effort Estimation Models for Web Applications. Proceedings Metrics”, IEEE Computer Society, pp 348-357 .

9. Déry, D., Abran, A.,(2005) “Investigation of the Effort Data Consistency in the ISBSG Repository”, in 15th International Workshop on Software Measurement -IWSM'2005, Montreal, Canada,

10. Jeffery, R., Ruhe M., Wieczorek, I., .(2001) “Using public domain metrics to estimate software development effort. Proceedings Metrics’01, pp 16-27.

11. Jeffery, R., . Ruhe M., Wieczorek, I., (2000) “A Comparative Study of Two Software Development Cost Modeling Techniques using Multi-organizational and Company specific Data. Information and Software Technology”, pp 1009-1016.

12. Wieczorek, I., Ruhe M.,. (2002) “How valuable is company specific data compared to multi-company data for software cost estimation?” Proceedings Metrics’02, ,pp 237-246.

13. Deng, K., (2008) “The value and validity of software effort estimation models built from a multiple organization data set”, Master Thesis, Auckland University of Technology,

14. Finnie G.R., Wittig G.E., Desharnais J.M., (1997) “A Comparison of Software Effort Estimation Techniques: Using Function Points with Neural Networks, Case-Based Reasoning and Regression Models”. Journal of Systems and Software, vol. 39, no. 3, pp 281-289,

15. European Space Agency, “SME training on Configuration Management”, Retrieved May 2011from http://esamultimedia.esa.int/docs/industry/SME/Configuration/Section_4-CM.pdf

16. Dolado, J., Fernandez, L., (1998) “Genetic Programming, Neural Networks and Linear Regression in Software Project Estimation. Proc. the INSPIRE III, Process Improvement Through Training and Education

17. Kumar, S., Krishna B.A,Satsangi P.S., (1994) Fuzzy systems and neural networks in software engineering project management, Journal of Applied Intelligence 4, 31-52, 1994

18. Chulani, S., Boehm, B., Steece B.,(1999) “Bayesian analysis of empirical software engineering cost models”. IEEE Transaction on Software Engineerining, vol. 25

19. Genuchten, .V, (1991) “Why is software late? An empirical study of reasons for delay in software development”. IEEE Transactions on Software Engineering

20. Lederer, A.L, Prasad J.,, (1995) “Causes of inaccurate software development cost estimates”. Journal of Systems and Software.

21. Subramanian, G.H., Breslawski S.,(1995) “An empirical analysis of software effort estimate alterations”. Journal of Systems and Software

22. Gray, A., MacDonnell S.,, Shepperd M.,. (1999) “Factors systematically associated with errors in subjective estimates of software development effort: the stability of expert judgment”. in Sixth International Software Metrics Symposium. IEEE Comput. Soc.

23. Jørgensen M., K. Moløkken,Ø.,, (2004) “Understanding Reasons for Errors in Software Effort Estimates”, IEEE Transactions on Software Engineering, vol. 30, no 12

24. Jorgensen M.,Gruschke T., (2009) “The impact of lessons-learned sessions on effort estimation and uncertainty assessments”. IEEE Trans. Softw. Eng., 35(3):pp 368–383,

25. Cocomo II Model Definition Manual (2000), “Univ. of Southern California Center for Software Eng”.;retrieved May 2011 from sunset.usc.edu/research/COCOMOII/Docs/modelman.pdf

26. McConnell, S.,(2006) “Software Estimation – Demystifying the Black Art”; Microsoft Press, RWa; ISBN 10: 0-7356-0535-1.

27. Stern, S.,(2009)“Practical experimentations with the COSMIC method in Automotive embedded software field", IWSM’09,

28. Stern, S., Guetta, O., (2010) "Manage the automotive embedded software development cost by using a Functional Size Measurement Method (COSMIC)", European Congress of ERTS (EMBEDDED REAL TIME SOFTWARE**)**

29. Khelifi, A.,, Abran, A.,, (2007) "Design Steps for developing Software Measurement Standard Etalons for ISO 19761 (COSMIC-FFP"), WSEAS International Conference on COMPUTERS

30. Mills, H. D.,(1983) "Software Productivity", Little Brown and Co.

31. Tausworthe, R.C., (1980) "The Work Breakdown Structure in Software Project Management", Journal of Systems and Software 1, pp 181-186

32. Albrecht, A.J., (1979), Measuring Application Development Productivity, IBM Applications Development Symposium,

33. Capers, J.,(2008) "A Short History of Lines of Code (LOC) metrics", Capers Jones & Associates LLC, , Retrieved May 2011 from http://www.measuresw.com/library/Papers/Others/LinesofCode2008_CJ.pdf

34. ISO/IEC IS 20968: (2002), "Software Engineering -- MK II Function Point Analysis - Counting Practices Manual", International Organization for Standardization

35. ISO/IEC IS 20926: (2003), "Software Engineering-IFPUG 4.1 Unadjusted Functional Size Measurement Method-Counting Practices Manual, International Organization for Standardization",ISO

36. ISO/IEC19761 (2003), "Software Engineering – COSMIC-FFP – A Functional Size Measurement Method", ISO,.

37. Kim, S., Lively, W., Simmons, D., (2006) "An Effort Estimation by UML points in the early stage of software development", proceedings of the international conference on software engineering research & practice, p 415-421,

38. Symons, C. (1988). "Function Point analysis: Difficulties and improvements". IEEE Trans. Softw. Eng. 14, 1, pp. 2–11.

39. Suryaningsih,(2000) "The applicability of function points to the domain of embedded telephone switching systems, CAESAR", Thesis from University of New South Wales, Retrieved April 2011 from http://www.caesar.unsw.edu.au/publications/pdf/Tech00-10.pdf

40. Jones C., (1986) 'The SPR Feature Point Method", Software Productivity Research Inc., 1986

41. Symons, C., (2001) “Come Back Function Point Analysis. (Modernized) – All is Forgiven!”, 4th European Conf. on. Software Measurement and ICT Control, FESMA, DASMA,

42. Banker, K., Kauffman, R., Wright C.,, Zweig D. De, (1994) ., , “Automating Output Size and Reuse Metrics in a Repository–Based Computer–Aided Software Engineering (CASE) Environment,” IEEE Trans. Software Engineering, Vol. 20 pp. 169–184,

43. Issa, A., Odeh, M., and Coward, D., (2007) “Can Function Points Be Mapped To Object Points?” International Arab Journal of Information Technology, 4 (1), p. 1,

44. Kim S., Lively W., Simmons D., (2006) “An Effort Estimation by UML points in the early stage of software development”, proceedings of the 2006 international conference on software engineering research & practice, p 415-421,

45. Kusumoto S., Matukawa F., Inoue K., Hanabusa S., and Maegawa Y., (2004) “Estimating Effort by Use Case Points: Method, Tool and Case Study,” Proceedings of the 10.International Symposium on Software Metrics METRICS04,

46. S. Berlin, S., Raz, T., Glezer, C., Zviran, M., (2009) “Comparison of estimation methods of cost and duration in IT projects”. Information and software technology. Vol. 51, Issue 4, pp.738-748

47. Stensrud, E., Myrtveit I., (1998)“Human Performance Estimation with Analogy and Regression Models”. Proc. the 5th METRICS 98 Symposium, pp205-213

48. Briand, L. C., El Emam, K., Surmann, D., Wieczorek, I., Maxwell, K. D., (1999) "An assessment and comparison of common software cost estimation modeling techniques," presented at The 21st international Conference on Software Engineering,

49. Tronto, I.F. D.B.,. Silva, J.D.S, Sant’Anna, N., (2007) “Comparison of artificial neural network and regression models in software effort estimation”, in:Proceedings of International Joint Conference on Neural Networks,.

50. Gray, A.R., MacDonell, S.G., (1997) A comparison of model building techniques to develop predictive equations for software metrics, Information and Software Technology,

51. Marza, V., Teshnehlab, M., (2009) “Estimating Development Time and Effort of Software Projects by using a Neuro_Fuzzy Approach”, Retrieved August 2011 from http://www.intechopen.com/source/pdfs/8711/InTech-Estimating_development_time_and_effort_of_software_projects_by_using_a_neuro_fuzzy_approach.pdf

52. Pedro, I., E.Colla, (2008) “Subjective Consistency”, Retrieved May 2011 from http://www.frcu.utn.edu.ar/deptos/depto_3/32jaiio/asse/asse_11.pdf

53. Heemstra, F.J., (1992) “Software cost estimation, Information and Software Technology”, 34:pp 627-639

54. Boehm, B., Abts, C., Chulani S., (2000) “Software development cost estimation approaches––a survey”. Annals of Software Engineering 10: pp 177-205.

55. Murmann, P.A., (1994), “Expected development time reductions in the German mechanical engineering industry”, Journal of Product Innovation Management 11, pp 236-252.

56. Norris, K.P., (1971), The accuracy of project cost and duration estimates in industrial R&D, R&D Management Vol.2, pp 25-36., Wiley

57. The CHAOS Report, (1995) The Standish Group International

58. Moløkken-Østvold., K. J., (2004) “Effort and schedule estimation of software development projects”. PhD thesis, University of Oslo, Norway

59. .Balsera, J. V, Fernandez, F. O.,.Montequin, V. R, Suarez, R. C., (2009) “Effort Estimation in Information Systems Projects using Data Mining Techniques”, Proceedings of the 13th WSEAS International Conference on COMPUTERS

60. Briand, L.C. and Wieczorek I.,(2001) “Resource modeling in software engineering, in Encyclopedia of Software Engineering”, J. Marciniak, Editor, Wiley.

61. Briand, L. C., and Wieczorek, I., (2002) “Resource estimation in software engineering,” in Encyclopedia of software engineering, J. J. Marcinak, Ed., 2nd ed. New York: John Wiley & Sons, 2002, pp. 1160-1196.

62. Leung, H., Fan, Z.: (2002) “Software Cost Estimation. Handbook of Software Engineering”, Hong Kong Polytechnic University

63. Briand, L. C., Wüst J., (2001) “Modeling Development Effort in Object-Oriented Systems Using Design Properties”. IEEE Trans. Software Eng., pp 963-986

64. Jiang, Z., Naudé, Z., P., (2007)”An examination of the factors influencing software development effort”.International Journal of Computer, Information, and Systems Sciences, and Engineering1(3), pp 182-191

65. Kemerer, C. F. (1987). “An empirical validation of software cost estimation models”. Communications of the ACM 30(5): 416–429.

66. Matson, J.,. Barrett, E. B. E., Mellichamp, J. M.,(1994), “Software Development Cost Estimation Using Function Points”, Transactions on Software Engineering vol. 20, no. 4, pp. 275-287, IEEE Computer Society

67. Pendharkar, P.C.,.Rodger, J.A., Subramanian G.H., (2008) “An empirical study of the Cobb-Douglas production function properties of software development effort”, Information and Software Technology, v.50 n.12, pp.1181-1188,

68. Mendes, E., Lokan, C., Harrison, R., Triggs, C., (2005) “A replicated comparison of cross-company and within-company effort estimation models using the isbsg database”. 11th IEEE International Software Metrics Symposium, p. 36,.

69. International Function Users Group, (2004) “Function Point Counting Practices Manual-Release 4.2”

70. Fenton, N. E., Pfleeger, S. L.,(1996) “Software Metrics: A Rigorous and Practical Approach”, 2nd Ed., International Thomson Computer Press

71. Nguyen, V. , Steece, B. , Boehm B. ,, (2008) “A Constrained Regression Technique for COCOMO Calibration”, New York: Assoc Computing Machinery

72. Phan, D., Vogel, D. , Nunamaker J.,(1998) “The search for perfect project management. Computerworld”, vol 22: pp 95–100

73. Oligny, S., Abran, A., Symons, C, (2000) “COSMIC-FFP Some results from the field trials”, in 15th International Forum on COCOMO and Software Cost Estimation

74. Zubrow, D., Can you trust your data? Measurement and Analysis Infrastructure Diagnosis, Presentation, Retrived May 2011 from http://www.sei.cmu.edu/library/assets/meas-infrastructure.pdf

75. 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

76. Wiegers, K.E., (2010) Lessons from Software Work Effort Metrics, retrieved August 2011 from www.processimpact.com/articles/metrics.pdf , 2010

77. Diab H., Koukane F., Frappier M., St-Denis R., (2005), “μcROSE: Automated Measurement of COSMIC-FFP for Rational Rose Real Time”, Information and Software Technology, Volume 47, Issue 3, 1 pp 151-166

78. Dolado, J. J. (2000). "A validation of the component-based method for software size estimation." IEEE Transactions on Software Engineering 26(10): pp 1006-1021

79. Santillo, L., Abran, A., (2006) Software Reuse Evaluation Based on Functional Similarity in COSMIC-FFP Size Components , in Proceedings of the Software Measurement European Forum - SMEF2006

80. Santillo, L., Noce D., (2005) “A Worked Function Point Model for Effective Software Project Size Evaluation”, SMEF 05 Procs.,

81. Meli, R., (2000) –“ Functional and technical software measurement: conflict or integration?” – FESMA

82. Kitchenham, B. , Känsälä, K., (1993) “Inter-Item Correlations among Function Points.” Proc. International Software Metrics Symposium. IEEE Computer Society Press.pp 11-14,

83. Gencel, C. and Buglione, L. (2007). “Do Base Functional Component Types Affect the Relationship between Software Functional Size and Effort?” Proceedings of IWSM/MENSURA, pp 72-85.

84. Buglione, L., Gencel, C.: (2008) “Impact of Base Functional Component Types on Software Functional Size based Effort Estimation”. PROFES 2008. LNCS, vol. 5089, pp. 75–89. Springer,

85. Chen, Z., Menzies, T. , Port, D., and Boehm, B., (2005) “Finding the right data for software cost modeling,” IEEE Software, Nov, 2005, pp. 38-46.

86. Wikipedia-“Software Development Effort Estimation” Retrieved May 2011 from http://en.wikipedia.org/wiki/Software_development_effort_estimation

87. Fenton, N. E., Pfleeger S. L., (1996) “Software Metrics – A Rigorous and Practical Approach”. 2nd. Edition, Thomson Computer Press

88. Kitchenham B., (1990) “Software Development Cost Models”. In: Software Reliability Handbook, P. Rook (editor.), Elsevier Applied Science, pp. 487-517

89. Jorgensen M.,, Boehm, B., Rifkin, S., (2009) “Software Development Effort Estimation: Formal Models or Expert Judgment? IEEE Software, pp: 14-19

90. M. Jorgensen. (2004) “A review of studies on expert estimation of software development effort”. Journal of Systems and Software, 70(1-2):37–60

91. T. De Marco. (1982) Controlling Software Projects. Yourdan Press

92. Hughes, R. T, (1996) “Expert judgement as an estimating method”. Information and Software Technology, vol. 38, no. 2, pp 67-75

93. Hihn, J., H. Habib-agahi (1991), “Cost estimation of software intensive projects: A survey of current practices”, International Conference on Software Engineering, pp. 276-287.

94. Aranda, J. and Easterbrook, S.M. (2005) “Anchoring and adjustment in software estimation”, In Proceedings of ESEC/SIGSOFT FSE. pp 346-355

95. Boehm, B. W. (1984). "Software engineering economics." IEEE Transactions on Software Engineering 10(1): pp 4-21.

96. Linstone, H.A. and M. Turoff, (1975) “The Delphi Method: Techniques and Applications”. Addison-Wesley

97. Kaczmarek, J. and Kucharski, M., (2004) “Size and effort estimation for applications written in Java”. Information and Software Technology. v46. 589-601.

98. Chemuturi, M., “Analogy based software estimation”, Retrieved August 2011 from http://www.chemuturi.com/Analogy%20based%20Software%20Estimation.pdf

99. Shepperd, M., Schofield, C., (1997). Estimating Software Project Effort Using Analogies, IEEE Transactions on Software Engineering, Vol. 23, No. 12 ,pp: 736-743

100. Dasarathy B.V., (1991) Nearest Neighbor (Nn) Norms, IEEE Computer Society Press,Washington

101. Shepperd, M., Schofield, C. , Kitchenham, B., (1996)“Effort estimation using analogy. Proceedings of the 18th international conference on software engineering,

102. Angelis L.,.Mitas, N., (2008) “Combining regression and estimation by analogy in a semiparametric model for software cost estimation”, in ESEM’ 08, pp. 70-79

103. Valerdi R. (2005) "The Constructive Systems Engineering Cost Model (COSYSMO)", PhD Thesis, The University of Southern California

104. Symons, C.R., Rule, P. G., (1999) “One size fits all – COSMIC aims, design principles and progress”, Proceedings of ESCOM ’99, pp. 197-207

105. Abts C., Boehm B., Clark B., (2000) “COCOTS: A COTS software integration cost model”,proceedings of ESCOM-SCOPE 2000 conference,

106. SEER-SEM, Cost Estimating Software, http://www.galorath.com/

107. Colbert, E.,, Yue Chen, D.W, Boehm, B., (2006) “Cost Estimation for Secure Software & Systems”, (Abstract), ISPA,

108. Jensen R. (1983), “An Improved Macrolevel Software Development Resource Estimation Model,” In Proceedings of 5th ISPA Conference, pp. 88–92.

109. 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 . METRICS. IEEE Computer Society,

110. Bajwa, S.S,(2008) : Investigating the Nature of Relationship between Software Size and Development Effort. Master Thesis, MCS-2008-45, School of Engineering, Blekinge Institute

111. Boehm, B., Clark, B., Horowitz, E., Madachy, R., Shelby, R., and Westland C.(1995) "Cost Models for Future Software Life Cycle Processes: COCOMO 2.0", Annals of Software Engineering

112. Fenton, N.E. and Neil, M., (2000) "Software metrics: roadmap". In Proceedings of the Conference on the Future of Software Engineering, pp. 357-370.

113. Walston, C.E., Felix. P.C., (1977) "A Method of Programming Measurement and Estimation". IBM Systems Journal, pp 55-73

114. Moher, T., Schneider, G. M. (1981) Methods for improving controlled experimentation in software engineering. 5th Conference on Software Engineering,

115. Park, R. , (1988) "The central equations of the price software cost model," 4th COCOMO Users Group Meeting.

116. Putnam, L.H., (1978) "A general empirical solution to macro software sizing and estimation problem, IEEE Transactions on Software Engineering",pp 345–361

117. NESMA, (1998) "Function Point Analysis for Software Enhancement, Guidelines, v. 1.0", Online: http://www.nesma.org

118. Capers, J., (1991) Applied Software Measurement, McGraw-Hill

119. Ebert, C., Dumke, R., Bundschuh, M., Schmietendorf, A.(2004)"Best Practices in Software Measurement – How to use metrics to improve project and process performance". Springer-Verlag

120. Whitmire S.A, (1992) "3-D Function Points: Scientific and Real-Time Extensions to Function Points", Proceedings of the Pacific Northwest Software Quality Conference,

121. Abran, A., Maya, M., Desharnais, J.M., St-Pierre, D.,(1997) "Adapting Function Points to Real-Time Software". American Programmer, pp 32-42

122. Forselius, P., (2004) "Finnish Software Measurement Association Functional Size. Finnish Software Metrics Association"

123. Briand, L.C., El Emam, K., Wieczorek. I., (1998) "A Case Study in Productivity Benchmarking: Methods and Lessons Learned", Proc. ESCOM-ENCRESS 98 Project Control For 2000 and Beyond

124.Cruickshank, R. D., Gaffney, J. E., (1992) "A Software Cost Model of Reuse within a Single System," presented at MITRE-Washington Econ. Analysis Ctr. Conf. on Analytical Methods in Software Eng. Econ. II,.

125.Fenton, N., (1991) Software Metrics: A Rigorous Approach. London: Chapman & Hall,

126.Turetken, O., Demirors, O., Gencel, C., Ozcan Top, O., Ozkan, B., (2008) "The Affect of Entity Generalization on Software Functional Sizing: A Case Study", scheduled to be presented in PROFES'08 and published in LNCS, Springer.

127.IEEE Standard 830-1998, Institute of Electrical and Electronics Engineers, Inc., USA. IEEE Recommended Practice for Software Requirements Specifications, .

128.Top, O.O, (2008), "Functional Similarity Impact On The Relation Between Functional Size And Software Development Effort", Msc.Thesis, METU

129.Abran A., Desharnais J.M., (1995) "Measurement of Functional Reuse in Maintenance" in Journal of Software Maintenance: Research and Practice, Vol 7, pp 263-277

130.The Common Software Measurement International Consortium (COSMIC): Guideline for Sizing Business Applications Software Using COSMIC-FFP, Version 1.0 (2005)

131.Abran, A., Maya, M., (1997) "Measurement of Functional Reuse", WISR8, Ohio State University,

132.Leach, R. J., (1996) "Methods of Measuring Software Reuse for the Prediction of Maintenance Effort," Journal of Software Maintenance

133.Ho, V., Abran, A., and Oligny, S., (2000) "Using COSMIC-FFP Quantify Functional Reuse in Software Development" ESCOM-SCOPE 2000

134.Abran A., Gil B., Lefebvre E., (2004) Estimation Models Based on Functional Profiles. International Workshop on Software Measurement -- IWSM/MetriKon, , Shaker Verlag, , pp 195-211.

135.Abran, A., Panteliuc, A., (2007) "Estimation Models Based on Functional Profiles". III Taller Internacional de Calidad en Technologias de Information et de Communications,

136.DO-178 , http://www.highrely.com/do178b_questions.php

137.DOD-2167 "Defence System Software Development", http://www.product-lifecycle-management.com/download/DOD-STD-2167A.pdf

138. Springsteen B., for DOD (1994) ", Survey of Software Metrics in the Department of Defense and. Industry, IDA Paper P-2996,1994, Retrieved May 2011 from http://www.dtic.mil/cgi-bin/GetTRDoc?Location=U2&doc=GetTRDoc.pdf&AD=ADA290804

139.Braungarten, R.; Kunz, M.; Dumke, R., (2005) “An Approach to Classify Software Measurement Storage Facilities”. Preprint No 2, Dept. of Computer Science, University of Magdeburg

140.Song, T. H., Yoon, K. A., Bae, D. H., (2007) “An approach to probabilistic effort estimation for military avionics software maintenance by considering structural characteristics. Asia-Pacific Software Engineering Conference, pp 406–413

141.Greves, D., Schreiber, B., “The ESA Initiative for Software Productivity and Benchmarking” , Retrieved June 2011 from http://www.esa.int/esapub/bulletin/bullet87/greves87.htm

142.ISBSG Software Benchmarking Database, http://www.isbsg.org/

143.Laird, L. M., Brennan, M. C.,(2006) - Software Measurement and Estimation: A Practical Approach (Quantitative Software Engineering Series),Wiley

144.Myrtveit, I., E. Stensrud, et al. (2001). "Analyzing data sets with missing data: An empirical evaluation of imputation methods and likelihood-based methods." IEEE Transactions on Software Engineering 27(11) pp 999-1013.

145.Rios D., Neuro AI - Intelligent systems and Neural Networks, Retrieved May 2011 from http://www.learnartificialneuralnetworks.com/

146.Zhang, G., Patuwo, B.E. and Hu, M.Y. (1998). Forecasting with Artificial Neural Networks: The State of the Art. International Journal of Forecasting, pp 35-62

147.Heaton, J., “Introduction to Neural networks for Java, Second Edition”, Heaton Research Inc., Retrieved February 2011 from http://www.heatonresearch.com/node/707

148.Shukla, K.K.,(2000) “Neuro-genetic prediction of software development effort”. Information and. Software Technology 42(10), 701–713

149.Blanco, A., Delgado, M., Pegalajar M. C.,(2000) “A genetic algorithm to obtain the optimal recurrent neural network” Int. Journal of approximate reasoning vol.23

150.Wang, J., (1994) “A Neural Network Approach to Multiple Criteria Decision Making Based on Fuzzy”. Preference Information. Information Sciences,

151.Rumelhart, D.E., Hinton, G.E., Williams, R.J., (1986) “Learning internal representations by Error Propagation” Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Cambridge, MA., MIT Press. Volume 1, pp. 318- 362.

152.Hassoun, M.H., (1995), “Fundamentals of Artificial Neural Networks”, The MIT Press

153. Werbos, P. J. (1974). “Beyond regression: New tools for prediction and analysis in the behavioral sciences”. Ph.D.Thesis, Harvard University

154. Silvert, W. ,Baptist, M., (1998), ”Can Neural Networks be used in Data-Poor Situations?' in Artificial Neuronal Networks: Application to Ecology and Evolution”, Eds S. Lek & J. Guegan, Springer-Verlag, Berlin, pp. 241-248.

155. IEEE EIA 12207, Software Life Cycle Processes, http://www.12207.com/

156. Söderholm, A., (2007) “Project management of unexpected events; in: International Journal of Project Management”, Vol. 26

157. Tunalilar, S., Demirors O.,(2011) “An Exploration of Functional Size Based Effort Estimation Models”, International Journal of Software Enginering and Knowledge Engineering, 2011

158. April, A., Abran, A., Dumke, R., (2004) “Software maintenance productivity measurement: How to assess the readiness of your organization”. Proceedings International Workshop on Software Metrics and DASMA Software Metrik Kongress

159. International Function Point User Group (IFPUG): (1994) “Function Point Counting Practices Manual-Release 4.0, Westerville: IFPUG Inc. <http://www.ifpug.org/>

160. Tunalilar, S., Demirors, O.,(2009) “A Comparison of Neural Network Model and Regression Model Approaches Based on Sub-functional Components, IWSM '09 /Mensura '09 Proceedings of the International Conferences on Software Process and Product Measurement

161. Morris, P., (2006), “COSMIC-FFP – A Method for Sizing All the Software Not Just What the User Sees”, in ‘DOD Software Tech News’, Editor: Data & Analysis Center for Software, Vol. 9 No.3, http://www.compaid.com/caiinternet/ezine/morris-cffp.pdf

162. Yin, R.K. (1994). Case Study Research: Design and Methods, Applied Social Research Methods Series, Vol.5, 2nd ed., Sage Publications Inc.

163. Stake, R. (1995). The art of case study research. Thousand Oaks, CA: Sage Publications

164. Brereton, P., Kitchenham, B. , Budgen, D., Li, Z., (2008) “Using a protocol template for case study planning”, 12th International Conference on Evaluation and Assessment in Software Engineering (EASE'08),

165. Rowley, J. (2002), “Using case studies in research”, Management Research. News, vol. 25, no. 1, pp. 16-27.

166. Lother, M., Dumke, R.R, (2001). Points Metrics - Comparison and Analysis, In Proceedings of the International Workshop on Software Measurement (IWSM'01), Montréal, Québec, pp. 155-172.

167. Boehm, B., Horowitz, E., Madachy E.,, Reifer, D., Clark, B., Steece, B., Brown, W., Chulani, S., Abts, C., (2000) “Software Cost Estimation with COCOMO II. Prentice Hall”

168. Knowledge Structures Inc., Estimating Guideline, Retrieved June 2011 from http://www.ksinc.com/itpmcptools/ EstimatingGuidelines.pdf

169. Strike, K., El Emam, K., Madhavji, N.H., (2001) “Software cost estimation with incomplete data”. IEEE Transactions on Software Engineering

170. Baker, D., (2007) “A hybrid approach to expert and model-based effort estimation,” Master’s thesis, Lane Department of Computer Science and Electrical Engineering, West Virginia University, 2007, Retrieved May 2011 from https://eidr.wvu.edu/etd/documentdata.eTD?documentid=5443

171. Bisio, R., Malabocchia, F., (1995)“Cost Estimation of Software Projects through Case Based Reasoning. Case Based Reasoning Research and Development”, Proc. International Conference on Case-Based Reasoning, pp 11-22

172. Molokken-Ostvold, K., Jorgensen, M., (2004)“Group processes in software effort estimation. Empirical Softw. Eng., 9(4):pp 315–334,

173. Marín, B., Fernández, N.C, Pastor, O., (2008) Towards a Method for Evaluating the Precision of Software Measures (Short Paper). QSIC

174. Niessink, F.,Vliet, H.van,, (2001) “Measurement Program Success. Factors Revisited. Information and Software Technology”,. 43(10):pp 617–628

175. Song T.H., Yoon, K.A, Bae, D.H, (2007) An Approach to Probabilistic Effort Estimation for Military Avionics Software Maintenance by Considering Structural Characteristics. APSEC

176. Yoon K A, Bae D H, Seo Y S, (2008) “An Empirical analysis of software effort estimation with outlier elimination” in Proceedings International Conference on Software Engineering

177. Chan V., and Wong, W., (2007) “Outlier elimination in construction of software metric models. Proceedings of the 22nd ACM Symposium on Applied Computing, pp 1484–1488

178. Garmus, D., (2006) “The Principles of Sizing and Estimating Projects Using IFPUG Function Points," Software Tech News

179. ISO/ IEC 15939-2002 software measurement process model. Retrieved June 2011 from http://segoldmine.ppi-int.com/content/standard-isoiec-15939-software-measurement-process

180. CUBIT, (2011) Cosmic Database of Middle East Technical University, Software Engineering Research Group, available at http://smrg.ii.metu.edu.tr/cubit

181. DataFit, “Curve Fitting and Data plotting Tool” , http://www.oakdaleengr.com/datafit.htm

182. SPSS Statistics Ver 19. www-01.ibm.com/software/analytics/spss/statistics

183. Tunalilar, S., Demirors, O., (2008) “Effect of Functional Similarity for Establishing Relation Between Effort and Functional Size, Asia Pasific Software Engineering Conference, SPACE special session

184. Top O.O, Tunalilar S., Demirors O., (2008) “Evaluation of the Functional Similarities on Development Effort, Euromicro Conference on Engineering and Advanced Applications,

185. CMMI, “Capability Maturity Model Integrated, (2011), Software Engineering Institute www.sei.com.tr/cmmi

186. Cosmic Functional Size Measurement Version 3.0 Retrieved June 2011 from http://www.cosmicon.com/portal/public/COSMIC%20Method%20v3.0%20Advanced%20&%20Related%20Topics.pdf

APPENDICES

1. DECOMPOSITION OF SOFTWARE PROJECT INTO SCIs

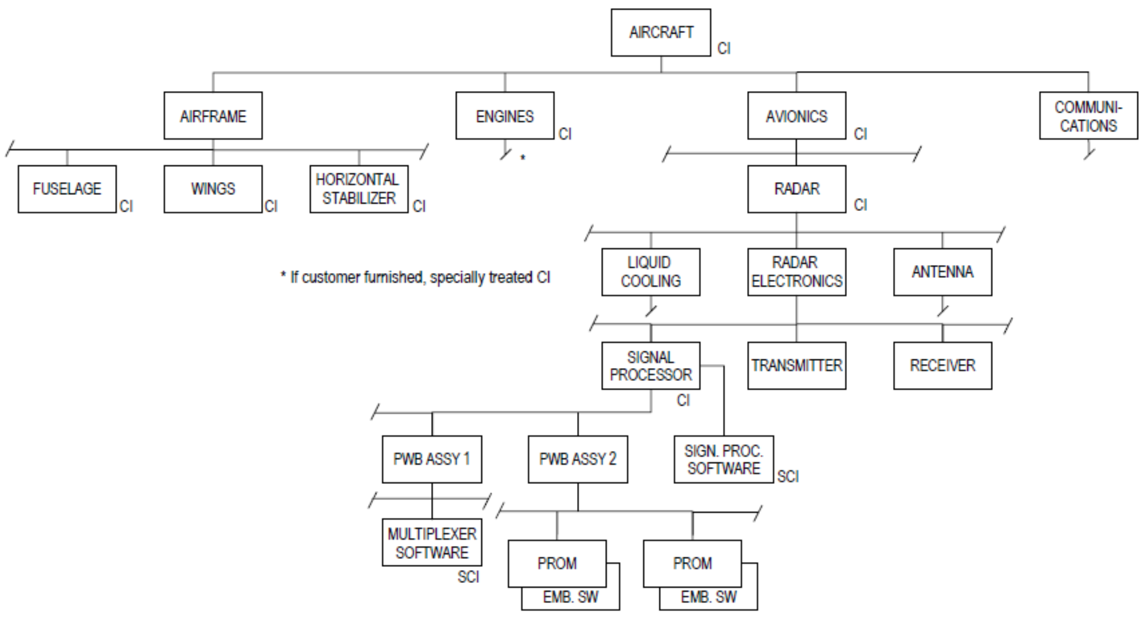


Figure 28 Example Decomposition of Software Project from study European Space Agency, [15]

APPENDIX

2. LIST EXAMPLES

Table 31 General List: Phase and Activity Groupings in Company

Phases and Tasks	**Activities**
System requirements analysis	(*Included in meetings with Customers and Working with System Engineers to understand functionalities*)
System architectural design	(*Included as meetings for SW developers but can be added for System Design Team*)
Software requirements analysis	-Preparation of prototypes to specify the requirements of the system -Working with System Engineers to understand functionalities
Software architectural design	-Preliminary Design Activities
Software detailed design	-Detailed Design Activities
Software coding and unit testing	-Coding/Updating the Program -Writing and executing unit tests
Software Test	- Interface testing -Integration at SW level -Integration at HW Level
Software qualification testing	-Coverage Testing -Qualification of Software Component - *Software Safety Assessment Testing*
System integration	-Integration at System Level -Integration at Platform Level
System qualification testing	-System Level testing -Test According to a standard -Platform level testing -Defect Removal During Qualification -*Safety Validation at System Level*
Software installation	-Installation Time on Different Platforms
Software acceptance support	-Tests with Customers -Defect Removal during acceptance

Table 32 Additional Phase and Activity Definitions for SAFETY to ISO-IEC 15504 :

Phase	**Activity**
Test	Software Safety Assesment Testing Safety Validation at System Level
Support	Overall management of Safety Requirements Qualification of software tools Qualification of Software Components Qualification of hardware components

Table 33 General List Example Phase Definitions from Literature

Chinese [75]	**Phases in 12207 [155]**	**Wiegers[76]**
Plan, Preliminary Requirement Analysis Requirement Analysis Design Product Design, Detailed Design Code Code, Unit Test, Integration Test System Test Transition Installation, Transition, Acceptance Test, User Training, Support	System requirements analysis System architectural design Software requirements analysis Software architectural design Software detailed design Software coding and testing Software integration Software qualification testing System integration System qualification testing Software installation Software acceptance support	Preliminaries Project Planning, Specification, Design, Implementation, Testing, Writing documentation. Adaptive maintenance, Fixing bugs, Adding Enhancements, User support

Table 34 Activity Definitions Recommended by McConnell

(From study [26])

Functional Requirements Area	Setup/installation program Data conversion utility Glue code needed to use third party software or open source software Help System Deployment models Interfaces with external systems
Non Functional Requirements Area	Accuracy Interoperability Modifiability Performance Portability Reliability Responsiveness Reusability Security Survivability Usability Software Development Activities Management coordination/manager meetings Maintaining the revision control system Supporting the build Maintaining the scripts required to run the daily build Maintaining the automated smoke test used in conjunction with the daily build Installation of test build at user locations Participation in technical reviews Integration work Processing Change requests …. Input to user documentation and review of user documentation Demonstrating software to customers or users Demonstrating software at trade shows Demonstrating the software or the prototype to upper management, client and end users Interacting with client or end users; supporting beta installations at client locations Reviewing plans, estimates, architecture, detailed designs. stage plans, code, test cases and so on

APPENDIX 1

3. SUPPORTING & EXTRA EFFORT LIST

Table 35Applied Supporting & Extra effort List (Unexpected/Unplanned/Unrelated activities)

Supporting & Extra effort	**Explanation**	**Effect Level**
Unavailability of developer	Include the wasted time in hours for project, if resource allocation is not done	App.Domain Type
Hardware problem,	Include the wasted time in hours for each SCI, if HW not available or HW has problem	App.Domain Type
Specific Standard Appliance	Include extra expended effort not planned for each SCI	App.Domain Type
Requirement Changes	Include extra expended effort not planned for each SCI	App.Domain Type
Support to Other Projects	Record support time not wasted for current project	App.Domain Type
Demo	Include this unplanned Effort	App.Domain Type
12207/OrganizationalProcess Planning/Execution	-Time spent on planning and tracking activities throughout the project's life. -Evaluating and selecting tools, computers, operating systems, and so on for a specific project.	Project
12207/Organizational Process Training	-Trainings required for the usage of tools -Trainings required for the understanding of the product	App.Domain Type
12207/Organizational Process-Infrastructure	-Evaluating and selecting tools, computers, operating systems, and so on for a specific project.	App.Domain Type
12207/SupportingProcess-Documentation	Writing and inspecting the software requirements specification. Writing and inspecting the software design specification. Writing and inspecting the software test specification. Writing technical note or giving presentations	App.Domain Type, collect in seperate phases
12207/SupportingProcess-JReviews,ProblemResolution, Audits, Other Meetings	-Peer reviews -Meetings with Other Team Members -Meetings with Customers -Specially arranged Design Reviews (ex:PDR, CDR) -Duties	Project
12207/SupportingProcess-Configuration Management	Configuration Management (CM) Activities CM Personnel CM Activities of SW Development Personnel	Project
12207/SupportingProcess-Quality Assurance	QA Activities of QA Personnel	Project

Table 36 Example Lists for Supporting & Extra efforts (Unexpected Event/Out Of Development)

From Company Reviews, 2009	**From Study So¨derholm,2008[156]**
Unavailability of developer	Delivery Delay of Subcontractors
Hardware problem,	Organizational Change
Specific procedure appliance	
Requirement Changes	
Support to Other Projects	
From McConnel[26],2006	**From Study Alain April et al[158],2004**
Vacations	Optimizing code and resources;
Holidays	Restructuring code logic;
Sick Days	Clarifying and improving system docs.;
Training	Minor functionality enhancements.
Weekends	Compiler and utility changes;
Company Meetings	· Hardware upgrades;
Department Meetings	· Media conversions;
Setting up new work stations	· Making adjustments to accommodate
Installing new ver.of tools on workstations	changes in load;
Troubleshooting hw and sw problems	Evolving the System Recovery Manual

APPENDIX 2

4. EFFORT COLLECTION PROCESS TABLE EXAMPLES

Table 37 Example Effort Collection Table for Project1__ES_ControlInterface

Phase	**Activity**	**Assigned Developer**	**Planned Effort(day)**	**Real Effort (day)**	**Completeness**
Coding&UnitTest	WriteVideo Blocks and UnitTest	T.K	10	9	%100
Integration&Qual	Video Interface Test	T.K.	6	7	%100
Integration&Qual	Serial Communication Test-RS232	S.Y	8	3	%40
Hw Problem	-	-	-	23	-
Requirement Chg.	-	-	-	17	-
Support	Overall management of Safety Requirements	A.Y	(New-40)	11	% 35

APPENDIX 3

5. APPLICATION DOMAIN CATEGORIZATION

Table 38Applied Categorization of Company

Embedded Real-time SW
GUI and Simulation
Board Support Package
Algorithm Development

Table 39 Application Domain Categorization in Literature

Desharnais[73]	**DeMarco[91]**	**Brungarten & Kunz &Dumke[139]**
Business Software	Data Strong Systems	Microcode/Firmware
Embedded & Control Software	Control Strong Systems	Real time,
Utility Software	Function Strong Systems	Avionic,
User's Tool Software	Hybrid Systems	System Software,
Developer's Tool Software		Command & Control,
Systems Software		Telecom/Message
		Switching,
		Scientific,
		Process control,
		Business/Commercial

APPENDIX 4

6. SIZE MEASUREMENT PROCESS TEMPLATES

Table 40 SRS Template Example

SRS Template (Adapted from IEEE-830)
1. Introduction 1.1 Purpose 1.2 Document conventions 1.3 Intended audience 1.4 Additional information 1.5 Contact information/SRS team members 1.6 References **2. Overall Description** 2.1 Product perspective 2.2 Product functions 2.3 User classes and characteristics 2.4 Operating environment 2.5 User environment 2.6 Design/implementation constraints 2.7 Assumptions and dependencies **3. External Interface Requirements** 3.1 User interfaces 3.2 Hardware interfaces 3.3 Software interfaces 3.4 Communication protocols and interfaces **4. System Features** 4.1 System feature A 4.1.1 Description and priority 4.1.2 Action/result 4.1.3 Functional requirements 4.2 System feature B **5. Other Non functional Requirements** 5.1 Performance requirements 5.2 Safety requirements 5.3 Security requirements 5.4 Software quality attributes 5.5 Project documentation 5.6 User documentation **6. Other Requirements** Appendix A: Terminology/Glossary/Definitions list Appendix B: To be determined

Table 41 SRS Review Checklist Template Example

A sample checklist for SRS Review		**Status**
1.1.	Are all the requirements verifiable?	Yes
1.2.	Are the requirements complete?	Yes
1.3.	Are all the requirements clear to measurers ?	Check Req12, 18, 67
1.4.	Are all the requirements stated only once?	
1.5.	Are all the requirements broken down into their most elementary form?	Yes Chech Req 5
1.6.	Are all the requirements can be represented with functional processes with triggering events, data groups, data movements?	Yes
1.7.	Is there any information to be necessary to be included?	

Table 42 Measurement Results Template Example

Measurement Results Template from [29]
1. Overview 1.1 Introduction 1.2 Measurement viewpoint, purpose and scope 2. Requirements as documented in ISO 14143-3-4 : 2000 2.1 Context 2.2 Input 2.3 Output 3. COSMIC-FFP measurement procedure 3.1 Identification of layers 3.2 Identification of users 3.3 System boundary 3.4 Identification of triggering events 3.5 Identification of data groups 3.6 Identification of functional processes 4. Identify data movements 4.1 Message sequence diagram 4.2 List of data movements 4.3 Observations on the clarity of the requirements 5. Analysis of measurement results

Table 43 Detailed Measurement Results Template Example

Project-1 Measurements

Process Description	**Subprocess-Decsription**	**Data Group**	**Cosmic-Data Movement**	**No FS**	**FS Reflected**
Press-A	Read Letter Write Letter Send.Channel	LetterA	E X X	3	3
Press-B	Read Letter Write Letter Send.Channel	LetterB	E X X	3	0

Table 44 BFC Grouping Results of all Projects Template Example

Project name	**Standard Measurement**				**FS Reflected Measurement**				**Effort (in hour)**
	# of Entry	# of Exit	#of Read	#of Write	# of Entry	# of Exit	#of Read	#of Write	-
Project-1	57	73	236	222	27	21	157	159	1020
Project-2	226	745	174	247	55	84	40	29	396
Project-3	58	41	149	164	49	32	145	148	476

APPENDIX 5

7. FUNCTIONAL SIMILARITY APPLICATION NOTES:

- Basic- Zero Effort FS method

As it is given in in "Detailed Measurement Results", If two functional process has subprocesses that are completely same, then only one of them will be included in size measurement, others' effect on effort are ignored.

Table 45 FS Measurements Calculation

Functional Process		**Sub-Process**	**Data Movement**	**No FS**	**FS (Zero Effort)**	
Select NOT		StartSelectNot	E	1	1	
		ShutRemote	W	1	1	
		WriteRemote	W	1	1	
NearMode		StartNear	E	1	1	
		WriteNear	W	1	1	
		SendNearData	X	1	1	
WideMode		StartWide	E	1	0	Zero Effort
		WriteWide	W	1	0	
		SendWideData	X	1	0	
NarrowMode		StartNarrow	E	1	0	Zero Effort
		WriteNarrow	W	1	0	
		SendNarrowData	X	1	0	

APPENDIX 6

8. DATA ANALYSIS PROCESS EXAMPLE TEMPLATES

Table 46 #of BFC Table for Analysis of Standard Measurement

	# of Entry	# of Exit	#of Read	#of Write	Effort In Hours
Project1-SCIName2	57	73	236	222	1020
Project2-SCIName3	226	745	174	247	396
Project6-SCIName5	58	41	149	164	476

Table 47 Accuracy Record Example Template for Neural Network

	Application Type= BSP Other info…. **# of Projects = 15**								
		No FS				**FS Reflected**			
	Iteration No	**R2**	**MMRE**	**PRED (0.30)**	**PRED (0.20)**	**R2**	**MMRE**	**PRED (0.30)**	**PRED (0.20)**
	100								
	500								
	1000								
	..								

Table 48 Effort Model Weights Table

Model Name	Weight for Entry	Weight For Exit	Weight For Read	Weight For Write	Addition
SCI_A	121,05	-12,33	32,97	277,52	113
SCI_B	87,11	35,62	-431,20	5,63	12

Table 49 Supporting & Extra effort Analysis Results Example table

	Requirements Change	**Meetings**	**Hardware Problems**	**Uplanned Activities**	**Documentation**
GUI Applications	%8,92	%10	%2,81	%23	%20,34
BSP Applications	%1,39	%21	%13,6	%3	%24,98
ES Applications	% 3,16	%7	%10,81	%5	%10,63

Table 50 Effort Estimation of new projects Example table

	Size Measurement (E,X,R,W)	**Base Effort Value (man-hour)**	**HW Problems.**	**Documentation**	**Req. Change**	**Total Effort**
Prj_New_GUI-1	23,12, 9, 35	828	2,81	20,34	20,34	1125
Prj_New_GUI-2	234,31,112,39	216	2,81	24,98	20,34	146
Prj_New_ES_1	% 3,16	1081	10,81	10,63	10,63	217

VITA

Seçkin TUNALILAR was born in Bandırma, Turkey. She received her bachelor degree in Electrical and Electronics Engineering in Middle East Technical University (METU) in 1995. In 1998, she had her M.S. degree in the same department. During 1995 and 2011, she worked in Aselsan and participated in number of projects. She worked as a part time lecturer at Defense Sciences Institute of Turkish Military Academy. Now she's working as a program manager for R&D projects of Airborne platforms. Previously she lead business development activities for Naval programs, design activities of Electro-optical Imaging systems and Tank Fire control systems. Her research interests include; project management, system design, software process improvement and software measurement

Journal & International Conferences:

- Tunalılar, S., Demirors O.,(2011) "An Exploration of Functional Size Based Effort Estimation Models", International Journal of Software Enginering and Knowledge Engineering, 2011
- Tunalilar, S., Demirors, O.,(2009) "A Comparison of Neural Network Model and Regression Model Approaches Based on Sub-functional Components, IWSM '09 /Mensura '09 Proceedings of the International Conferences on Software Process and Product Measurement
- Tunalılar, S., Demirors, O., (2008) "Effect of Functional Similarity for Establishing Relation Between Effort and Functional Size, Asia Pasific Software Engineering Conference, SPACE special session
- Top, O.O, (2008), "Functional Similarity Impact On The Relation Between Functional Size And Software Development Effort", Msc.Thesis, METU

National Conferences:

- Aktürk, D., Yağcıoğlu, M., Tunalilar S.,(2010) "Kafa Takip teknolojileri", "Aviyonik ve Sistem Entegrasyonu Sempozyumu"
- Top, O.O. , Tunalilar, S., Demirors, O., (2008) "Fonksiyonel Benzerlik ve İş gücü: Bir durum çalışması" , Yazılım Kalitesi ve Yazılım Geliştirme araçları Sempozyumu
- Tunalilar, S., Demirors O.,(2005) "Yazılım Süreç İyileştirmede Başarı Faktörleri" II. Ulusal Yazılım Mühendisliği Sempozyumu