

A UNIFICATION MODEL AND TOOL SUPPORT FOR
SOFTWARE FUNCTIONAL SIZE MEASUREMENT METHODS

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

BY

PINAR EFE

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF SCIENCE

IN

THE DEPARTMENT OF INFORMATION SYSTEMS

JUNE 2006

Approval of the Graduate School of Informatics

Assoc. Prof. Dr. Nazife BAYKAL
Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of Master of Science.

Assoc. Prof. Dr. Yasemin YARDIMCI
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 Master of Science.

Dr. Çiğdem GENCEL
Co-Supervisor

Assoc. Prof. Dr. Onur DEMİRÖRS
Supervisor

Examining Committee Members

Prof. Dr. Semih BİLGİN (METU, EE) _____

Assoc. Prof. Dr. Onur DEMİRÖRS (METU, IS) _____

Dr. Çiğdem GENCEL (METU, IS) _____

Dr. Ali ARİFOĞLU (METU, IS) _____

Vedat USLU (MBA) (SIEMENS) _____

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name and Surname: _____

Signature:

ABSTRACT

A UNIFICATION MODEL AND TOOL SUPPORT FOR SOFTWARE FUNCTIONAL SIZE MEASUREMENT METHODS

Efe, Pınar

M.S., Department of Information Systems

Supervisor: Assoc. Prof. Dr. Onur Demirörs

Co-Supervisor: Dr. Çiğdem Gencel

June 2006, 126 pages

Software size estimation/measurement has been the objective of a lot of research in the software engineering community due to the need of reliable size estimates. FSM Methods have become widely used in software project management to measure the functional size of software since its first publication, late 1970s. Although all FSM methods measure the functional size by quantifying the FURs, each method defines its own measurement process and metric. Therefore, a piece of software has several functional sizes when measured by different methods. In order to be able to compare functional sizes of software products measured by different methods, we need to convert them to each other.

In this thesis study, the similarities and differences between four FSM methods, IFPUG FPA, Mark II FPA, COSMIC FFP and ARCHI DIM FSM are investigated and the common core concepts are presented. Accordingly, a unification model of the measurement process of all four methods is proposed. The main

objective of this model is to measure the functional size of a software system by applying all four methods simultaneously, using a single source of data. In order to have an infrastructure to validate the unification model by conducting empirical studies, a software tool is designed and implemented based on the unification model. Two empirical studies are conducted by utilizing the data of a real project to evaluate both the unification model proposed and the developed tool and the measurement results are discussed.

Keywords: Functional Size Measurement, IFPUG FPA, Mk II FPA, COSMIC FFP, Convertibility of FSM Methods

ÖZ

YAZILIM FONKSİYONEL BÜYÜKLÜK ÖLÇME YÖNTEMLERİ İÇİN BİRLEŞTİRME YÖNTEMİ VE YÖNTEMİ DESTEKLEYEN BİR ARAÇ

Efe, Pınar

Yüksek Lisans, Bilişim Sistemleri

Tez Yöneticisi: Doç. Dr. Onur Demirörs

Ortak Tez Yöneticisi: Dr. Çiğdem Gencel

Haziran 2006, 126 sayfa

Güvenilir büyüklük ölçümlerine duyulan ihtiyaç nedeniyle, yazılım büyüklük kestirme/ölçme yöntemleri yazılım mühendisliği dünyasında bir çok araştırmanın konusu olmuştur. Fonksiyonel Büyüklük Ölçme (FBÖ) yöntemleri 1970'lerin sonlarında ilk yayınlanmasından bu güne kadar yazılımın fonksiyonel büyüklüğünü ölçmek için yaygın olarak kullanılmıştır. Bütün FBÖ yöntemleri fonksiyonel kullanıcı gereksinimlerine dayalı ölçüm yapmasına rağmen, her biri farklı ölçme süreçleri ve metrikleri tanımlamaktadır. Dolayısıyla, aynı yazılım farklı yöntemlere göre farklı büyüklüklerde olabilmektedir. Bu büyüklüklerin karşılaştırılabilmesi için yöntemlerin birbirlerine çevrilebilmesi gerekmektedir.

Bu tezde dört farklı FBÖ yöntemi olan IFPUG FPA, Mk II FPA, COSMIC FFP ve ARCHI DIM FSM arasındaki benzerlikler ve farklılıklar incelenmiştir ve ortak temel kavramlar sunulmuştur. Bu dört yöntem için bir birleştirme modeli önerilmiştir. Modelin temel amacı bir uygulamanın fonksiyonel büyüklüğünü, bu

dört yöntem ile aynı anda yalnızca bir veri setini kullanarak ölçmektir. Bu modeli kullanan bir araç tasarlanmış ve geliştirilmiştir. Gerçekleştirilmiş iki projenin verileri ile geliştirilen araç kullanılarak, aracın ve yöntemin değerlendirilmesi yapılmış; elde edilen ölçüm sonuçları karşılaştırılmıştır.

Anahtar Kelimeler: Fonksiyonel Büyüklük Ölçme, IFPUG FPA, Mk II FPA, COSMIC FFP, Yazılım Büyüklük Ölçme Yöntemlerinin Birbirlerine Çevrimi

ACKNOWLEDGMENTS

I express my sincere thanks to my supervisor Onur Demirörs for his guidance, support, encouragement, endless patience and stimulating suggestions during this study.

I am indebted to my co-supervisor Çiğdem Gencel, who read my numerous revisions, helped make some sense of the confusion and gave her assistance, support and collaboration.

I would like to express my gratitude to my directors in Siemens A.Ş., Erdem Alptekin, Vedat Uslu and Tunca Gerdaneri for their support and helps during this study.

Many thanks to Ahmet Dikici, who endured this long process with me, for his continuous encouragement, infinite patience and moral support.

I also would like to express my deepest appreciation to my mother, father and sister who always gave me their love, understanding, patience and emotional support. They always gave me the courage and strength I needed to complete my goals.

TABLE OF CONTENTS

SELF DECLARATION AGAINST PLAGIARISM.....	iii
ABSTRACT.....	iv
ÖZ.....	vi
ACKNOWLEDGMENTS	viii
TABLE OF CONTENTS.....	ix
LIST OF TABLES	xii
LIST OF FIGURES	xiv
LIST OF ABBREVIATIONS.....	xvi
CHAPTER	
INTRODUCTION	1
1.1. Background	1
1.2. Problem Statement	3
1.3. The Approach.....	5
1.4. Thesis Structure.....	6
RELATED RESEARCH	8
2.1. Software Size Measurement/Estimation Methods	8
2.2. FSM Processes of Applied FSM Methods	17
2.2.1. IFPUG FPA, Release 4.1.1	17
2.2.2. Mk II FPA, Version 1.3.1	20
2.2.3. COSMIC FFP Version 2.2.....	23
2.2.4. ARCHI DIM FSM	25
2.3. Tool Support for Functional Size Measurement	27
2.3.1. Tools Survey.....	29
2.3.2. Comparison of the Existing Tools	32
UNIFICATION MODEL FOR FSM METHODS	41
3.1. Common Concepts in Selected FSM Methods	42

3.1.1. Definitions of Common Concepts	42
3.1.2. Terminology Mapping	44
3.1.3. Constituent Parts of Methods.....	44
3.2. Data Type Concept.....	47
3.2.1. Data Group Concept	47
3.2.2. Data Element Type Concept.....	49
3.2.3. Sub-group Data Concept	50
3.3. Transaction Concept.....	51
3.4 Summary of the Unification Model	55
TOOL ANALYSIS AND DESIGN	58
4.1. Tool Overview.....	58
4.2. Product Perspective	59
4.3. Constraints, Assumptions and Dependencies.....	60
4.4. Specific Requirements.....	60
4.4.1 Project Operations	61
4.4.2 Requirement Operations	64
4.4.3 Data Group Operations	66
4.4.4 Transaction Operations.....	69
4.4.5 Counting Operations.....	73
4.4.6 Reporting Operations.....	75
4.4.7 Estimation Effort Operations.....	76
4.5. Logical Database Requirements	77
4.6. Decomposition Description.....	79
EMPIRICAL STUDIES.....	82
5.1. Case Study.....	82
5.1.1. Project Characteristics	82
5.1.2. Manual Size Measurement of the Case Project	83
5.1.3. Size Measurement of the Case Project Size using EasyEstimate	86
5.1.4. Discussion of the Case Study Results.....	93
5.2. Experimental Study	94
5.2.1. Previous Measurement Results.....	96
5.2.2. Subsystem Measurement Results by EasyEstimate.....	97
5.2.3. Discussion of the Results.....	97

CONCLUSION AND FUTURE DIRECTIONS	101
6.1 Conclusion.....	101
6.2 Future Work	104
REFERENCES	106
APPENDICES	
A ADJUSTMENT FACTORS.....	110
B EASYESTIMATE SCREEN SHOTS	112
C REPORTS FROM EASYESTIMATE	124

LIST OF TABLES

Table 1 FSM Methods	11
Table 2 Complexity Matrix for ILF/EIF	18
Table 3 Complexity Matrix for EI	19
Table 4 Complexity Matrix of the EOs or EQs	19
Table 5 Complexity Weights for ILF/EIF/EI/EO/EQ.....	20
Table 6 Examined Size Estimation Tools	28
Table 7 Checklist for the Comparison of the FSM Tools	33
Table 8 Comparison of Examined FSM Tools	34
Table 9 Terminology used in the selected FSM methods.....	45
Table 10 Different constituent parts and classifications for the selected FSM methods	46
Table 11 The mapping of the constituent parts of IFPUG FPA, Mk II FPA and COSMIC FFP Methods to the unification model	56
Table 12 Mapping of the constituent parts of ARCHI DIM FSM Method to the unification model	57
Table 13 Software Products Required Implementing the Software.....	60
Table 14 Case Project IFPUG FPA Size Measurement Details.....	84
Table 15 Case Project Mk II FPA Size Measurement Details	84
Table 16 Case Project COSMIC FFP Size Measurement Details	84
Table 17 Case Project ARCHI DIM FSM Size Measurement Details	85
Table 18 Case Project IFPUG FPA Size Measurement Results by EasyEstimate	93
Table 19 Case Project Mk II FPA Size Measurement Results by EasyEstimate.....	93
Table 20 Case Project COSMIC FFP Size Measurement Results by EasyEstimate .	93
Table 21 Case Study Results Comparison	94
Table 22 Case Project ARCHI DIM FSM Size Measurement Results by EasyEstimate	95

Table 23 Mk II FPA Size Measurement Results of Subsystem.....	96
Table 24 COSMIC FFP Size Measurement Results of Subsystem	96
Table 25 Mk II FPA Size Measurement Results of Subsystem by EasyEstimate	97
Table 26 COSMIC FFP Size Measurement Results of Subsystem by EasyEstimate	97
Table 27 Experimental Study Results Comparison	98
Table 28 ARCHI DIM FSM Size Measurement Results of Subsystem	99
Table 29 ARCHI DIM FSM Size Measurement Results of Subsystem by EasyEstimate	100
Table 30 IFPUG FPA Value Adjustment Factor Calculation Table.....	110
Table 31 Mk II FPA Technical Complexity Adjustment Factor Calculation Table	111

LIST OF FIGURES

Figure 1 Use Case Diagram for Project Operations.....	61
Figure 2 Use Case Diagram for Requirement Operations	64
Figure 3 Use Case Diagram for Data Group Operations	66
Figure 4 Use Case Diagram for Transaction Operations	71
Figure 5 Use Case Diagram for Counting Operations	74
Figure 6 Use Case Diagram for Reporting Operations	75
Figure 7 Use Case Diagram for Estimation Effort Operations	76
Figure 8 E-R Diagram of EasyEstimate.....	78
Figure 9 Packages of the tool.....	79
Figure 10 Class Diagram of EasyEstimate	81
Figure 11 The Project List and Definition	87
Figure 12 Data Group List and Definition.....	88
Figure 13 DET List and Definition	88
Figure 14 Import of Transactions from CaseStudy.csv file	89
Figure 15 Transaction List.....	89
Figure 16 Input Part of Transaction (all four FSM methods selected)	90
Figure 17 Processing Part of Transaction (all four FSM methods selected)	90
Figure 18 Output Part of Transaction (all four FSM methods selected).....	91
Figure 19 Assigning Data Group to the Transaction (all four FSM methods selected).	91
Figure 20 Assigning DET to the Transaction	92
Figure 21 Measurement Results.....	92
Figure 22 Main Menu and Project Sub Menu.....	112
Figure 23 Main Menu and Estimation Sub Menu.....	112
Figure 24 Main Menu and Estimation Effort Sub Menu	113
Figure 25 Version Entrance Dialog	113

Figure 26 List of Requirements Screen	114
Figure 27 Transaction Link with Requirement Screen	115
Figure 28 Transaction Input Part (only IFPUG FPA and Mk II FPA selected).....	115
Figure 29 Transaction Processing Part (only IFPUG FPA and Mk II FPA selected)	116
Figure 30 Transaction Output Part (only IFPUG FPA and Mk II FPA selected)....	116
Figure 31 Assigning Data Group to the Transaction (only IFPUG FPA and Mk II FPA selected)	117
Figure 32 Transaction Input Part (only COSMIC FFP selected).....	118
Figure 33 Transaction Processing Part (only COSMIC FFP selected).....	118
Figure 34 Transaction Output Part (only COSMIC FFP selected)	119
Figure 35 Assigning Data Group to the Transaction (only COSMIC FFP selected)	119
Figure 36 Transaction Input Part (IFPUG FPA, Mk II FPA and COSMIC FFP selected)	120
Figure 37 Transaction Processing Part (IFPUG FPA, Mk II FPA and COSMIC FFP selected)	120
Figure 38 Transaction Output Part (IFPUG FPA, Mk II FPA and COSMIC FFP selected)	120
Figure 39 Transaction Data Group Conversions Screen.....	121
Figure 40 Confirmation Dialog for Transaction Data Group Conversion Screen ...	121
Figure 41 Value Adjustment Factor Screen	122
Figure 42 Calculated Value Adjustment Factor Dialog.....	122
Figure 43 Estimation Effort Screen	123
Figure 44 IFPUG FPA Report	124
Figure 45 Mk II FPA Report.....	124
Figure 46 COSMIC FFP Report	125
Figure 47 ARCHI DIM FSM Report	125
Figure 48 Transaction Detailed Report.....	126

LIST OF ABBREVIATIONS

AFPI	Adjusted Function Point Index
BFC	Base Functional Component
COSMIC	Common Software Measurement International Consortium
CPM	Counting Practices Manual
CRUDL	Create, Read, Update, Delete, List
DBMS	Database Management System
DET	Data Element Types
EI	External Inputs
EIF	External Interface File
EO	External Outputs
EQ	External Inquiries
FFP	Full FP
FP	FP
FPI	Function Point Index
FSM	Functional Size Measurement
FTR	File Types Referenced
FUR	Functional User Requirements
GUI	Graphical User Interface
LOC	Lines of code
IFPUG	International Function Point User's Group
ILF	Internal Logical File
ISO	International Standards Organization
JDK	Java Development Kit
MIS	Management Information System
MVC	Model-View-Controller
NESMA	Netherlands Software Metrics Association

RET	Record Element Types
SELAM	Software Engineering Laboratory in Applied Metrics
SPR	Software Productivity Research Inc.
SRS	Software Requirements Specification
TCA	Technical Complexity Adjustment
TDI	Total Degrees of Influence
UC	Use Case
UKSMA	United Kingdom Software Metrics Association
UML	Unified Modeling Language
UQAM	Université du Québec à Montréal
VAF	Value Adjustment Factor

CHAPTER 1

INTRODUCTION

1.1. Background

Over the past few decades, software development industry has a reputation for software projects being over budget, late in delivery and low quality. Every year billion dollars spend for unsuccessful software development projects. Many of these projects do not meet the expected needs. Some of them are abandoned before completion.

Among numerous possible reasons for the software project failures, the most significant one is poor project management. The fundamental driver of the software estimation is software size. As the quality of size measurement directly influences the effort, cost and schedule estimates, accurate size measurement is highly critical to reliable planning and effective management of the software development process.

Size estimation should be established as early as possible in the project. Actually, at no other time, the estimates are so important than at the beginning of a project, since the sooner you can quantify what a project is delivering, the sooner it is under control. However, in the initial stages when the estimates are of most use, size estimation is very difficult and subject to higher margins of error, as very little is known in terms of requirements. Therefore, size estimation and measurement should not be a one-time activity at project initiation. Estimation is an iterative process and estimates should be refined continually throughout a project. As more details become

available, it becomes possible to get better estimates and measures with lesser and lesser margins of errors.

There have been proposed many different approaches and various methods for size estimation/measurement up to now. The length of code and functional size are two widely-known measures which are often used to determine the size of an application.

The number of Lines of Codes (LOC) is one of the famous metric used to measure the length of code. LOC is a technical measure as it measures the software from the developer's technical point of view, not the user's functional point of view. It is a direct measure that can easily be counted and manipulated (Pressman, 2001).

Functional size is the functionality based size measure of software. ISO defines that functional size as a size of the software derived by quantifying the Functional User Requirements (FURs) (ISO/IEC 14143-1, 1998). Function Point Analysis method (FPA) was first proposed by Alan Albrecht as an alternative to LOC (Albrecht, 1979). Albrecht wanted to measure the functionality of software from the user viewpoint, independently of the implementation and so introduced Function Point (FP) as a measure of functional size (Fetcke, 1999).

FPA is independent of the development methodology, programming language and capability of the project team used to develop the application. It completely depends on the functionality delivered to the user. It provides an objective, comparative measure that assists in the evaluation, planning, management and control of software production. As FPA relates directly to the business requirements, it can be applied throughout the life of a development project, from early requirements definition to full operational use. The capability to accurately quantify the size of software in early phases and to control the functionality delivered during the software lifecycle is highly critical to software project managers for developing accurate project estimates, evaluating potential risks and having early project indicators.

Since their initial publication, the use of FPA has grown worldwide and many changes, improvements, extensions and alternative approaches to the original version have been introduced. Today, there are many variants of FPA in use.

The current promoter of Albrecht's FPA is International Function Point Users Group (IFPUG). IFPUG improves the FPA method and periodically releases the Counting Practices Manual for consistent counting of FP across different organizations. Today IFPUG FPA is known as the most commonly used Functional Size Measurement (FSM) method (Lothar and Dumke, 2001).

Around 1996, the International Standards Organization (ISO) was formed a committee to establish an international standard for FSM. According to work of this group, ISO published a series of standards for FSM. As of 2005, there are six parts of this standard.

Currently IFPUG FPA-ISO/IEC 20926:2003, NESMA FPA-ISO/IEC 24570:2003, Mark II FPA (Mk II FPA) - ISO/IEC 20968:2003, Common Software Measurement International Consortium Full FP (COSMIC FFP) - ISO/IEC 19761:2003 methods are approved by ISO as international standards for FSM.

1.2. Problem Statement

As the number and variation of FSM methods has increased, the need to convert the functional sizes, obtained by utilizing different methods, have arisen. One of the improvement opportunities of FSM is identified as the convertibility between different FSM methods (Gencel, 2005).

There are few studies to define the convertibility of functional sizes measured by different FSM methods. Symons (1999) studied the convertibility of Mk II FPA and IFPUG FPA measurement results to each other, and gave a mathematical formula. Abran et al (2005) studied on the convertibility IFPUG FPA to COSMIC FFP measurement results based on the three case study data sets. They set up

different formulas for each data set and cannot come up with a unique conversion formula.

It would be ideal that sizes measured with the method could be converted to another by utilizing one mathematical formula. However, there are practical and theoretical reasons why this may not be easy and why it is not recommended for certain measurements.

Existing convertibility studies mostly aim to develop statistically-based conversion formula. To set up statistically-based conversion formula practically, numbers of measurements should be collected by applying different methods to measure the functional size of the same project. Such repeat measurements require a lot of effort as well as expertise in more than one method and few organizations have done up to date (COSMIC FFP Measurement Manual, 2003). Moreover, in order to develop mathematically-based conversion formula, the conceptual mapping between the FSM methods and their measurement rules should be studied.

The FSM methods share a core view and certain core concepts, even though each makes use of these concepts differently giving different names and defines their own measurement rules. Instead of using the concepts and models of a particular development method, each variant of FSM method defines its own concepts for the representation of a software application. Basically, most of these concepts point to the same information.

Moreover, the rules that they use in the measurement processes are quite related. Sometimes, they use the same information in a different way or look at the same concepts from different point of views or use different abstractions during the measurement process. Naturally, all concepts and counting rules in one FSM method may not have one-to-one mapping to others.

Another difficulty for defining conversion formulas is related to the collection of measurement data. According to Bundschuh et al (2000), a method without tool support has little chance to survive. Tool support is significant for

assistance so as to handle, store and evaluate the data. Many tools have been developed to improve size measurement efficiency. Primarily, they provide data repository for recording FSM results. However, they do not enable to keep the details of the elements used during the measurement process which hinders to make conversion studies at a detailed level. Moreover, most of them support only one type FSM method, particularly IFPUG FPA.

Frequently, the estimators use some spread sheet software in order to keep the detailed information on the measurement process of FSM methods. However, this approach is error-prone and time-consuming. Furthermore, the traceability of the changes is very difficult since the estimator should trace one change throughout all measurement records manually.

1.3. The Approach

In this thesis, we primarily focus on developing an infrastructure to help to measure the functional size of software applications by utilizing different FSM methods simultaneously from a single source of data.

We selected four FSM methods; IFPUG FPA, Mk II FPA, COSMIC FFP and Architectural Dimensions Based Functional Size Measurement (ARCHI DIM FSM) methods. The reason why we have selected these methods among many approaches is that IFPUG FPA, Mk II FPA and COSMIC FFP comply with international ISO FSM standards and ARCHI DIM FSM has been developed to be conformant to ISO FSM standard. Although NESMA FPA is another ISO certified FSM method, we do not include it in this study as it is quite similar to IFPUG FPA.

In order to help to solve some of the issued discussed in the previous section, we set the objectives of this thesis as follows;

- To propose a unification model by investigating the similarities and differences between the four FSM methods in order to measure the functional size of a software system by applying different FSM methods simultaneously, using a single source of data. These are the steps taken so that this model is established;

- ❖ Study the covered FSM methods in detail.
- ❖ Investigate similarities and differences between the concepts of the methods and define common concepts.
- ❖ Map the concepts of each method to the other if they are common for the methods.
- ❖ Examine the similarities and differences between the measurement processes of the methods.
- ❖ Map the measurement rules and construct possible ways to apply common measurement rules.
- To provide an infrastructure to enable the estimator enter additional data if it is required by one of the methods specifically,
- To develop a size estimation/measurement tool, based on the unification model proposed. These are the steps taken in order to develop the tool;
 - ❖ Prepare a checklist and evaluate existing estimation/measurement tools according to the checklist,
 - ❖ Develop the tool based on the requirements of the unification model and the deficiencies found in the existing tools.
- To conduct empirical studies in order to validate the unification model and the tool.

1.4. Thesis Structure

Chapter 2 provides the related research including the functional size estimation/measurement methods, details of the FSM processes of the selected FSM methods and existing software size estimation tools. In addition, these tools are evaluated with respect to the established checklist for the size estimation tools.

Chapter 3 presents the unification model proposed for selected FSM methods. The terminology mapping, common core concepts and the mapping of rules for FSM methods are given.

Chapter 4 explains the software requirements and design specifications of the FSM tool, called EasyEstimate.

Chapter 5 includes the empirical studies conducted to evaluate the unification model and EasyEstimate. The measurement results of the empirical studies are discussed. The usage scenario of EasyEstimate is also explained on this case study.

Chapter 6 concludes the thesis and includes the future work directions about unification of FSM methods and the automation possibilities of FSM.

CHAPTER 2

RELATED RESEARCH

This chapter presents the results of the literature review on software size measurement/estimation methods and tools as well as detailed measurement processes of the selected functional size measurement methods.

2.1. Software Size Measurement/Estimation Methods

Software size measurement/estimation is considered as a fundamental activity regarding software management tasks. Effort and cost estimations, resource allocation and scheduling activities are performed subsequently based on the size of the software. The reasonable project plan, realistic allocations of time and resources can only become possible with accurate size estimate. Hence, software size is an important measure for productivity management, quality management and contract management activities.

Size measurement/estimation has been the object of a lot of research in the literature, as the reliable and reasonable estimates have such vital importance in project management. There have been proposed many different approaches and various methods for size estimation up to now like the FSM methods, e.g. IFPUG FPA, Mk II FPA, COSMIC FFP, length of code based methods e.g. counting LOC, expert opinion methods, e.g. Wideband-Delphi method and comparing with past history.

Measuring the length of code is the traditional way of measuring the application size. LOC has been one of the first size metrics used by software developers to measure the length of code. LOC became popular since most of the developer time was spent writing the code that was the most visible output of the development cycle and it measured what software developers actually do, that is, the number of lines of code. It was also easy to verify whether the estimate was accurate by counting the lines of code at the end of the project.

Even though these advantages, LOC as a measure of the system size no longer holds the domination in size measures, because of its several drawbacks.

LOC is programming language dependent. It depends on how code is written, and so skill of the programmer. The skillful programmer can develop the same function with fewer LOC. Higher-level languages such as Java require far fewer lines of code than Assembler, COBOL, or C to perform the same functionality. Therefore, programs written in different languages cannot be directly compared.

LOC of a system cannot be correctly known till the system is developed. Estimation of size is often needed at very initial stages and expressing the size of a project in LOC at an early stage is almost impossible.

There is no agreed upon standard definition of what a LOC is and what to count. As many different LOC definitions exist, problems arise while counting data declarations, blank lines, comment lines, macros, and statements extending several lines. There is no organization exists to publish some guidelines in an attempt to standardize LOC counting. Whether LOC for maintenance functions and utilizes and LOC of the programs written for testing should be counted are the one of the common problems.

The methods based on measuring the amount of functionality have been proposed to overcome the deficiencies of source code based approaches. It is the goal of these methods to measure the functionality of the software, independent of its implementation. These methods measure the logical external view of the software

from the user's perspective by evaluating the amount of functionality to be delivered. The capability to accurately quantify the size of software in an early phase of the development project and to control the functionality delivered during the software lifecycle, is critical to software project managers for evaluating potential risks, developing accurate project estimates and having early project indicators.

The idea of measuring the size of software in terms of its 'functionality' as opposed to LOC was first proposed by Allan Albrecht of IBM in 1979 (Albrecht, 1979). He wanted to measure the functionality of software from the user viewpoint, independently of the technology used for its development. He proposed a method, called FPA. The first version of the method was presented in 1979 at a conference in California. Then the model is refined in 1983 by Albrecht and Gaffney (Albrecht and Gaffney, 1983).

In Albrecht's FPA method, the software is counted from the view of four characteristics; External inputs and outputs, User interactions, External interfaces and Files used by the system. Each of these characteristic is individually assessed for complexity and given a weighting value which varies from 3 (simple) to 15 (complex).

In 1986, IFPUG was formed to maintain this method. IFPUG has modified Albrecht's original method several times. IFPUG has published its own Counting Practices Manual's (CPM) to clarify and standardize rules for the application of FPA in 1988 (Release 2.0), 1990 (Release 3.0), 1994 (Release 4.0), 1999 (Release 4.1) and 2005 (Release 4.2). FPA is formulated as a counting method of several steps in these CPMs. Each release of these IFPUG publications contained refinements to the technique originally presented by Albrecht. This organization always claimed to be consistent with his original thinking. In truth, it is still very close considering the nearly two decades that have elapsed since Albrecht's original publication (Boehm, 1997).

Today, the IFPUG FPA is the most-widely used FSM technique and has become a quasi standard (Lothar and Dumke, 2001).

Albrecht's FPA laid the foundations for the subject of FSM. Since its first presentation to the public, Albrecht's FPA has grown worldwide and has been a basis for several improvements and alternative proposals. Several developments from the Albrecht/IFPUG FPA approach have been made to improve the size measure, to extend its domain of applicability or completely replace the work done by Albrecht during the eighties and nineties. The most significant FSM methods are given in Table 1.

Table 1 FSM Methods

Year	Method	Developer
1979	Albrecht FPA	Albrecht, IBM
1986	Feature Points	Jones, SPR
1988	IFPUG FPA CPM 2.0, 1988 CPM 3.0, 1990 CPM 4.0, 1994 CPM 4.1, 1999 CPM 4.2, 2005	International Function Point Users Group (IFPUG)
1988	Mk II FPA	Symons
1990	NESMA FPA	The Netherlands Software Metrics Users Association
1992	3-D FP	Whitmire, Boeing
1997	Full FP	Software Engineering Laboratory in Applied Metrics of The University of Québec
1999	COSMIC FFP	Common Software Measurement International Consortium (COSMIC)

Feature Points method was developed by Software Productivity Research, Inc. (SPR) in 1986 (Jones, 1987). It is an extension of Albrecht's FPA, with the aim of gaining gain better measurements for real-time process control, mathematical optimization and various embedded systems, as well as Management Information Systems (MIS).

This technique considers the number of algorithms used in the application and slightly modifies some of the weights of the traditional function point constituents (Jones, 1987). It was designed in such a way MIS applications would have the same size whether calculated with FP or feature points.

Feature Points add one extra element, Algorithm, to the set of the five Function Point components of the Albrecht's FPA. An algorithm defined as the set of rules, which must be completely expressed in order to solve a significant computational problem (Jones, 1987). With the other Function Point components, the algorithm component is assigned a weighted value. When used the Feature Points method, the values assigned to Internal Logical Files are reduced (Herron and Garmus, 1999). When applied to the computationally complex applications, this will yield a higher count than the Albrecht's FPA.

The Feature Points method have been experimental for a long time, thus there is not enough statistical evidence that it can be applied in a consistent fashion. Today this method is not longer supported by SPR (Lothar and Dumke, 2001).

Mk II FPA was developed by Charles Symons in 1988 to overcome the weaknesses of Albrecht's FPA. Symons claims that the Albrecht approach suffers from the following weaknesses (Boehm, 1997):

- It is often difficult to identify the components of an application. For example, what is a logical file?
- Albrecht had assigned weights to function point components based on "debate and trial."
- The above two criticisms were also leveled at the identification and weighting of Value Adjustment Factors.
- Albrecht did not provide a means of accounting for internal complexity. This is the same problem regarding algorithms that the Feature Points technique was developed to address.
- When small systems are combined into larger applications, Albrecht's approach makes the total function point count less than the sum of the components.

The method decomposes the application being counted into a collection of logical transactions. Each transaction consists of an input, a process and an output. For each transaction, FP becomes a function of the number of input data element

types; entity-types referenced and output data element-types. The FP for the entire system are then summed (Boehm, 1997).

This method was basically designed to measure the business information. To apply Mk II FPA to other domains such as scientific and real-time software may be possible or may require some extensions to or new interpretations of the rules given.

The weightings introduced by Mk II FPA were designed to deliver a size scale of similar magnitude for the Mk II FPA method as for the IFPUG FPA method. On average therefore, the methods give roughly the same software sizes up to around 400 FP. For larger sizes, Mk II FPA method tends to produce increasingly higher sizes than the Albrecht/IFPUG FPA method.

The design authority of Mk II FPA is the United Kingdom Software Metrics Association (UKSMA). Mk II FPA is widely used in the United Kingdom and increasingly in places like India, Singapore, Hong Kong and Europe.

NESMA FPA was published a variant of IFPUG FPA in 1990 with the aim of simplifying some of the IFPUG FPA sizing rules (NESMA Manual, 2005). In those days, FPA was particularly applied to measure productivity. The NESMA wanted to use FPA for budgeting purposes and, therefore, wanted to count beforehand on the basis of an application's functionality. In order to do this, it adapted a number of 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 those days.

After publication of IFPUG CPM 4.0 in 1994, the counting guidelines of the NESMA and the IFPUG continuously came closer and closer. With the publication of IFPUG CPM 4.1 in 1999 the FPA counting guidelines became the same, except a few minor guidelines (NESMA Manual, 2005).

Both the NESMA FPA (NESMA CPM 2.0) and the IFPUG FPA (IFPUG CPM 4.1) now use the same philosophy, the same concepts and terms, and the same

rules and guidelines within FPA. Nevertheless, at the request of its members, the NESMA published concrete, operational guidelines on complex counting issues for helping counters. These additional FPA guidelines fit within the general IFPUG guidelines; they just tend to clarify or interpret the IFPUG guidelines. These guidelines are also applicable for those FPA counters using the IFPUG Counting Practices Manual (NESMA Manual, 2005).

3D FP was publicly introduced by the Boeing Computer Services Software Metrics Team in 1991 (Whitmire, 1995). The 3D FP was developed in response to the call for a technology-independent size metric suitable for all domains. The new technique was designed to address two classic problems associated with the Albrecht approach; difficult to use and not properly measure scientific and real-time systems (same as emergence of feature points) (Whitmire, 1995).

The 3D method is based upon the premise that the application problem can be expressed in three dimensions: data function and control. Each dimension contains some of the characteristics that create complexity in a problem. Sometimes one dimension dominates, but all dimensions of the problem must be analyzed if accurate measurement is desired (Herron and Garmus, 1999). The data dimension is similar to Albrecht's FPA method. Data strong problems are typically associated with MIS/business software environments. The function dimension includes the number and the complexity of functions that represent internal processing required transforming input data into output data, and the sets of semantic statements that manage the process, which are similar to the algorithms. Function strong problems are associated with scientific/engineering environments. The control dimension adds transitions, which enumerate changes in application state. Control strong problems are associated with real time environments. It has been claimed 3D FP scale downward to the class level when used in conjunction with object-oriented development (Boehm, 1997).

3D FP is still used successfully in Boeing but unfortunately no details have been published outside Boeing (Symons, 2001).

Full FP (FFP) method was introduced in 1997 by Software Engineering Management Research Laboratory of The University of Québec, Montréal and its industrial research partner Software Engineering Laboratory in Applied Metrics (SELAM) to adapt FPA to real-time software (FFP CPM, 1997).

FFP uses the IFPUG FPA rules for business application software and adds six additional data and function types for sizing real-time software. The two new Control Data Function Types have a structure similar to that of the IFPUG FPA Data Function Types; Update Control Group like ILF of IFPUG FPA and Read-only Control Group like EIF of IFPUG FPA. The four new Control Transactional Function Types address the sub-process of real-time software; External Control Entry, External Control Exit, Internal Control Read, Internal Control Write (FFP CPM, 1997).

FFP measurement practice in many organizations and field tests led by UQAM's Software Engineering Management Research Laboratory and the Software Engineering Laboratory in Applied Metrics have demonstrated that FFP not only has the ability to capture the functional size of real-time software, but also to capture the functional size of technical and system software (COSMIC FFP Measurement Manual, 2003). Furthermore, these field tests have shown that Full FP is also suited to measuring the functional size of MIS software, leading, in such applications, to similar result (Oigny et al, 1998). The method has evolved to COSMIC FFP in 1999.

COSMIC FFP method was introduced in 1999 as a second version of FFP method by Common Software Measurement International Consortium (COSMIC). The COSMIC group was formed in 1998 to design and bring to market a new generation of software measurement method. They reviewed existing methods; IFPUG FPA, Mk II FPA, NESMA FPA and version 1.0 of the FFP method, studied their commonalities, and proposed the basic principles on which a new generation of software FSM method could be based (Oigny et al, 2000).

COSMIC FFP was designed to measure the functional size of real-time, multi-layered software such as used in telecoms, process control, and operating

systems, as well as business application software, all on the same measurement scale. However, it does not take into account the functional size of software which are characterized by complex mathematical algorithms or other specialized and complex rules, such as expert systems, simulation software, self-learning software, weather forecasting systems and process continuous variables such as audio sounds or video images, for instance, in computer game software, musical instruments (COSMIC FFP Measurement Manual, 2003).

COSMIC group has been maintaining and revising the method regularly and accordingly published version 2.0 of COSMIC FFP in 1999, version 2.1 in 2001 and version 2.2 in 2003 (COSMIC FFP Measurement Manual, 2003).

ARCHI DIM FSM stands for ARCHItectural DIMensions Based Functional Size Measurement. It was developed in the scope of a PhD thesis by Gencel (Gencel, 2005). ARCHI DIM FSM measures the FURs and quantifies different types of functionalities delivered to the users according to the architectural dimensions of the software. It measures the amount of Interface, Permanent Storage, Control Process, Algorithmic/Manipulation Process functionalities separately. It is designed to be applicable to measure application software from the domain of data-strong, control-strong, function-strong and hybrid systems.

The measurement guideline of ARCHI DIM FSM has been prepared in the scope of the PhD thesis by Gencel (ARCHI DIM FSM Measurement Guideline, 2005).

In 1996, a new group of the ISO/IEC Joint Technical Committee was formed to establish an international standard for FSM. Because of the lack of generally accepted and sufficiently rigorous validation processes for FSM methods, it seems difficult to evaluate new proposals, both on a practical and on a theoretical level. In response, according to work of this group, ISO published a series of standards for FSM. Part 1 of the standard, ISO/IEC 14143-1, provides the concepts of FSM and establishes a basis against which existing and new FSM methods can be compared. Part 2, ISO/IEC 14143-2, provide a process for checking whether a candidate FSM

method conforms to the concepts described in Part 1. Part 3, ISO/IEC TR 14143-3, provide a framework for verifying the statements of an FSM method and/or for conducting tests with the defined performance criteria. Part 4, ISO/IEC TR 14143-4 presents the standard Reference User Requirements together with guidance on Reference FSM Methods. Part 5, ISO/IEC TR 14143-5, describe the characteristics of Functional Domains and the procedures by which characteristics of Functional User Requirements (FUR) can be used to determine Functional Domains. Part 6, ISO/IEC TR 14143-6 (2005), is a guide for the use of ISO/IEC 14143 and related International Standards.

Currently, four methods - IFPUG FPA (ISO/IEC 20926:2003), NESMA FPA (ISO/IEC 24570:2003), Mk II FPA (ISO/IEC 20968:2003) and COSMIC FFP (ISO/IEC 19761:2003) - have been approved as being an ISO standard for FSM. Moreover, ARCHI-DIM FSM Method is intended to comply with ISO/IEC 14143-1.

2.2. FSM Processes of Applied FSM Methods

In this section, we present the details of FSM processes of IFPUG FPA, Mk II FPA, COSMIC FFP and ARCHI DIM FSM methods.

2.2.1. IFPUG FPA, Release 4.1.1

Counting rules of IFPUG FPA is described precisely in IFPUG Counting Practices Manual (IFPUG FP CPM, 2000). We present counting rules of IFPUG FPA briefly based on the IFPUG 4.1. CPM.

Before starting to count FP, the type of function point count is determined and the counting scope and application boundary is identified. Type of function point count can be Development project, Enhancement project or Application function point count.

In IFPUG CPM 4.1, the complexity of the Transactional Functions and Data Functions constitute the function point count.

Data Functions represent the functionality provided to the user to meet internal and external data requirements. Data function types are defined as internal logical files (ILF) and external interface files (EIF):

- An ILF is logically related data maintained within the boundary of the application.
- An EIF is logically related data referenced by the application, but maintained within the boundary of another application. If at least one transaction of the application writes to the particular data group, it is ILF. Otherwise it must be classified EIF.

The complexity of data functions is assigned based on the number of data element types (DET) and record element types (RET). A DET is a unique user recognizable, non-repeated field. A RET is a user recognizable subgroup of data elements within an ILF or EIF. There are two types of subgroups: Optional, Mandatory.

After identifying and counting the number of DETs and RETs of the data functions, functional complexity is calculated using complexity matrix in Table 2.

Table 2 Complexity Matrix for ILF/EIF

	1 to 19 DETs	20 to 50 DETs	51 or more DETs
1 RET	Low	Low	Average
2 to 5 RETs	Low	Average	High
6 or more RETs	Average	High	High

Transactional Functions represent the functionality provided to the user to process data. Transactional functions are three types of elementary processes: external inputs (EI), external outputs (EO) and external inquiries (EQ).

- An EI is an elementary process that processes data or control information that comes from outside the application's boundary. The primary intent of an EI is to maintain one or more ILF and/or to alter the behavior of the system.
- An EO is an elementary process that sends data or control information outside the application's boundary. The primary intent of an EO is to present information to the user or the retrieval of data or control information. The processing logic must

contain at least one mathematical formula or calculation, or create derived data. An EO may also maintain one or more ILFs and/or alter the behavior of the system.

- An EQ is an elementary process that sends data or control information outside the application boundary. The primary intent of an EQ is to present information to a user through the retrieval of data or control information. The processing logic contains no mathematical formula or calculation, and creates no derived data. No ILF is maintained during the processing, nor is the behavior of the system altered.

The complexity of transactional functions is based on the number of DETs and File Types Referenced (FTR). A FTR is an ILF read or maintained by a transactional function or an EIF read by a transactional function. Each transactional function type has specific rules for the identification of FTRs and DETs.

After identify and count the number of DETs and FTRs of the transactional functions, complexity of these transactions are rated using the complexity matrixes in Table 3 and Table 4.

Table 3 Complexity Matrix for EI

	1 to 4 DET	5 to 15 DET	16 or more DET
0 to 1 FTR	Low	Low	Average
2 FTRs	Low	Average	High
3 or more FTRs	Average	High	High

Table 4 Complexity Matrix of the EOs or EQs

	1 to 5 DET	6 to 19 DET	20 or more DET
0 to 1 FTR	Low	Low	Average
2 to 3 FTRs	Low	Average	High
4 or more FTRs	Average	High	High

The counts for data and transactional function types are classified according to complexity and then weighed using Table 5. The sum of all function types' weights gives the unadjusted function point count.

Table 5 Complexity Weights for ILF/EIF/EI/EO/EQ

	ILF	EIF	EI/EQ	EO
Low	7	5	3	4
Average	10	7	4	5
High	15	10	6	7

The adjusted function point count is calculated using a specific formula for a development project, enhancement project, or application function point count from the function point and the value adjustment factor (VAF).

VAF specifies the general functionality provided to the user of the application. It consists of 14 general system characteristics of the application (Table 30). After rating the characteristics, the sum of them is used to calculate VAF based on the following formula;

$$\text{VAF} = 0.65 + \text{total} / 100$$

2.2.2. Mk II FPA, Version 1.3.1

UKSMA defines the counting rules of the Mk II FPA in detail in the Counting Practices Manual. We state that counting rules of Mk II briefly according to Mk II FPA CPM Version 1.3.1 (Mk II FPA CPM, 2003).

At the beginning, viewpoint, purpose, type and boundary of the count are defined. There are three commonly encountered viewpoints; The Project Viewpoint, The Application Manager Viewpoint, The Business Enterprise Viewpoint.

Only Logical Transactions are counted in Mk II FPA. Changes to applications are counted by summing the size of the added, changed, and deleted Logical Transactions. Data Entity Types are just used to characterize transactions.

Mk II FPA defines data entity types as something in the real world about which the user wants to hold information and data element types as hold information about data entity types.

Data entity types are either Primary entities or Non-Primary entities;

- *Primary entities* are the main things in the real world to hold data. Mk II uses the Third Normal Form while identifying the primary entities.
- *Non-Primary entities* are all other entities except primary entities and treated as a single entity called *System Entity*. An application has only one system entity. However, an entity type that is non-primary for one application could be primary for another.

Moreover, there are some variations primary data entity types according to usage of the data entity type in the logical transaction.

- A *sub-entity type* is a specific type of a primary entity, for which there are distinct processing rules.
- A *self-referential entity type* is other specific primary entity-type, which is related to itself and traverses up and down the hierarchy for a transaction.

Mk II FPA method treats the system as a collection of logical transactions. Each logical transaction comprises three components; input across an application boundary, process on data entity types within the boundary, and output back across the boundary.

Mk II FPA makes the following assumption regarding the functional size of these three elements:

- The size of the input element (N_i) is proportional to the number of uniquely processed DET's crossing the application boundary
- The size of the processing element (N_e) is proportional to the number of Primary Entity Types referenced during the course of the logical transaction. If the system entity is also referenced, the count is incremented by one. Sub-entity types are also counted separately when different processing logic is used in the transaction. If there is self-referential entity, two entity references are counted, one for the initial reference and one for the repetitive loop, irrespective of how many times the hierarchy is traversed.
- The size of the output element (N_o) is proportional to the number of uniquely processed DET's crossing the application boundary back.

Conventionally, any transaction must have at least one input DET, must make one reference to an entity type and must have one output DET according to Mk II FPA.

The Functional Size (or Function Point Index) is the weighted sum over all Logical Transactions, of the Input Data Element Types (N_i), the Data Entity Types Referenced (N_e), and the Output Data Element Types (N_o).

$$FPI = W_i * \sum N_i + W_e * \sum N_e + W_o * \sum N_o,$$

W_i , W_e , and W_o are weights used for this computation. The industry standard weights stated in the Mk II manual are as follows: Input Weight (W_i) is 0.58 (per Input Data Element Type), Processing Weight (W_e) is 1.66 (per Entity Type Reference), and the Output Weight (W_o) is 0.26 (per Output Data Element Type).

These weights may be calibrated and adjusted based on past data. Obtained function point count should refer to the CPM Version Number when it is reported, for example:

‘318 MK II FP (V1.3)’, or

‘MK II FP Index = 318 (V1.3)’

FPI measures the functional or pure information processing size of the application as seen by the user. Besides them, Mk II FPA offers to take into account the technical complexity and certain quality requirements of the application. When the FPI is multiplied by the TCA (Technical Complexity Adjustment), the result is called the Adjusted Function Point Index (AFPI).

TCA method attempts to measure the influence of defined technical characteristics on the size of the application. There are 19 characteristics that contribute to the TCA for an application (Table 31). After rating the characteristics, the sum of them (Total Degrees of Influence, TDI) is used to compute TCA using below formula;

$$TCA = (TDI * C) + 0.65 \text{ where the current industry average value of constant } C \text{ is } 0.005.$$

The AFPI is obtained by multiplication of FPI and TAC as expressed following formula;

$$\text{AFPI} = \text{FPI} * \text{TCA}$$

2.2.3. COSMIC FFP Version 2.2

COSMIC FFP Measurement Manual Version 2.2 states the details of the counting rules of COSMIC FFP (COSMIC FFP Measurement Manual, 2003). We provide the rules of the measurement in brief in this section.

Before starting a measurement using the COSMIC FFP method it is imperative to define the purpose, the scope and the measurement viewpoint. The COSMIC FFP method offers a choice to the measurer in terms of viewpoint. It can be used to measure a functional size from the 'End-user' viewpoint, the same as that measured by the IFPUG FPA and Mk II FPA, or it can be used to measure a functional size as seen from the 'Developer' viewpoint. Sizes from different measurement viewpoints should obviously never be combined or used together.

Furthermore, software layers are identified. A layer is the result of the functional partitioning of the software environment such that all included functional processes perform at the same level of abstraction. Besides, boundary is defined.

Data Groups and Functional Processes are identified. In this version of the Measurement Method, it is not mandatory to identify the data attributes. Therefore, COSMIC FFP can be applied earlier in the development life cycle than others.

Data groups do not contribute to functional size. Nevertheless, data groups are essential in the identification of sub-processes because a sub-process may handle only one data group. COSMIC FFP characterizes data groups by its persistence as Transient, Short, and Indefinite. Transient data group exists only for the life of a functional process. Short data group survives beyond the life of a functional process for as long as the software is operational. Indefinite data group survives even when the software using it ceases to operate. In practice COSMIC FFP currently only

distinguishes ‘Transient’ from ‘Persistent’ (i.e. short or indefinite) data groups. Once persistence type identified, a data group must be materialized and the materialization of a data group takes three forms, as data group on I/O device, data group in volatile storage and data group on persistent storage.

Functional processes are broken down into four types of sub-processes, i.e. Data Movement Types. A data movement is a component of a functional process that moves one or more data attributes belonging to single data group. The Base Functional Component (BFC) Types of the COSMIC FFP are the Data Movements of the Functional Processes. There are four types of Data Movements;

- *An Entry (E)* moves a data group from a user across the boundary into the functional process s where it is required. It does not update the data it moves. It includes certain associated data manipulations, e.g. validation of the entered data.
- *An Exit (X)* moves a data group from a functional process across the boundary to the user that requires it. It does not read the data it moves. It includes certain associated data manipulations, e.g. formatting and routing associated with the data to be exited.
- *A Read (R)* moves a data group from persistent storage within reach of the functional process which requires it. It includes certain associated data manipulation sub-processes necessary to achieve the Read.
- *A Write (W)* moves a data group lying inside a functional process to persistent storage. It includes certain associated data manipulation sub-processes necessary to achieve the Write.

For each functional process, the numbers of data movements are identified. The functional sizes of individual data movements are aggregated into a single functional size value by arithmetically adding them together.

$$\begin{aligned} \text{Size}_{\text{Cfsu}}(\text{functional process}_i) &= \Sigma \text{size}(\text{entries}_i) + \\ &\quad \Sigma \text{size}(\text{exits}_i) + \\ &\quad \Sigma \text{size}(\text{reads}_i) + \\ &\quad \Sigma \text{size}(\text{writes}_i) \end{aligned}$$

The unit of the measurement is Cfsu (Cosmic Functional Size Unit). The simplest functional process has a size of 2 Cfsu (one Entry, plus an outcome, either as a Write or an Exit).

2.2.4. ARCHI DIM FSM

ARCHI DIM Measurement Guideline Version 1.0 explains the rules of ARCHI DIM FSM Method and gives the measurement guidelines (ARCHI DIM FSM Measurement Guideline, 2005). We express the counting rules of ARCHI DIM FSM based on this guideline.

In ARCHI DIM FSM, at the beginning of measurement process, the measurement purpose and scope shall be defined. The application boundary is determined. The type of measurement is identified, which is either measurement of development projects, enhancement projects or application.

Afterwards, Data Groups, Data Element Types and Elementary Processes are identified. The BFCs of ARCHI DIM FSM are Elementary Processes. Although data groups are not directly counted, they are used as a part of elementary processes. Data groups may have different forms in a piece of software as data groups on I/O device, data group in volatile storage, data group in permanent.

There are three essential parts of elementary process, which serve different functionalities:

- **Interface:** Involves the functionalities provided to an interfacing entity – a person who enters and receives output or automated user that move data in/out of a process via an interface.
- **Business Process:** may be of two types depending on the software functional domain. A hybrid software system may have more than one of these process functionality types:
 - ❖ **Control Process:** Involves the functionalities provided to an interfacing entity to control the behavior of a system.

- ❖ Algorithmic / Data Manipulation Process: Involves the functionalities provided to transform data item to create another one by means of mathematical and/or logical operations.
- Permanent Data Access/Storage: Involves the functionalities provided to an interfacing entity to access (read, write) Data Groups.

An elementary process may involve one or more constituent parts. For example, in data-strong systems, most of the elementary processes involve Interface and Permanent Data Access/Storage functionalities. In real-time systems, Control Process functionalities are also present.

In ARCHI DIM FSM, constituent parts of elementary processes have different functional sizes. Thus, the effort for each functionality type could also be estimated separately since development effort for each might be different. There may be some cases to develop these different types of functionalities with different technologies and by different teams.

Moreover, the constituent parts defined above contain different BFC Types. The size of each constituent part is proportional to the number of DETs of their BFC Types. These BFC Types are as follows;

- Size of Interface Part Functionalities:
 - ❖ Read from I/O Device
 - ❖ Write to Volatile Storage
 - ❖ Read from Volatile Storage
 - ❖ Write to I/O Device
- Size of Permanent Data Access/Storage Part Functionalities:
 - ❖ Read from Permanent Storage
 - ❖ Write to Volatile Storage.
 - ❖ Read from Volatile Storage
 - ❖ Write to Permanent Storage
- Business Process Functionalities:
 - ❖ Size of Control Process Part Functionalities:
 - Read from Volatile Storage.

- Write to Volatile Storage
- ❖ Size of Algorithm / Data Manipulation Part Functionalities:
 - Read from Volatile Storage
 - Write to Volatile Storage

The functional size of each constituent part of an Elementary process is the arithmetic sum of the values of their BFC Types. The functional size of the application is the arithmetic sum of the functional sizes of the Elementary Processes.

The unit of measurement is ADfsu (ARCHI DIM Functional Size Unit). The functional size of an application measured by ARCHI DIM is designated in four dimensions, as Interface, Control Process, Algorithm / Data Manipulation Process and Permanent Data Access/Storage.

2.3. Tool Support for Functional Size Measurement

There are numerous tools available to facilitate software size estimation/measurement process. In the Function Point Tool Market survey made by Software Engineering Management Laboratory (Mendes et al, 1996), there exist 52 tools as of 1996. This survey is the most important and detailed survey on function point tools so far. As this survey is very old, most of the tools in that survey are unreachable, ceased development or acquired by other companies.

FSM tools can be categorized as Repository Tools, Expert Tools, and Automated Tools. Moreover, these tools can be integrated with other management tools. IFPUG makes this categorization as Type 1, Type 2 and Type 3 respectively and certifies tools according to defined types.

The Repository Tools provides functional size data collection and calculation functionality, where the user performs the count manually and the tool acts as a repository of the data and performs the appropriate functional size calculations. The Expert Tools provides functional size data collection and calculation functionality, where the user and the tool determine the count interactively. The user answers the

questions presented by the tool and the tool makes decisions about the count, records it and performs the appropriate calculations. The Automated Tools carries out an automatic functional size count of an application using multiple sources of information such as the application software, database management system and stored descriptions from software design and development tools. The tool records the count and performs appropriate calculations. The user may enter some data interactively, but his or her involvement during the count is minimal.

We select the following tools to examine the functionalities in detail; FP Workbench¹, FPRecorder², PQMPlus³, EstimatorPal⁴, MkMAN (Özdamar, 2001), µcROSE (Diab et al, 2005) and Counter⁵ (see Table 6). In the scope of this survey, trial versions of FPW, FPRecorder, PQMPlus and demo version of EstimatorPal are installed and checked. Two papers about µcROSE are reviewed (Diab et al, 2005) (Azzouz and Abran, 2004). The related master thesis study of MkMAN is reviewed (Özdamar, 2001). MkMAN has been installed and investigated. Counter Web pages of the product vendors are examined. Moreover, product and help documentations of each tool are read and utilized.

Table 6 Examined Size Estimation Tools

	Supported FSM Method	Tool Category
FP Workbench	IFPUG FPA	Repository Tool
Counter	IFPUG FPA	Repository Tool
FPRecorder	IFPUG FPA	Repository Tool
PQMPlus	IFPUG FPA	Repository Tool, Expert Tool
EstimatorPal	IFPUG FPA, Mk II FPA	Repository Tool
MkMAN	Mk II FPA	Automated Tool
µcROSE	COSMIC FFP	Automated Tool

In the first part, brief descriptions of these tools are given. The deficiencies and the comparison of the tools are specified in the second part. Also, we form a checklist to evaluate the functionalities of the FSM tools and assess the tools according to this checklist in this part.

¹http://www.charismatek.com.au/_public1/html/fpw_overview.htm

²<http://www.fprecorder.com/>

³<http://www.qpmg.com/pqmplus.htm>

⁴<http://www.chemuturi.com/estimatorpal.html>

⁵<http://www.ddbsoftware.com/counterpage.htm>

2.3.1. Tools Survey

Function Point WORKBENCH

FP Workbench is probably the most sophisticated function point tool available in the market (Rudolph, 1997). It was developed by CHARISMATEK Software Metrics Company. It is in the market since 1992 and current version, 6.0, released in November 2005. It is most widely used tool to assist with function point counts.

It based on IFPUG FPA and supports different IFPUG CPM versions, CPM 4.0, CPM 4.1 and CPM 4.2. It is one of the IFPUG certified tools.

FP Workbench is a typical functional size repository. It provides recording files and transactions and calculates functional size in IFPUG FP utilizing arithmetic operations. It expresses the system graphically and visualizes transactions hierarchically. It keeps the results in system, project or phase level. Besides, system changes can be tracked at these levels. It supports FP counting for both application systems and software delivery projects. Also, it encourages an organization to build and maintain a database of FP counts for its applications and projects. It has enhanced reporting capability and the reports are available in text or graphical format. It also includes a range of external interfaces allow data to be easily transferred in and out.

Counter

Counter is another IFPUG certified tool for functional size repository. It is offered by DDB Software, Inc. It is mainly based on IFPUG FPA and supports IFPUG 3.4, 4.0 and 4.1 CPM versions.

Counter is designed to accept entry of FP for an application, a project, or a baseline. It presents the option of entering complete detail level information, detail within appropriate ranges, or just the final complexity (Low, Average, and High). Counter has an in-depth tutorial as part of its comprehensive Help function. Once an

application count has been completed, enhancements can be entered and tracked. An enhancement count will provide both a project and revised application count.

FP Recorder

FP Recorder was developed by Chis Pty Ltd in 1999. The current version, 3.0 was released in 2004. It is a repository tool supporting function point counts carried out using the IFPUG FPA. It supports IFPUG 4.0 and 4.1 counting practices.

It keeps count details, files, transactions and requirements and calculates function point by using these data. It provides the link between transaction and requirements. It has import and export functionality to their specific file format. Files, transactions and requirements can be saved in xml, html and text formats.

PQMPlus

PQMPlus is a project management tool that assists the information systems and application development team throughout the life cycle of a project. It is a product of Q/P Management Group, Inc. The current version, 4.0, was released in February 2005.

PQMPlus uses the measurement of IFPUG FPA to estimate the size of an application. It is compliant with version 4.0 and 4.1 of the IFPUG CPM. It has been awarded Type 1 and Type 2 Software Certification by IFPUG. It is the only tool to earn Type 2 Certification as an Expert System, as it provides computer-aided assistance in the application of the IFPUG counting rules through a structured interview process. It displays a series of interview questions to determine the type of logical functions analyzed. These questions are based on the IFPUG CPM version 4.1. Once all of the questions have been answered and PQMPlus determines the type of function being analyzed. Afterwards, it requires additional information in order to determine how many FP should be assigned to the function i.e. RET and DET for Logical Files or FTR and DET for Logical Transactions.

PQMPlus determines and documents the size and complexity of an application or development project, and assesses the project's risk and corporate

value. The project's productivity, cost, and quality can also be calculated using this tool. PQMPlus uses FP as a unit of measure. It provides fully documenting and tracking application function point details, establishing project estimates using in-house or industry benchmark data. Beyond the count, it helps assessing risk and corporate value, managing project scope, generating project schedules, creating quality measurements, flexible reporting.

EstimatorPal

EstimatorPal was developed by Chemuturi Consultants in 2005. It assists software developers estimate the size, effort and cost required during development of software. In addition, it facilitates preparing a schedule that can be exported to specialized scheduling tools.

This tool facilitates use of the following estimation techniques;

- IFPUG FPA
- Objects Points
- Use Case Points
- Task-Based Estimation
- Intermediate COCOMO
- LOC
- Mk II FPA

It is very basic tool that allows user to enter required numbers for the estimations and it makes calculations.

MkMAN

MkMAN was developed in the scope of a master thesis in METU Informatics Institute (Özdamar, 2001). It is an automated function point measurement tool. It automatically measures function point of the system that is defined with Unified Modeling Language (UML) notation in Rational Rose Software using Mk II FPA.

This tool has two parts;

- First part converts Rational Rose model file to the predefined XML format.

- Then, second part manipulates the xml file and establishes function point count.

Use Case, Class and Scenario diagrams of UML are included in the tool.

μcROSE

μcROSE is a software tool that automatically measures functional software size, as defined by the COSMIC FFP method, for Rational Rose Real-Time models (Diab et al, 2005). This tool is available for β-testing currently.

This tool is based on the direct mapping between COSMIC FFP and UML concepts and notation. It provides to derive the accurate functional size automatically after all specifications have been completed with UML. Therefore, it almost eliminates measurement costs and advanced measurement training and removes measurement variance and ensures perfect repeatability.

It is the first tool to address automatic measurement of COSMIC FFP. Three levels of measurement have been proposed in the tool according to different diagrams in UML; business modeling level, analysis level, design level. Each level has its own level of granularity and its own unit of measurement. The design level corresponds exactly to the COSMIC functional size unit.

2.3.2. Comparison of the Existing Tools

When the existing tools from the perspective of the thesis' motivation are evaluated, we conclude that:

- Most of the tools support only IFPUG FPA.
- Most of the tools do not support more than one FSM method.
- None of the tools provides converting functional size measure obtained by one FSM method to another.
- None of the tools enables measuring functional size by one FSM method just using data of other method.
- Most of the tools are repository tools that only store FP measurement results.

- Most of the tools do not have the capability to keep detailed information about the BFC Types. They only record the number of BFC Types.
- None of the tools keeps the effort utilized to make the measurement
- Most of the tools do not keep FURs and link between the FUR and Transactions identified during FSM process.

Based on these results, we formed the following checklist given in Table 7 and compared the tools in Table 8.

Table 7 Checklist for the Comparison of the FSM Tools

1	Does the tool provide the ability to measure functional size by applying different FSM methods?
2	If item 1 is yes, does the tool provide the ability to measure functional size with the single source of data from different FSM views
3	If item 1 is yes, does the tool enable the convertibility of the functional size measures obtained by different FSM methods to each other?
4	Does the tool support early functional size estimation of a software system?
5	Does the tool provide the ability to record data groups and data element types of them other than count?
6	Does the tool provide the ability to record the BFC Types (for example transactions, data groups) in detail rather than just their count?
7	Does the tool enable to reflect BFC Type changes to whole application directly?
8	Is the tool requires size measurement expertise for the estimator?
9	Does the tool have versioning/baseline capability?
10	Does the tool have the ability to export/import data to/from external programs?
11	Does the tool have reporting capability?
12	Does the tool enable to keep efforts utilized for measurement?
13	Does the tool keep FURs of which the functional size is measured?
14	If item 13 is yes, does the tool provide traceability between requirements and Transactions?

Item	FP Workbench	Counter	FPRecorder	PQMPlus	Estimator Pal	MkMAN	µcROSE
1	No (only supports IFPUG FPA)	No (only supports IFPUG FPA)	No (only supports IFPUG FPA)	No (only supports IFPUG FPA)	Yes (supports IFPUG FPA and Mk II FPA)	No (only supports Mk II FPA)	No (only supports COSMIC FFP)
2	No	No	No	No	No (Supports different size estimation methods but each method needs different and their own data)	No	No
3	No	No	No	No	No	No	No

Table 8 Comparison of Examined FSM Tools

Table 8 Comparison of Examined FSM Tools (Cont.)

Item	FP Workbench	Counter	FPRecorder	PQMPlus	Estimator Pal	MkMAN	µcROSE
4	Partially (Only needs to number of DETs)	Largely (Three Options for entering complete detail level information, detail within appropriate ranges, or just the final complexity)	Partially (Only needs to number of DETs)	Partially (Only needs to number of DETs)	Partially (Only needs to number of DETs)	No (Need to define DETs for class and interaction diagrams)	Yes (COSMIC FFP provides estimation of size without DETs)
5	No (Only data groups and the number of DETs are recorded, not the DETs)	No (Only data groups and the number of DETs are recorded, not the DETs)	No (Only data groups and the number of DETs are recorded, not the DETs)	No (Only data groups, the number of DETs and are recorded, not the DETs)	No (Only data groups and the number of DETs are recorded, not the DETs)	Yes (Need to define DETs for class and scenario diagrams)	No (COSMIC FFP does not need to DETs)

Table 8 Comparison of Examined FSM Tools (Cont.)

Item	FP Workbench	Counter	FPRecorder	PQMPlus	Estimator Pal	MkMAN	µcROSE
6	Partially (Only transactions, referenced data groups and the number of referenced DETs are recorded, not the DETs)	Partially (Only transactions, referenced data groups and the number of referenced DETs are recorded, not the DETs)	Partially (Only transactions, referenced data groups and the number of referenced DETs are recorded, not the DETs)	Partially (Only transactions, referenced data groups and the number of referenced DETs are recorded, not the DETs)	No (Only transactions and complexities are recorded, not the referenced data groups and DETs)	Yes (In the scenario diagrams data groups and DETs are kept)	Partially (In the scenario diagrams data groups are kept but not necessarily DETs)

Table 8 Comparison of Examined FSM Tools (Cont.)

Item	FP Workbench	Counter	FPRecorder	PQMPlus	Estimator Pal	MkMAN	µcROSE
7	Partially (As the referenced data groups are being kept, the tool reflects changes in associated transactions, but as only number of DETs are kept in data groups and transactions, the user have to make changes in all related data groups and transactions)	Partially (As the referenced data groups are being kept, the tool reflects changes in associated transactions, but as only number of DETs are kept in data groups and transactions, the user have to make changes in all related data groups and transactions)	Partially (As the referenced data groups are being kept, the tool reflects changes in associated transactions but as only number of DETs are kept in data groups and transactions, the user have to make changes in all related data groups and transactions)	Partially (As the referenced data groups are being kept, the tool reflects changes in associated transactions, but as only number of DETs are kept in data groups and transactions, the user have to make changes in all related data groups and transactions)	No (As only complexities of transactions are recorded and the tool does not keep relations between transactions and data groups, the user have to change complexities of all related transactions)	Yes (When automatically calculated from class and scenario diagrams)	Yes (When automatically calculated from scenario diagrams)

Table 8 Comparison of Examined FSM Tools (Cont.)

Item	FP Workbench	Counter	FPRecorder	PQMPlus	Estimator Pal	MkMAN	µcROSE
8	No	No	No	Yes (Tool provides an detailed interview method for guiding inexperienced estimators)	No	Yes	Yes
9	Partially (There are three count statuses as In Progress, Completed, and Locked. Locked is like baseline. No versioning capability)	No	Largely (It has baseline and versioning like capability. Instead of versions counts can be saved with different names)	Partially (It has Baseline and Completed status for applications but no versioning capability)	No	No	No

Item	FP Workbench	Counter	FPRecorder	PQMPlus	Estimator Pal	MkMAN	µROSE
10	Yes (Export to a count to another Workbench database, SPR's KnowledgePL AN, Galorath's SEER-SEM, DDB Software's S.M.A.R.T. Predictor. Import a complete count from another Workbench database and import files, transactions, value adjustment factor data from cvs files)	Partially (Export to MS Excel, MS Word, and DDB Software's S.M.A.R.T. Predictor)	Yes (Export and Import their specific file format. Export dialog allows copying the details to the clipboard. Also, It is possible to save a file, transaction and requirement as xml, html and text files)	Yes (Export and Import their specific text file format)	No	Yes (Import from Rational Rose)	Yes (Import from Rational Rose Real Time)

Table 8 Comparison of Examined FSM Tools (Cont.)

Item	FP Workbench	Counter	FPRecorder	PQMPlus	Estimator Pal	MkMAN	µROSE
11	Yes (Enhanced reporting capability in text or graphical format)	Yes (Simple reports are available)	Yes (Summary report are available)	Yes (Advanced reporting capability)	Yes (Simple reports are available)	No	No
12	No	No	No	No	No	No	No
13	No	No	Yes	No	No	Partially (Use cases are assumed as transaction)	Partially(Use cases are assumed as transaction)
14	No	No	Yes	No	No	Partially (Use cases are assumed as transaction)	Partially(Use cases are assumed as transaction)

Table 8 Comparison of Examined FSM Tools (Cont.)

CHAPTER 3

UNIFICATION MODEL FOR FSM METHODS

Convertibility between different FSM methods is one of the hot topics of FSM, due to the sharp increase in the number and variation of FSM methods. Since each FSM method defines its own metric and measurement processes, the convertibility plays important role in comparing functional sizes obtained by different FSM methods.

The simplest way of converting functional size in one method to another would be using a mathematical formula. However, setting up a conversion formula between methods is not easy. It requires collecting data and conducting numerous case studies.

Alternatively, the FSM methods share some concepts and related rules of measurement processes. These relations would be helpful to convert functional sizes or allow measuring functional sizes using a largest set of data concurrently.

In this chapter, we propose a unification model for the FSM methods. The main purpose of this model is to provide an infrastructure to obtain measurement results for any of these four selected methods from the same set of data. We studied the measurement processes of the four methods in detail in the previous chapter. Based on this survey and ISO standard for FSM (ISO/IEC 14143-1, 1998), the common concepts of the FSM Methods are identified, terminologies are mapped and the common measurement rules are identified to establish the unification model.

3.1. Common Concepts in Selected FSM Methods

While FSM Methods differ in their views on functional size, a number of methods share a core view and certain core concepts (Fetcke, 2001). Although the FSM methods differ in some respects when measuring the amount of software functionality such as in the measurement steps and the terminology used, they rely on the same core concepts defined in ISO/IEC 14143-1 (1998).

3.1.1. Definitions of Common Concepts

Some of these major concepts stated in ISO/IEC 14143-1 (1998) are the followings;

- **User:** any person that specifies FUR and/or any person or thing that communicates or interacts with the software at any time.
- **FUR:** a subset of the user requirements. They represent the user practices and procedures that the software must perform to fulfill the users' needs.
- **Boundary:** A conceptual interface between the software under study and its users.
- **Scope of Measurement:** the set of FURs to be included in a specific FSM instance. It is determined by the purpose for measuring the software.
- **BFC:** an elementary unit of FURs defined by and used by a FSM Method for measurement purposes. They recognize within the FUR and assigning values that are used to calculate the functional size.
- **BFC Type:** a defined category of BFCs.

Among the FSM methods, the ones within the scope of this thesis, namely IFPUG FPA, Mk II FPA and COSMIC FFP are ISO certified and ARCHI DIM FSM is developed to comply with ISO/IEC 14143-1.

We studied the FSM process of the methods in detail and identified the additional concepts used. The brief explanations of the common core concepts that we use in establishing the unification model are as follows;

- **User:** same as ISO/IEC 14143-1 definition

- **FUR:** same as ISO/IEC 14143-1 definition
- **Application:** The application is the object of the measurement. Applications provide functions to the users.
- **Boundary:** same as ISO/IEC 14143-1 definition
- **Type of Measurement:** the type of the project to be measured. The project can be a Development Project, Enhancement Project...etc.
- **Purpose of Measurement:** the statement that defines why the measurement is being undertaken, and/or what the result will be used for (COSMIC FFP Measurement Manual, 2003).
- **Scope of Measurement:** same as ISO/IEC 14143-1 definition
- **Viewpoint:** a form of abstraction for example; End User Viewpoint or Developer Viewpoint.
- **BFC:** same as ISO/IEC 14143-1 definition
- **BFC Type:** same as ISO/IEC 14143-1 definition. In addition to this definition, ISO/IEC 14143-5 (2004) decomposes BFC Types into 4 categories as Transaction classes, Data Types Recognized, Information Creation Function Types and Data Retention Requirements. Information creation function types corresponds to the processing way of transaction, range in complexity from very simple Boolean operations to complex mathematical algorithms. The FSM methods studied in the scope of this thesis do not recognize any specific Information Creation Function Types. IFPUG FPA takes into account these BFC Types implicitly in one type of Elementary Processes, the External Output. Moreover, the data retention type is the degree of persistency of the data types. The Data Retention Type is the topic of data concept.

In the scope of this thesis, we studied the ‘Data Types’ and ‘Transactions’ as the BFC Types in the unification model.

- **Transaction:** Transactions represent the functionality provided to the user. Transactions are process of interaction of the user with the application form a “logical” or “functional” perspective. They are one of the BFC Types of the FSM methods covered.
- **Data Types:** Data is stored by the application and transactions involve groups of data. *Data Element Type (DET)* represents the smallest data items meaningful to

the user. DETs are structured in logically related groups, *Data Groups*, similar to tables in a database (Fetcke, 2001). The data concept recognizes data elements as elementary items. Subgroups are the logical group of DETs within Data Group. The Data Retention Type, which is the degree of persistency of the data types, is also defined in some of the FSM methods when identifying data types.

3.1.2. Terminology Mapping

After introduced the common core concepts of the unification model, the overview of the terminology used for these core concepts by each FSM method is given in Table 9.

3.1.3. Constituent Parts of Methods

Although Transaction and Data concepts are common to four FSM methods discussed in Section 2.2., the detailed definitions, the terminology used for these concepts, identification and counting procedures differ notably. Additionally, they have different constituent parts and different classifications for different BFC Types, i.e. data and transaction concepts.

These differences of the studied FSM techniques are summarized in Table 10. As seen in the table, data concept are contributed to functional size directly only by IFPUG FPA. The other FSM methods count transactions and related data implicitly under the Transaction concept. Also, COSMIC FFP does not take into account DETs during measurement. ARCHI DIM FSM also considers algorithmic processes different from other FSM methods.

As we considered the Transaction and/or Data Types concepts as the BFC Type categories in establishing the unification model and FSM Methods measure functional size by identifying the BFC Types, we focus on the BFC Types in the next two sections. We present the similarities and differences of these concepts for the FSM methods in the scope of this study in detail. Furthermore, we involve common and specific measurement rules covering all counting rules of these methods in this

model. This unification model is to be the base for the measurement tool we developed as a part of this thesis study.

Table 9 Terminology used in the selected FSM methods

	IFPUG FPA 4.1	Mk II FPA 1.3.1	COSMIC FFP 2.2	ARCHI DIM 1.0
User	User	User	User	User
Functional User Requirements	Functional User Requirements	Functional User Requirements	Functional User Requirements	Functional User Requirements
Application	Application	Application	Application	Application
Boundary	Boundary	Boundary	Boundary	Boundary
Type of Measurement	Type of Count	-	-	Type of Measurement
Purpose of Measurement	Purpose of Count	Purpose of Count	Purpose of Measurement	Purpose of Measurement
Scope of Measurement	Counting Scope	-	Scope of Measurement	Scope of Measurement
Viewpoint of Measurement	-	Viewpoint of Count	Viewpoint of Measurement	Viewpoint of Measurement
Transaction	Transactional Process	Logical Transaction	Functional Process	Elementary Processes
Data Group	Data Function	Data Entity Types	Data Group	Data Group
Sub-group	RET	Subgroup	-	-
Data Element Type	Data Element Type	Data Element Type	Data Attribute	Data Element Type

Table 10 Different constituent parts and classifications for the selected FSM methods

	Data Group Types	Data Group Contribution	Transaction Types	Transaction Parts	Transaction Contribution
IFPUG FPA 4.1	ILF	# of DETs, # of RETs	EI	-	# of DETs, # of File Types Referenced
	EIF		EQ		
			EO		
Mk II FPA 1.3.1	Primary	-	-	Input	# of Input DETs
	System			Process	# of Referenced Data Entities
				Output	# of Output DETs
COSMIC FFP 2.2	On I/O Device	-	-	Entry	# of Data Movement
	In Volatile Storage			Exit	# of Data Movement
	On Persistent Storage			Read	# of Data Movement
					Write
ARCHI DIM FSM 1.0	On I/O Device	-	-	Interface	# of DETs-Read from I/O Device # of DETs-Write to Volatile Storage # of DETs-Read from Volatile Storage # of DETs-Write to I/O Device
	In Volatile Storage			Control Process	# of DETs-Read from Volatile Storage # of DETs-Write to Volatile Storage
				Algorithmic Process	# of DETs-Read from Volatile Storage # of DETs-Write to Volatile Storage
	On Permanent Storage			Permanent Data Access	# of DETs-Read from Permanent Storage # of DETs-Write to Volatile Storage # of DETs-Read from Volatile Storage # of DETs-Write to Permanent Storage

3.2. Data Type Concept

The data concept assumes the data element types as elementary items. A data group is a set of data elements stored by the application.

3.2.1. Data Group Concept

All four FSM methods agree on the data group concept but each use different terminology:

- IFPUG FPA defines data function as “User identifiable group of logically related data or control information referenced by the application”.
- Mk II FPA defines entity type as “Something (strictly, some type of thing) in the real world about which the business user wants to hold information”.
- COSMIC FFP describes data group as “Distinct, non empty, non ordered and non redundant set of data attributes where each included data attribute describes a complementary aspect of the same object of interest”.
- ARCHI DIM FSM terms data group as “The groups or collections of related and self-contained data about which the user wants to hold information”.

Each method classifies data groups in its own way:

- IFPUG FPA has two types of data functions; as ILFs and EIFs as defined in Section 2.2.1, page 18.
- Mk II FPA has two entity types; *Primary Entities* and *Non-Primary Entities*. There are some variations of primary entity types according to usage of the data entity type in the logical transaction like sub-entity type and self-referential entity type. These types are defined in Section 2.2.2, page 21.
- COSMIC FFP characterizes data groups by its persistence as Transient, Short, and Indefinite. The details of persistency characteristics are given in Section 2.2.3, pages 23 and 24. After persistence type identified, a data group shall be materialized as data group on I/O device, data group in volatile storage and data group on persistent storage.

- ARCHI DIM FSM states that data groups may have different forms in a piece of software such as data group on I/O device, data group in volatile storage, data group in permanent storage.

Mapping of Concepts Used for Data Groups by Different FSM Methods

ILF type of IFPUG FPA and Primary Entity Type of Mk II FPA are quite similar. However, IFPUG FPA differentiates data groups which are not in the application boundary as EIF but Mk II FPA does not. Mk II FPA considers Non-Primary Entity Type. COSMIC FFP characterizes data groups by its persistence as Transient, Volatile, and Indefinite. According to COSMIC FFP, materialization of a data group takes three forms, as data group on I/O device, data group in volatile storage and data group on persistent storage. However, ARCHI DIM FSM states that data groups can be on I/O device, in volatile storage, or in permanent storage. ARCHI DIM FSM puts only indefinite data groups in permanent storage but COSMIC FFP includes volatile and indefinite data groups in persistent storage. Volatile data groups are assumed in volatile storage of control process in ARCHI DIM FSM. Transient, volatile and indefinite data groups can be present at I/O device and transient and volatile data groups can be found on volatile storage for ARCHI DIM FSM and COSMIC FFP. Additionally, both of them do not declare any classification of data groups concerning place of storage with respect to application boundary. Moreover, IFPUG FPA and Mk II FPA do not mention about the persistence characteristics of data groups. Though these two methods count transient data element types, they do not contribute transient data groups. ILFs and EIFs of IFPUG and Primary Entity Types and Non-Primary Entity Types of Mk II FPA can be indefinite or volatile data groups.

Based on the different types and findings explained, we have categorized data groups from two perspectives in the unification model. According to retention characteristic, a data group can be *Transient*, *Volatile* or *Indefinite* data group and according to place of storage with respect to application boundary, a data group can be maintained inside the boundary of the application, maintained seldom inside the boundary of the application or maintained outside the boundary of the application.

The following rules for Data Group concept are applied for mapping FSM methods in the unification model.

1. All four methods concur on the data group concept.
2. Data groups contributes to functional size directly only for IFPUG FPA.
3. Default value for retention characteristic of data group is indefinite.
4. Default value for place of storage of data group is inside the boundary.
5. IFPUG FPA ILF type is the same with the data groups maintained inside the boundary.
6. IFPUG FPA EIF type is the same with the data groups maintained outside the boundary.
7. Primary Data Entity Type of Mk II FPA are data groups, maintained inside the boundary.
8. Non-primary Data Entity Type of Mk II FPA are identical to data groups maintained seldom outside the boundary. They are treated as a single entity, System Entity, during calculations.
9. According to COSMIC FFP, transient and volatile data groups can be in volatile storage. Volatile and indefinite data groups can be on persistent storage. Transient, volatile and indefinite data groups can be on I/O device.
10. In ARCHI DIM FSM, transient and volatile data groups can be in volatile storage. Indefinite data groups can be on permanent storage. Transient, volatile and indefinite data groups can be on I/O device.

3.2.2. Data Element Type Concept

These methods also agree on data element type concept. Moreover, except COSMIC FFP, all use the same terminology, “Data Element Type”.

- According to IFPUG FPA, DET is a unique user recognizable, non-repeated field.
- Mk II describes DET as a unique user recognizable, non-recursive item of information about entity types.
- COSMIC FFP defines data attribute as the smallest parcel of information, within an identified data group, carrying a meaning from the perspective of the software’s FURs

- ARCHI DIM FSM states that DETs hold information about data groups.

DETs are categorized regarding the usage in the transaction. In data group level, there is no distinction or classification in DETs. Moreover, only in IFPUG FPA, DETs are contributed to the functional size in data type level. The other FSM methods measure DETs in transaction level.

Mapping of Concepts Used for DETs by Different FSM Methods

As all methods concur on the DET concept totally, we can say that there is an exact mapping for DET concept for all FSM methods covered.

11. The “data attributes” of COSMIC FFP and DETs of the other FSM methods address the same concept.
12. Only IFPUG FPA measures DETs in data type level.

3.2.3. Sub-group Data Concept

Sub groups can be defined on the data elements of a data group. IFPUG FPA defines these sub-groups as RET, which is a user recognizable subgroup of data elements within an ILF or EIF. There are two types of RETs; Optional and Mandatory subgroups. Mk II FPA also considers sub-groups but not in the data group. They make sense in the transaction level. We can define one more rule for sub-groups in Mk II FPA.

Mapping of Concepts Used for Sub-groups by Different FSM Methods

13. The Sub-entity Type of Mk II FPA is similar to the definition of RET of IFPUG FPA but IFPUG FPA considers RETs once during counting data functions, though Mk II FPA counts sub-entities each time referred by transaction. Other methods do not consider sub groups.

To sum up, we have the following entities in the data type concept; Data Groups, Sub-groups and DETs.

3.3. Transaction Concept

All four FSM methods agree on the Transaction concept although each uses different terminology for it. The definitions of the methods for Transaction are as follows:

- IFPUG FPA calls Transaction as the Elementary Process, which is the smallest unit of activity meaningful to the user, must be self-contained, and leaves the application in a consistent state.
- Mk II FPA calls Transaction as the Logical Transaction, which is the lowest level business process triggered by a unique event of interest in the external world, or a request for information and, when wholly complete, leaves the application in a self consistent state in relation to the unique event.
- COSMIC FFP calls Transaction as the Functional Process, which is an elementary component of a set of FUR comprising a unique cohesive and independently executable set of data movements. It is triggered by one or more triggering events and complete when it has executed.
- ARCHI DIM FSM calls Transaction as the Elementary Process, which is an elementary unit of FUR supported by the application and that is meaningful to the user(s). It is triggered by a unique event and complete when it has executed.

Even though the Transaction concept is the same for the methods, Transaction types are represented differently in the methods.

- IFPUG FPA defines three types of transactions as EI, EO, EQ, which are already explained in Section 2.2.1, page 18 and 19.

The others do not have transaction types but they split transaction into parts.

- Mk II FPA Logical Transactions contain 3 components; input across an application boundary, processing involving stored data within the boundary and output back across the boundary.
- COSMIC FFP Functional Process has 4 sub-process types, defines as a data movement type occurring during the execution of a functional process type, as Entry, Exit, Read, and Write.

- ARCHI DIM FSM Elementary Process has 4 parts; Interface, Data Storage, Control and Algorithmic. An elementary process may involve one or more parts. These constituent parts contain different BFC Types.

Summary of Measurement Rules for Transactions

Having defined transaction types and parts, we identified the measurement rules for Transaction in each FSM method before starting to map these rules:

- In IFPUG FPA, the functional size of transactional functions is based on the number of DETs, enters or exits the application boundary, and the number of Referenced Files which are ILF read or maintained by a transactional function or an EIF read by a transactional function.
- According to Mk II, the functional size is sum of the size of the input component, the size of the processing component and the size of the output component. The size of the input element is proportional to the number of uniquely processed DET's crossing the application boundary. The size of the processing element is proportional to the number of Primary Entity Types referenced during the execution of the transaction. If the system entity is also referenced, the count is incremented by one. Sub-entity types are also counted separately when different processing logic is used in the transaction. If there is self-referential entity, two entity references are counted. The size of the output element is proportional to the number of uniquely processed DET's crossing the application boundary back.
- The functional size of COSMIC FFP transaction is proportional to the number of data groups. It does not take into account DETs. The size is the sum of Entry, Exit, Read and Write data movements.
- In ARCHI DIM FSM, the functional size of each constituent part is proportional to the number of DETs of their BFC Types. The BFC Types of Interface Part are Read from I/O Device, Write to Volatile Storage, Read from Volatile Storage, Write to I/O Device. The BFC Types of Permanent Data Access/Storage Part are Read from Permanent Storage, Write to Volatile Storage, Read from Volatile Storage, Write to Permanent Storage. The BFC Types of Control Process Part are Read from Volatile Storage and Write to Volatile Storage. The BFC Types of Algorithm / Data Manipulation are Read from Volatile Storage and Write to Volatile Storage.

Transaction Parts

ISO/IEC 14143-5:2004 states that a transaction takes data as input, processes them and outputs data as a result of the processing. Based on the different transaction types and parts described above and ISO/IEC 14143-5:2004, we split a transaction into 3 parts in the unification model as ***Input***, ***Processing*** and ***Output*** and also divide Processing part one more level; ***Maintained***, ***Read***, ***Control***, ***Algorithmic***.

Mapping of the Measurement Rules for Different FSM Methods in Transaction Parts

After dividing transaction into three main parts, we explain these parts and the mappings of the methods one to another and the unification model in detail below. The overview of these mapping is also presented in Table 11 and Table 12. Table 11 includes mapping for IFPUG FPA, Mk II FPA and COSMIC FFP and Table 12 for ARCHI DIM FSM. The reason for presenting a separate table for ARCHI DIM FSM is to make it easy to read.

- **Input Part** represents the user inputs across an application boundary. Input part has two element; the input DETs across an application boundary and the input data groups, which hold these input DETs.
 1. Both the number of Input DETs of Mk II FPA and the number of DETs Read from I/O Device of ARCHI DIM FSM refers to the number of input DETs.
 2. The number of Entries of COSMIC FFP is equal to the number of input data groups.

- **Output Part** represents the outputs across back the application boundary. Output part has two element; the output DETs across an application boundary and the output data groups, which hold these output DETs.
 3. The number of Output DETs of Mk II FPA and the number of DETs Write to I/O Device of ARCHI DIM refers to the number of output DETs.
 4. The number of Exits of COSMIC FFP is equal to the number of output data groups.

5. As the number of DETs of IFPUG FPA is the number of input and output DETs, which enter or exit the application boundary, we can say that it is equal to the number of “union” of Input DETs and Output DETs.
- **Processing Part** holds the processed elements during the execution of the transaction. Processing part has four sub-types; Maintained, Read, Control and Algorithmic. In this part, only ARCHI DIM points to DETs. Other methods consider just referenced data groups. Also, only ARCHI DIM FSM has algorithmic part unlike the others. Mk II FPA and IFPUG FPA contain control data groups implicitly.
 - Maintained Part** has three elements; maintained permanent data groups, DETs write to permanent data groups, DETs read from volatile storage. The permanent data groups shall be indefinite type.
 - Read Part** has three elements; read permanent data groups, DETs read from permanent data groups, DETs write to volatile storage. The permanent data groups shall be indefinite type.
 - Control Part** has four elements; read control data groups, DETs read from control data groups, maintained control data groups, DETs write to volatile storage. The control data groups shall be volatile type.
 - Algorithmic Part** has three elements; read permanent data groups, DETs read from permanent groups, DETs write to volatile storage. The data groups in this part can be indefinite, volatile or transient type.
6. Only ARCHI DIM FSM maps these four types.
7. IFPUG FPA and Mk II FPA do not differentiate between Read or Maintained data groups and Control or Permanent Data Groups. They just consider referenced data groups. If a data group both read and write, they count it once. Therefore, we count the number of the union of Read, Maintained and Control, both Read and Maintained Control, data groups for them. The number of referenced entity types of Mk II FPA is equal to the number of file types referenced of IFPUG FPA and they are the union of these data groups. The EQ transaction types of IFPUG do not write to data functions, it only reads data.
8. The number of Reads of COSMIC FFP is equal to the number of the union read permanent and control data groups.

9. The number of Writes of COSMIC FFP is equal to the number of the union maintained permanent and control data groups.
10. IFPUG FPA, Mk II FPA and COSMIC FFP do not have Algorithmic Part.

The details of the mapping for IFPUG FPA, Mk II FPA and COSMIC FFP are given in Table 11.

As ARCHI DIM FSM needs the largest set of data, the atomic parts of Processing Parts are represented as in ARCHI DIM FSM.

11. Maintained and Read Parts correspond to the Permanent Data Access/Storage Part of ARCHI DIM FSM.
12. Control Part corresponds to the Control Process Part of ARCHI DIM FSM.
13. Algorithmic Part corresponds to the Algorithm / Data Manipulation Part of ARCHI DIM FSM.

The details of the mapping for ARCHI DIM FSM are given in Table 12.

3.4 Summary of the Unification Model

In this chapter, we defined the unification model that we proposed. The common concepts for FSM methods are identified. The terminology mapping for the concepts is accomplished. The different constituent parts of the FSM methods are determined. The rules in the measurement processes of each FSM method are investigated. Consequently, the BFC Types are studied in detail and the commonalities are depicted.

The FSM methods have many common features and it is possible to identify the relations of these concepts and measurement rules. By using these common concepts and rules, the functional size of the software systems can be measured simultaneously by applying different FSM methods to a single source of data. The unification model we proposed involves the largest set of these common concepts and measurement rules.

Table 11 The mapping of the constituent parts of IFPUG FPA, Mk II FPA and COSMIC FFP Methods to the unification model

	INPUT	OUTPUT	PROCESSING			
			Read	Maintained	Control	
	Entry from outside the application boundary	Exit to outside of the application boundary	Read from permanent data groups	Write to permanent data groups	Read from volatile storage	Write to volatile storage
	DET_R_IO, DG_R_IO	DET_W_IO, DG_W_IO	DG_R_PS	DG_W_PS	DG_R_CTR	DG_W_CTR
Mk II FPA	# of Input DETs	# of Output DETs	# of references to Data Entity Types			
	DET_R_IO	DET_W_IO	DG_R_PS Union DG_W_PS		DG_R_CTR Union DG_W_CTR	
COSMIC FFP	# of Entries	# of Exits	# of Reads	# of Writes	# of Reads	# of Writes
	DG_R_IO	DG_W_IO	DG_R_PS	DG_W_PS	DG_R_CTR	DG_W_CTR
IFPUG FPA	Number of DETs		Number of FTRs			
EI, EO	DET_R_IO Union DET_W_IO		DG_R_PS Union DG_W_PS		DG_R_CTR Union DG_W_CTR	
EQ	DET_R_IO Union DG_W_IO		DG_R_PS	-	DG_R_CTR	-
DET_R_IO : DETs read from I/O device, DET_W_IO : DETs write to I/O device, DG_R_PS : Data groups of DET_ R_PS DG_W_PS : Data groups of DET_ W_PS			DG_R_IO : Data groups of DET_ R_IO DG_W_IO: Data groups of DET_ W_IO DG_R_CTR : Data groups of DET_ R_CTR DG_W_CTR : Data groups of DET_ W_CTR			

Table 12 Mapping of the ARCHI DIM FSM constituent parts to the unification model

	INPUT		OUTPUT		PROCESSING							
	Entry from outside the application boundary		Exit to outside of the application boundary		Read from permanent data groups		Maintained to permanent data groups		Control		Algorithmic	
	Read from I/O Device	Write to volatile storage	Read from volatile storage	Write to I/O Device	Read from permanent data storage	Write to volatile storage	Read from volatile storage	Write to permanent data storage	Read from volatile storage	Write to volatile storage	Read from volatile storage	Write to volatile storage
ARCHI DIM FSM	Interface				Permanent Data Access/Storage				Business Process			
									Control Process		Algorithmic/Data Manipulation Process	
	Input		Output		Read		Write		Read	Write	Read	Write
	# of DETs Read from I/O Device	# of DETs Write to Volatile Storage	# of DETs Read from Volatile Storage	# of DETs Write to I/O Device	# of DETs Read from Permanent Storage	# of DETs Write to Volatile Storage	# of DETs Read from Volatile Storage	# of DETs Write to Permanent Storage	# of DETs Read from Volatile Storage	# of DETs Write to Volatile Storage	# of DETs Read from Volatile Storage	# of DETs Write to Volatile Storage

CHAPTER 4

TOOL ANALYSIS AND DESIGN

This chapter presents the requirements and the design of the EasyEstimate tool. This tool is the implementation of the unification model we proposed in Chapter 3. Moreover, it is designed to overcome some of the issues of the existing commercial products discussed in Chapter 2.

4.1. Tool Overview

EasyEstimate shall be a FSM tool that supports four different FSM methods, IFPUG FPA, Mk II FPA, COSMIC FFP and ARCHI DIM FSM. It shall be based on the unification model proposed in the previous chapter. The user of the tool is the estimator, who is conducting size estimation/measurement.

The main objective of EasyEstimate is to enable the estimator conduct size estimation/measurement with these selected methods simultaneously time by using the single source of data. It also enables estimating/measuring functional size for each method individually in case if only one of the methods is needed. Additionally, it allows converting the functional size obtained by one method to the other by asking the user additional information for the other method.

In addition, one of the important features of this tool is to keep detailed information on the measurement process instead of collecting data only on the counts of BFC Types from which the functional size is obtained.

This tool can be used by the researchers, who are conducting case studies on different FSM and studying on the convertibility of the measures of different methods, and the organizations applying different FSM methods to measure different types of software systems as well as demanding to compare their results.

Object-oriented analysis and design methodology has been performed during the analysis and design phases of the development. UML is used as the analysis and design notation. The use case diagrams, class diagrams, package diagrams and E-R diagrams were drawn.

The basic usage scenario of the tool is as follows; firstly, the estimator defines project and count information. After data groups and data element types are entered, the estimator defines transactions and transaction parts. Then, transaction parts are made related with the data groups and DETs. Finally, the estimator calculates the functional size for selected methods and can report results.

The details of the features of EasyEstimate are explained in the following sections.

4.2. Product Perspective

EasyEstimate is a client-server application. The user will connect to the application via Graphical User Interface (GUI).

TogetherDesigner was used for modeling requirement and design of the software. Use case diagrams, class diagrams and E-R diagrams were drawn with TogetherDesigner.

EasyEstimate was developed in Java programming language. Java Development Kit (JDK) 1.4.2 was used for java runtime environment. Java Swing API was used for developing GUI. The Model-View-Controller (MVC) pattern was used to separate the user interface of an application from its business logic and data model.

MySQL was used as the database management system. MySQL 3.0.8 Java Database Connectivity (JDBC) driver was utilized to connect to the database management system.

The software products required implementing the system and the libraries used are shown in Table 13.

Table 13 Software Products Required Implementing the Software

Product Name	Version	Description
JBuilder 2005 Enterprise Trial	11.0.236.0	The software development environment to develop the software.
Together Designer 2005	5.4.3	UML Modeling tool to be used to draw requirement and design related diagrams of the software.
MySQL	3.0.8	The Database Management System (DBMS) to perform database transactions.
MS Excel 2003		Used for reporting

4.3. Constraints, Assumptions and Dependencies

The following assumptions have been made for the EasyEstimate software.

- It is assumed that the user of EasyEstimate has knowledge of FSM and the FSM methods included.
- It is assumed that Excel program is installed on the target computer on which the software executed.

4.4. Specific Requirements

This section presents the specific requirements of EasyEstimate software by means of UML use cases. There is only one actor for the tool; Estimator, who measures the software size. The Use Cases (UC) are classified into logical groups. The UC diagrams of each group are given first and thereafter each UC in the diagram is explained in detail.

4.4.1 Project Operations

At the beginning of the measurement, the estimator should define new project to measure by Add New Project UC. The use case diagram for project operations shows the project related operations in Figure 1.

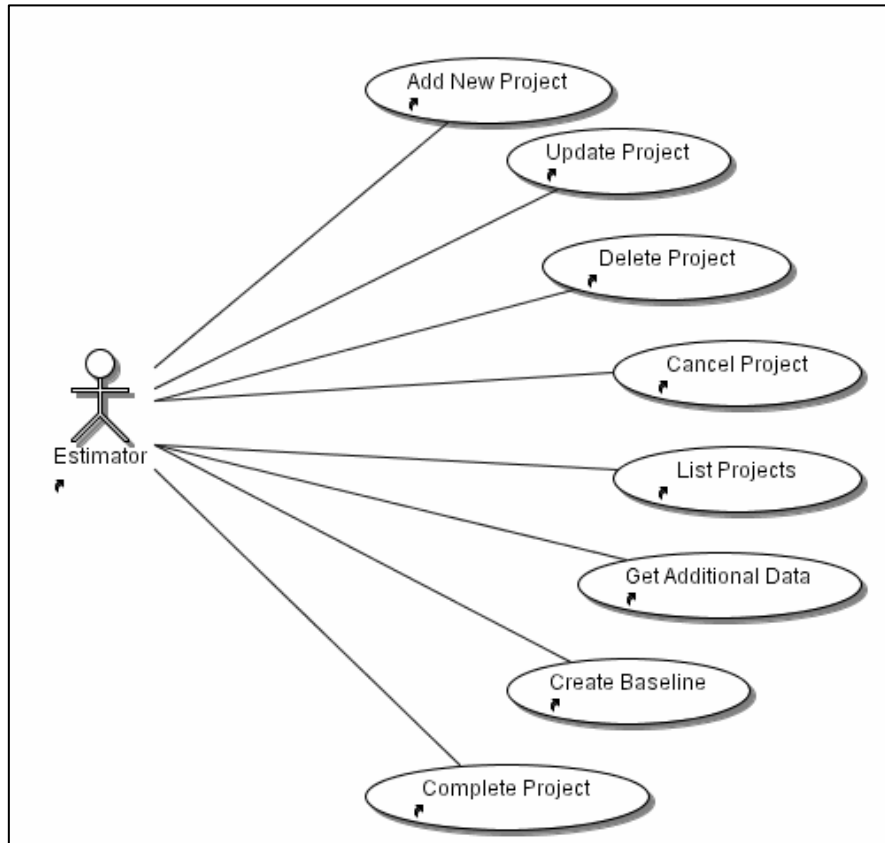


Figure 1 Use Case Diagram for Project Operations

This project information consists of project name, description, created/modified dates, phase names, scope of count, purpose of count, boundary of count, type of count, status of count, FSM methods to be used, and early estimation check.

The count status can be “In Progress”, “Baseline” and “Completed”. The default value is “In Progress”. The baseline for the counts can be created using “Create Baseline” use case. When the status of the project is “Baseline” or “Completed”, the project will be read-only and no information can be changed and related information cannot be added. The version of the count can be assigned during baseline creation.

The FSM methods to be used for the count are selected. According to the selected FSM methods, the estimator will face different kinds of questions and screens during measurement.

The estimator can specify whether this count is early count. Early count means that the estimator has little information about the project details. Only data groups are defined and number of data element types can be described instead of the data element types. If the count is early count, the estimator will have additionally number of count fields.

Having defined project and count, the estimator can enter requirements, data groups, transactions and related information about them. The project is an umbrella entity for all count related data, which are requirements, data groups, data element types, transactions and calculations for the methods.

Add New Project

The estimator selects “New Project” button to add a new project. He/she must provide project name, count type and at least one FSM method to be used. The estimator selects the “Save” button. If any of these mandatory fields is missing, the system warns the estimator and does not realize saving operation. Otherwise, the project is added. At the end, a project is placed in the project tree with the details on project and count panels.

Update Project

The system allows estimator to update project information of an existing project if the project status is “In Progress”. The estimator selects a project within listed projects and then presses the “Update Project” button. The estimator updates information in displayed fields and presses “Save Project” button. The system updates the project information and displays a project form.

Get Additional Data (in Selected FSM Methods Change)

During update operation, if the previously selected FSM methods are modified, the conversion can be made in the data groups of processing part. For

example, Mk II FPA does not distinguish between the read and maintained data groups but COSMIC FFP does distinguish. If firstly measurement for Mk II FPA is made, and the measurement will be extended for COSMIC FFP, the conversion screen provides this operation. By default, all data groups are assumed to be maintained. On this screen, the estimator can select data groups as read or maintained.

Delete Existing Project

The system allows estimator to delete an existing project when that project is not needed anymore, only if there is no dependent baseline for the project. If the project has dependent baseline, first baseline shall be deleted. The administrator selects a project to be deleted from project tree and then presses the “Delete Project” button. The system shows confirmation message for the deletion. The estimator can cancel deletion at this point. Otherwise, selected project is deleted and the project tree that contains the other projects is displayed by the system.

List Projects

The system enables listing defined projects. The projects and versions for the project can be listed in tree structure.

Cancel Operation

Add and update operations for project can be cancelled if the estimator decides to desist from the operation. When the estimator selects “Cancel Project” button, the recent changes are lost and the old values are restored.

Create Baseline

Baseline can be created for the project to freeze available information, only if the project status is “In Progress”. When the estimator selects “Create Baseline” option from menu, the system asks for the version number. The appropriate version number, which is number string with dot (.) separated, shall be provided. The copy of the working project will be created and saved as “Baseline”. All project information, i.e. data groups, DETs, transactions, requirements, count is copied.

The working project remains as “In Progress” till complete operation. The baseline is placed on the project tree in the project list screen.

Complete Count

The count can be completed. After completion, the working project will be frozen and any operation except from delete operations regarding this project and count will not be allowed anymore.

4.4.2 Requirement Operations

The functional user requirements of the project can be defined and linked with transactions. The requirement includes requirement no and description of the requirement. Figure 2 shows the requirement related operations.

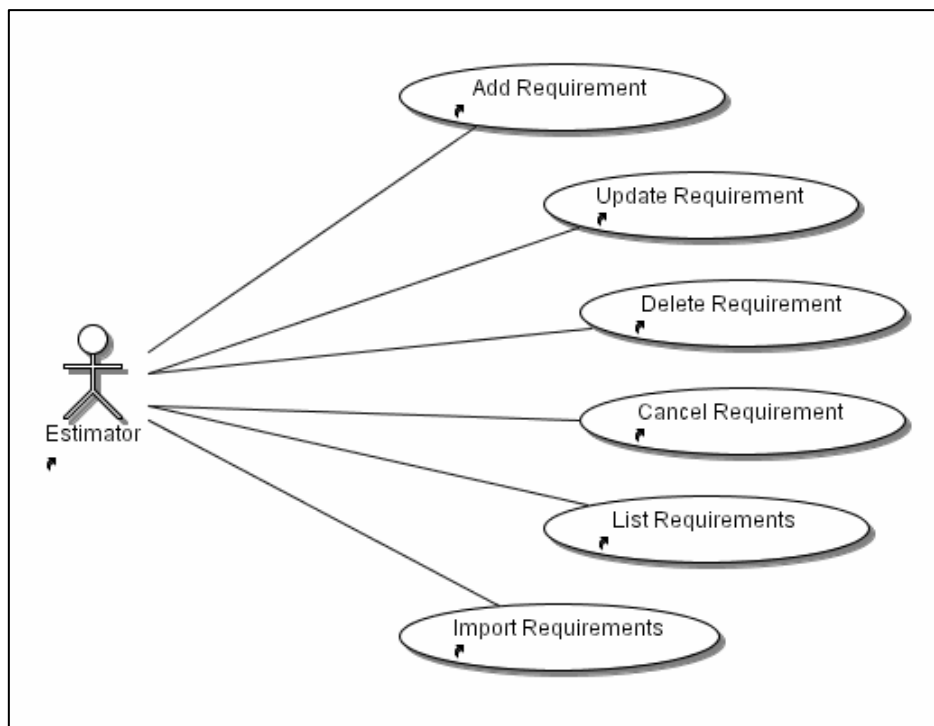


Figure 2 Use Case Diagram for Requirement Operations

Add Requirements

The estimator selects “New Requirement” button to add a new project. Both the requirement number and description of the requirement must be provided. The

estimator selects the “Save” button. The requirement is added to the database and requirement list.

Update Requirements

The system allows estimator to update existing requirement. Both the requirement no and description of the requirement can be changed. The estimator selects a requirement within listed requirements and then presses the “Update Requirement” button. The estimator updates information in displayed fields and presses “Save Requirement” button. The system updates the requirement.

Delete Requirements

The functional user requirements can be deleted with “Delete Requirement” button.

List Requirements

The system provides to list defined functional user requirements.

Cancel Operation

Add and update operations for requirement can be cancelled by “Cancel” button, if the estimator decides to desist from the operation. The recent changes are lost and the old values are restored as a result.

Import Requirements

The functional user requirements can be imported from csv file instead of defining requirements. The file format includes two column; first column is for requirement no and the second one is for requirement description.

For import operation, the estimator select “Import Requirements” button on requirement list screen. Then, the file dialog is displayed to pick up the csv file. After proper file is selected, requirements are imported to the system and displayed in the requirement list.

4.4.3 Data Group Operations

The data groups are one of two fundamental concepts for the measurement. A data group contains data group name, description, persistency type, maintainability, DETs and subgroups. The use case diagram for data group related operations are shown in Figure 3.

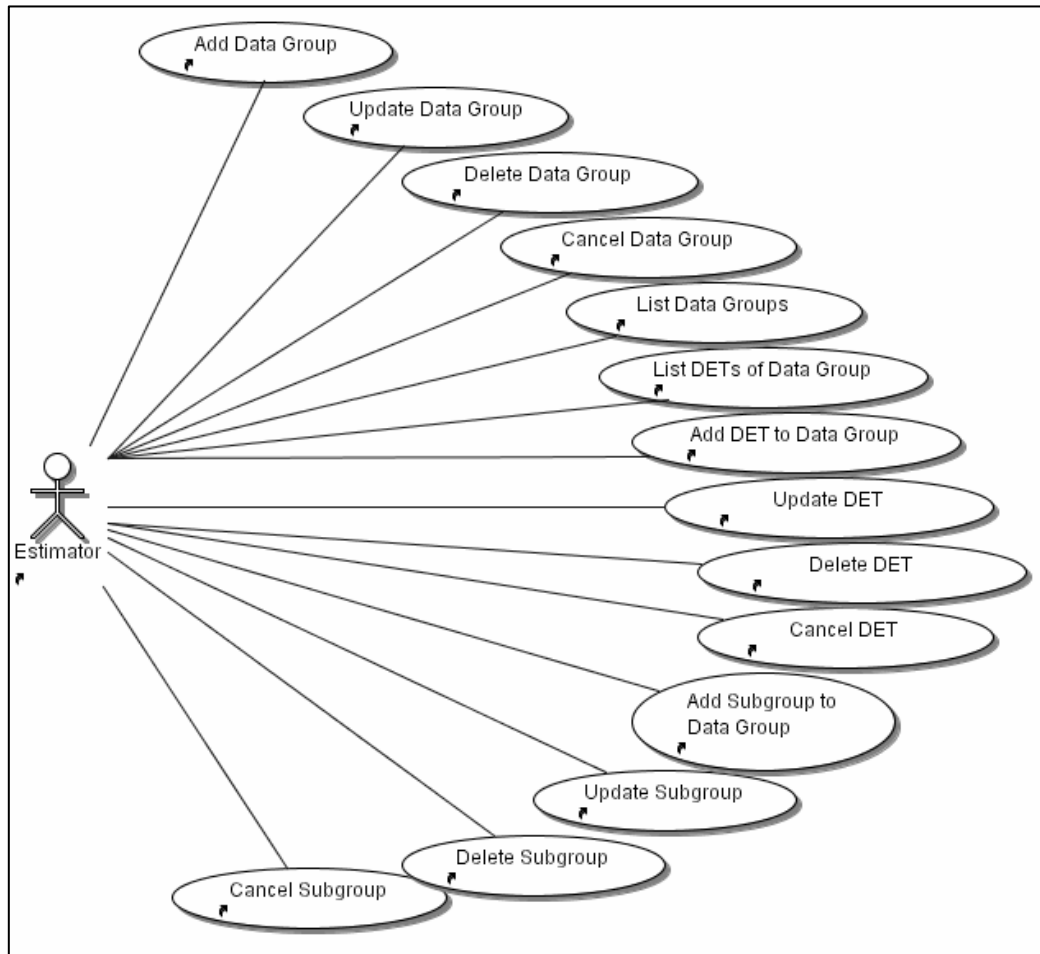


Figure 3 Use Case Diagram for Data Group Operations

Data groups have DETs to hold their data. A data element type should be owned by a data group. DETs include name and description fields. Only DET name is a mandatory field.

If the count is specified as early count, there is one additional field to enter the number of DETs. The estimator can both add DETs and enter the number of DETs for a data group at the same time in early count.

Data groups can have subgroups in case IFPUG FPA is selected as the other methods do not count sub-groups.

Add Data Groups

The estimator selects “New Data Group” button to add a data group. After the mandatory fields, data group name, persistency type and maintainability provided, the estimator selects the “Save” button. If any of these mandatory fields is missing, the system warns the estimator and does not realize saving operation. Otherwise, the data group is added.

As soon as data group created, data element types and subgroups of the data group can be entered. The estimator can enter the number of DETs for a data group if the count is specified as early count.

Update Data Groups

Existing data groups can be modified. The estimator selects a data group within listed data groups and then presses the “Update Data Group” button. Data group name, type and description can be modified. Then the estimator selects “Save Data Group” button. The system updates the data group.

Delete Existing Data Groups

The system allows estimator to delete a data group when that data group is not needed anymore. The estimator selects the data group to be deleted from data group list and then presses the “Delete Data Group” button. The system shows confirmation message for the deletion. The estimator can cancel deletion at this point. Otherwise, selected data group is deleted with the data element types and subgroups of the data group.

List Data Groups

The system provides listing defined data groups. The DETs and subgroups of the Data Group are displayed.

Cancel Data Group

Add and update operations for data groups can be cancelled by “Cancel” button, if the estimator decides to desist from the operation. The recent changes are lost and the old values are restored as a result.

List DETs of the Data Group

The system provides listing DETs of the data group. After select the data group, the DETs of the Data Group are displayed.

Add DET to the Data Group

The system allows adding DET to the existing data group. First, the estimator decides on the data group and selects “New DET” button. After DET name and description provided, the estimator selects the “Save” button. If DET name is missing, the system warns the estimator and does not realize saving operation. Otherwise, the DET is added to the selected data group.

Update DET

Existing DETs can be modified. The estimator selects a DET within listed DETs and presses the “Update DET” button. DET name and description fields can be modified. After the estimator selects “Save DET” button, the system updates DET.

Delete Existing DET.

The system allows estimator to delete a DET of a data group. The estimator selects the DET to be deleted from DET list and then presses the “Delete DET” button. The system shows confirmation message for the deletion. The estimator can cancel deletion at this point. Otherwise, selected DET is deleted and the other DETs of data group are listed.

Cancel DET

Add and update operations for DETs can be cancelled by “Cancel” button, if the estimator decides to desist from the operation. The recent changes are lost and the old values are restored as a result.

List Subgroups of the Data Group

The system provides listing Subgroups of the data group. After select the data group, the Subgroups of the Data Group are displayed.

Add Subgroup to the Data Group

The system allows adding subgroups to the existing data group. First, the estimator decides on the data group and selects “New Subgroup” button. After Subgroup name and description provided, the estimator selects the “Save” button. If subgroup name is missing, the system warns the estimator and does not realize saving operation. Otherwise, the subgroup is added to the selected data group.

Update Subgroup

Existing Subgroups can be modified. The estimator selects a Subgroup within listed Subgroups and then presses the “Update Subgroup” button. Subgroup name and description fields can be modified. After the estimator selects “Save Subgroup” button, the system updates Subgroup.

Delete Existing Subgroup

The system allows estimator to delete a Subgroup of a data group. The estimator selects the Subgroup to be deleted from Subgroups list and then presses the “Delete Subgroup” button. The system shows confirmation message for deletion. The estimator can cancel deletion at this point. Otherwise, selected Subgroup is deleted and the other Subgroups of the data group are listed.

Cancel Subgroup

Add and update operations for Subgroups can be cancelled by “Cancel” button, if the estimator decides to desist from the operation. The recent changes are lost and the old values are restored as a result.

4.4.4 Transaction Operations

The transactions are the second fundamental concepts for the measurement.

A transaction contains transaction no and description. It also has transaction type field only if IFPUG FPA is chosen as FSM method.

A transaction has three main parts, Input, Processing and Output like the unification model. In order to provide simplicity, processing part can have sub-parts according to chosen FSM method(s);

- If only IFPUG FPA or/and Mk II FPA are selected, processing part has only one group, Referenced Data Groups.
- If only COSMIC FFP is selected, processing part has two parts; Read Data Groups, Maintained Data Groups.
- If ARCHI DIM FSM is selected, processing part has two parts; Read Data Groups, Maintained Data Groups, Control Data Groups and Algorithmic Data Groups.

Transaction shall have referenced data groups and data element types. Based on the chosen FSM methods, the related forms appear different;

- If only IFPUG FPA or/and Mk II FPA are selected, processing part will not be related to DETs, instead only Referenced Data Groups.
- If only COSMIC FFP is selected, only data groups will be referenced to the transactions.

The counts are calculated according to this information. In early count, transaction may have the number of DETs instead of referencing all DETs individually.

The use case diagram for transaction operations are shown in Figure 4.

Add Transaction

The estimator selects “New Transaction” button to add a transaction. After transaction no and description provided, the estimator selects the “Save” button. The transaction no is mandatory field; so if it is missing, the system warns the estimator and does not realize saving operation. Otherwise, the transaction is saved. After

transaction created, referenced data groups and data element types of the transaction shall be defined.

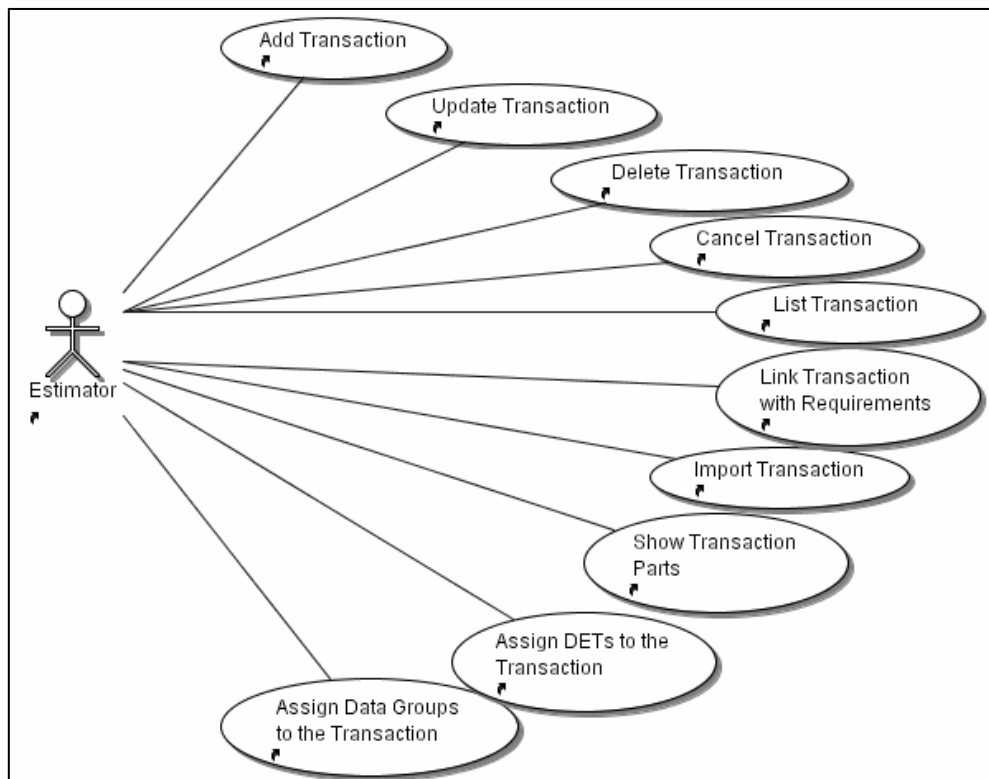


Figure 4 Use Case Diagram for Transaction Operations

Update Transaction

Existing transactions can be modified. The estimator selects a transaction within listed transactions and then presses the “Update Transaction” button. Transaction no, description and type can be modified. Then the estimator selects “Save Transaction” button. The system updates the transaction.

Delete Existing Transaction

The system allows estimator to delete a transaction when that transaction is not needed anymore. The estimator selects the transaction to be deleted from transaction list and then presses the “Delete Transaction” button. The system shows confirmation message for the deletion. The estimator can cancel deletion at this point. Otherwise, selected transaction is deleted with the referenced data groups and data element types of the transaction.

List Transactions

The system provides listing defined transactions.

Cancel Operation

Add and update operations for transactions can be cancelled by “Cancel” button, if the estimator decides to desist from the operation. The recent changes are lost and the old values are restored as a result.

Import Transactions

The transactions can be imported from csv file instead of defining one by one. The file format includes two column; first column is for transaction no and the second one is for transaction description. In the first place, the estimator select “Import Transactions” button. Then, the file dialog is displayed to pick up the csv file. After proper file is selected, transactions are imported to the system and displayed in the transaction list.

Link Transactions with Requirements

The system allows the estimator linking transactions with requirements to provide backward-traceability. When the estimator selects a transaction and presses “Link with Requirements” button, the new window for this linkage process will be appeared. The present requirements, which have been already related with the selected transaction, are listed on the window. The estimator selects requirements and presses “Link” button. The estimator can relate one transaction with more than one requirement.

Show Transaction Parts

The transaction has different parts according to selected FSM methods. When the estimator selects “Transaction Contribution” button, transaction contribution window is displayed with relevant transaction parts of the selected FSM method(s) as explained in Transaction Operations section.

In each transaction part, the data groups are referenced and data element types are assigned individually.

Assign Data Groups to the Transaction

For assignment of data groups, the estimator selects the desired part on Transaction Contribution window. The new window for the chosen part is display. This window contains all data groups, assigned data groups and assigned data element types. The estimator can assign one data group with “Assign Data Group” button or assigns all existing data groups with “Assign All Data Groups” button. Also, he/she can remove one assigned data group with “Remove Data Group” button, or all assigned data groups with “Remove All Data Groups” button.

Assign DETs to the Transaction

The window for chosen part includes “Assign DETs” button. When the estimator selects a data group and presses this button, the window including all DETs of the data groups and assigned DETs of this data group is displayed. The estimator can assign one DET with “Assign DET” button or assigns all existing DETs of the data group with “Assign All DETs” button. Also, he/she can remove one assigned DET with “Remove DET” button, or all assigned DETs with “Remove All DETs” button.

If the count is specified as early count, the window will include one additional field to enter the number of DETs. The estimator can both assign DETs and enter the number of DETs at the same time in early count.

After all DETs assigned and “Save” button selected, DETs are saved and the assigned DETs can be seen in the window for chosen part as assigned DETs.

4.4.5 Counting Operations

The functional sizes for each method are measured according to the manuals of the FSM methods and the model that is proposed in Chapter 3.

The use case diagram for data group related operations are shown in Figure 5.

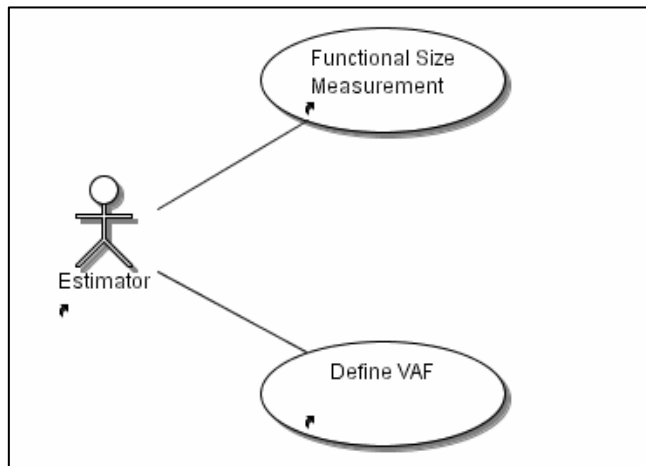


Figure 5 Use Case Diagram for Counting Operations

Functional Size Measurement

The functional size is calculated for the selected FSM methods in project. The “Calculate Functional Size” button is triggered to start functional size calculation. After calculated, the window including functional sizes of selected FSM methods will be shown. The calculations are made according to the unification model proposed in the previous chapter.

Additionally, regarding calculated value adjustment factors, the adjusted FP are calculated for IFPUG FPA and Mk II FPA.

Define VAF

The system allows defining value adjustment factor only for IFPUG FPA and Mk II FPA measurements. The estimator presses “Value Adjustment Factors” and the list of general system characteristic is displayed. If Mk II FPA is not selected as FSM method, this list includes only 14 general system characteristics. If Mk II FPA is selected, it contains 19 characteristics.

The estimator rates these characteristics and saves with “Save” button. Then separate value adjustment factors for IFPUG FPA and Mk II FPA are calculated and displayed to the estimator.

4.4.6 Reporting Operations

The system provides five different Excel reports. The estimator can have separate summary reports for IFPUG FPA, Mk II FPA, COSMIC FFP and ARCHIDIM FSM with the specified format in manuals and one detailed transaction report including their contributions for covered FSM methods (see Figure 6).

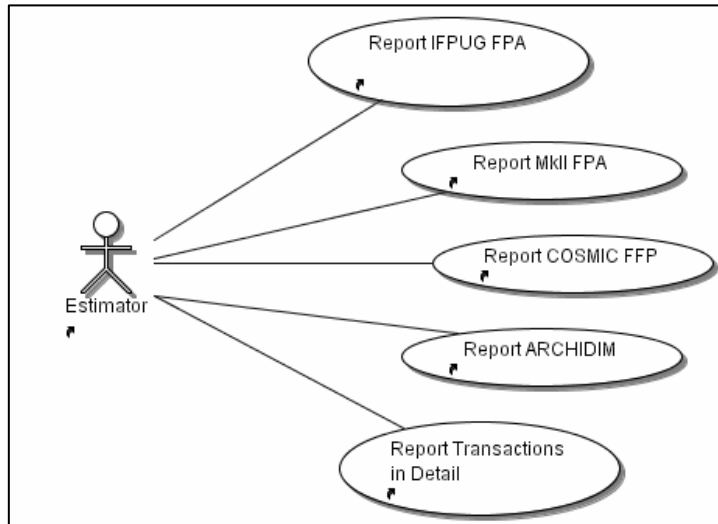


Figure 6 Use Case Diagram for Reporting Operations

Report IFPUG FPA Results

The estimator gets IFPUG FPA summary report via “Report” button on IFPUG FPA part of calculation results window.

Report Mk II FPA Results

The estimator gets Mk II FPA summary report via “Report” button on Mk II FPA part of calculation results window.

Report COSMIC FFP Results

The estimator gets COSMIC FFP summary report via “Report” button on COSMIC FFP part of calculation results window.

Report ARCHIDIM FSM Results

The estimator gets ARCHIDIM FSM summary report via “Report” button on ARCHIDIM FSM part of calculation results window.

Report Transactions in Detail

The estimator can have one joint detailed report for IFPUG FPA, Mk II FPA, COSMIC FFP and ARCHI DIM FSM including transactions and their contributions.

4.4.7 Estimation Effort Operations

The system provides to collect estimation effort utilized for FSM (see Figure 7). The estimator should record start date-time, end date-time, number of estimators and description of work for each effort record. The total effort is shown. This effort information can be used for statistical purposes.

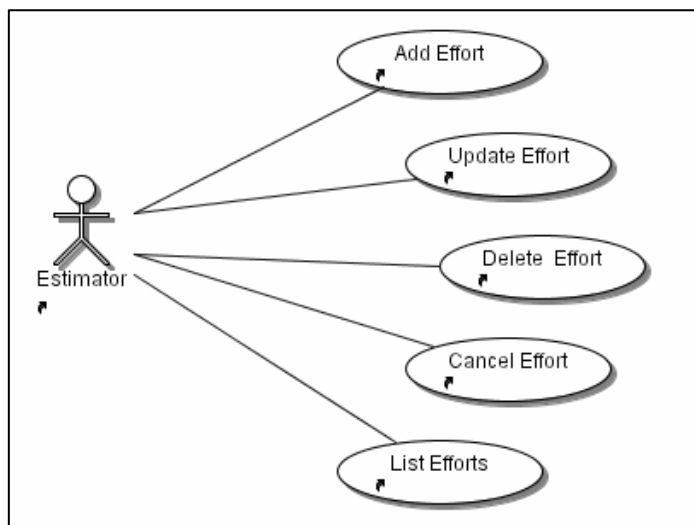


Figure 7 Use Case Diagram for Estimation Effort Operations

Add Effort

The estimator selects “New Effort” button. After start date-time, end date-time, number of estimators and description of work provided, the estimator selects the “Save” button. The time information is saved and based on given information, the system calculates duration, effort and total effort.

Update Effort

Existing efforts can be updated. The estimator selects an effort line within listed effort records and then presses the “Update New Effort” button. Any field of the effort can be modified. Then the estimator selects “Save Effort” button. The system updates the record and related effort calculations.

Delete Existing Effort

The system allows estimator to delete an existing effort record. The estimator selects the record to be deleted from estimation effort history list and then presses the “Delete Effort” button. As a result, selected effort information is deleted.

List Efforts

The system enables to list the effort history spent during measurement and total effort.

4.5. Logical Database Requirements

In the system, all the data will be kept on a relational database management system. Entity relationship diagram is given in Figure 8. Tables with a brief explanation are as follows:

- “PROJECT”: To keep information about the project to be measured.
- “DATA_GROUP”: To keep data groups of the project.
- “DET”: To keep data element types of the data groups of the project.
- “SUBGROUP”: To keep subgroups of the data groups of the project.
- “TRANSACTION”: To keep transactions of the project.
- “TRANSACTION_DATA_GROUP”: To keep data groups related to transactions. Relation table between TRANSACTION and DATA_GROUP tables.
- “TRANSACTION_DET”: To keep data element types related to transactions. Relation table between TRANSACTION and DET tables.
- “REQUIREMENT”: To keep requirements of the project.
- “TRANSACTION_REQUIREMENT”: To keep requirements related to transactions. Relation table between TRANSACTION and REQUIREMENT tables.
- “VAF”: To keep value adjustment factors for IFPUG FPA and Mk II FPA measurements.
- “ESTIMATION_EFFORT”: To keep estimation/measurement effort utilized during measurement.

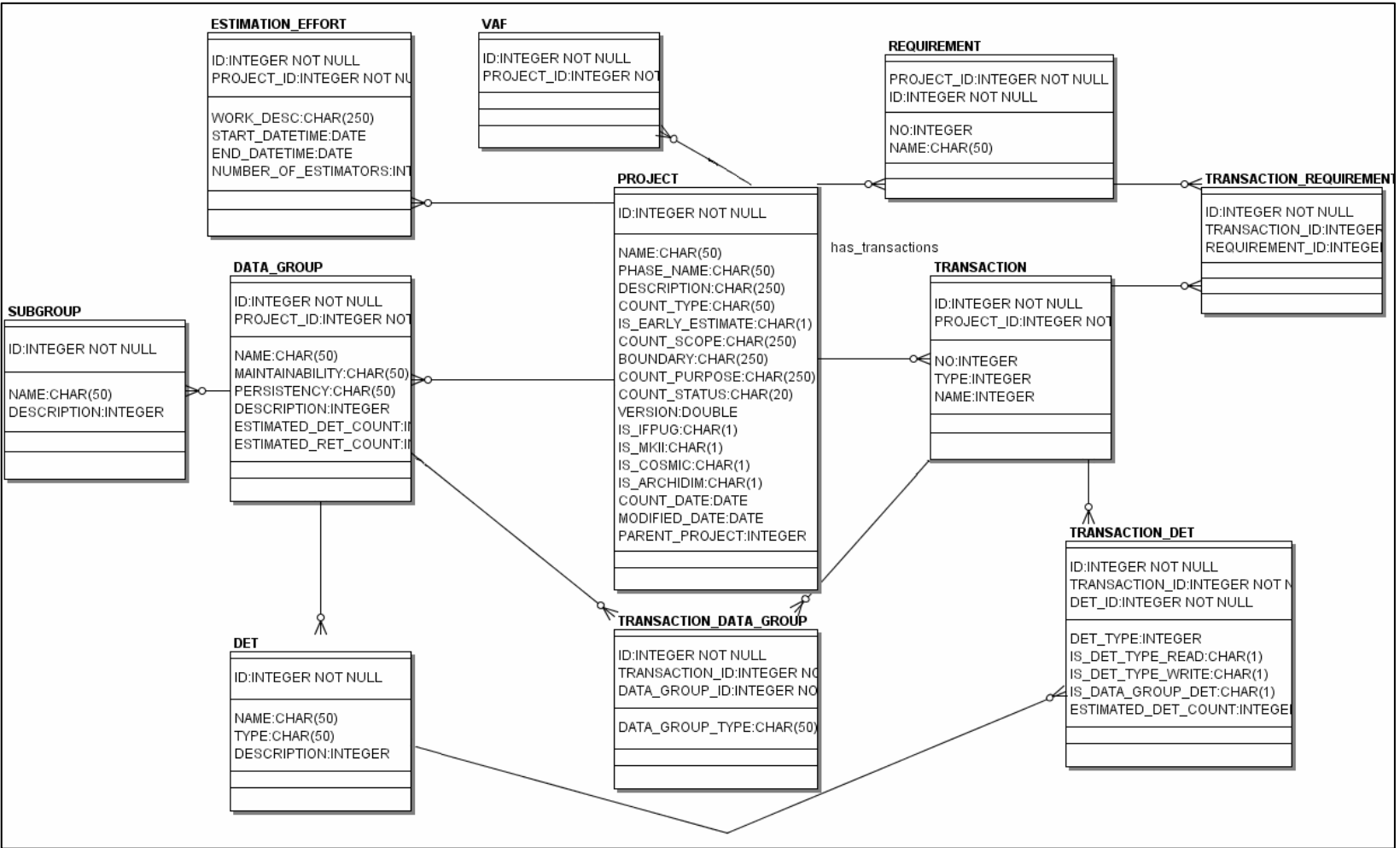


Figure 8 E-R Diagram of EasyEstimate

4.6. Decomposition Description

EasyEstimate software is developed under four packages; “tool.controller”, “tool.domain”, “tool.ui”, and “tool.util”. As MVC pattern are utilized, the java classes belong to different layers, model, view and controller, are kept separate packages. The decomposition of packages and relationship between them is shown in Figure 9.

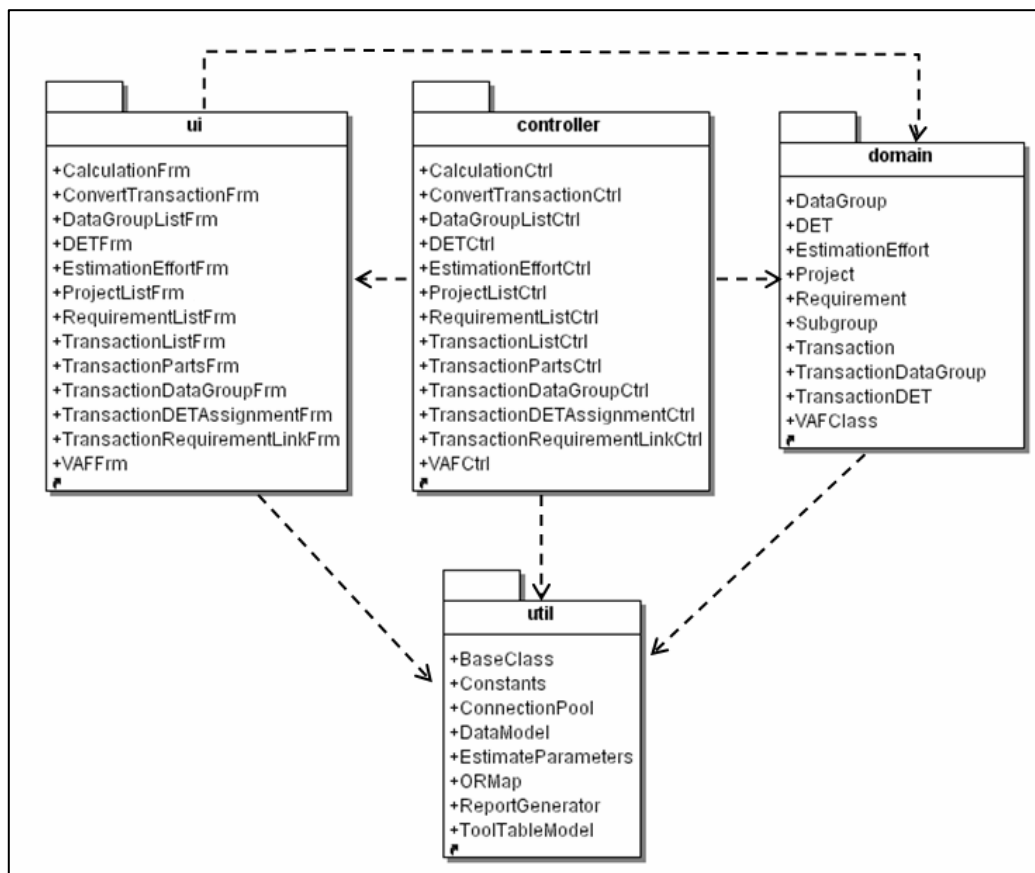


Figure 9 Packages of the tool

In the following, brief overview of the packages and their classes is given. The class diagrams of the EasyEstimate software is given in Figure 10.

“tool.domain” package contains the model layer Java classes that keep data entities of the software. All data classes extend BaseClass class of “tez.util” package. BaseClass provides CRUDL (Create, Read, Update, Delete, and List) operations in a generic way for all domain classes by using Java Reflection API. It utilizes *ORMap*

class of the “tez.util” package to enable object-to-relational mapping of domain classes and database tables.

“tez.ui” package contains the Java classes of view layer, which includes the user interfaces. The UI classes do not have any business logic. They only keep and display data. The main screen for the tool is *ProjectListFrm* (see Figure 11). The project list is can be seen in a tree view. The project related data is defined here. The methods to be used, whether estimation is early estimate can be specified. The main menu is placed at the top this screen. The snapshots of the menu operations can be seen in Appendix B. The data type related data is defined in *DataGroupListFrm* (see Figure 12 and Figure 13), and the transaction list is placed in *TransactionListFrm* (see Figure 15). *ConvertTransactionFrm* makes necessary conversions of transaction related data types between the methods (see Figure 39). The usage scenario is given in detail in Section 4.3.

“tez.controller” package contains the Java classes of controller layer, which realizes application logic of the software. The controller classes generate UI classes. All business logic operations are handled in these classes including validations of the user inputs. Each UI class mentioned in the previous paragraph shall one controller class.

“tez.util” package contains the Java classes that application logic of the software. *BaseClass* is the subclass of the all domain class. It makes all database operations utilizing *ORMap* class. *ORMap* maps domain classes to the database tables. Using this, proper SQL statements are formed using Java Reflection API. Constants class keeps constant values. *ConnectionPool* handles connection creations to the MySQL database in an efficient way. *ReportGenerator* provides generating Excel reports. *DataModel* and *ToolTableModel* are special table models, which extend *DefaultTableModel* of Java.

CHAPTER 5

EMPIRICAL STUDIES

In order to evaluate the unification model and EasyEstimate, we performed two empirical studies; a case study and an experimental study.

A case study is conducted using the data of a real project to evaluate both the proposed unification model and EasyEstimate tool. At first, the functional size of the project is measured manually by applying IFPUG FPA, Mk II FPA, COSMIC FFP and ARCHI DIM FSM methods separately. Then the functional size is measured by using EasyEstimate tool. After that the results are compared and discussed.

Another experimental study is conducted to measure the functional sizes of one of the subsystems of an industrial project by EasyEstimate. The subsystem was already measured applying Mk II FPA, COSMIC FFP and ARCHI DIM FSM methods separately out of this study. In this study, we utilize the functional size measurement data of a subsystem and enter this data to EasyEstimate. Then, the results of EasyEstimate with the previous measurement results are discussed.

5.1. Case Study

5.1.1. Project Characteristics

The case project is a web based, military inventory management project integrated with a document management system.

The project is a data-strong system, which also involves a number of algorithmic operations. The general characteristics of the case project are summarized in the following:

- The development life cycle was Waterfall
- The project staff consisted of 6 people, 1 project manager, 1 senior software engineer, 1 software engineer, 2 part time software engineers and 1 part time test engineer.
- The types of software products and programming language(s) used for the project are; Internal Development Framework and Java as programming languages, IBM WebSphere Application Developer as development environment, Rational Rose as Analysis and Design tool, Oracle 9i as Database Management System, Tomcat as Application Server.
- The project documents were prepared in compliance with the organizational document standards. The company uses an SRS standard developed by the company itself.
- The project was started in October 2004 and completed in December 2005
- The total effort spent for this project is approximately 7500 man-hour.
- Both size estimations presented in Section 5.1.2 and Section 5.1.3 are conducted using Software Requirements Specification (SRS) document of the project, which involves 123 UCs.

5.1.2. Manual Size Measurement of the Case Project

In the manual measurement part, two people performed the size measurement. One of the them works for the development organization and involved in this project. The other one is the creator of the ARCHI DIM FSM method. Both of them are experienced in using the methods, but they are not certified by IFPUG, UKSMA and COSMIC. Total effort spent for manual measurement is 110 person-hours.

IFPUG FPA, Mk II FPA, COSMIC FFP and ARCHI DIM FSM methods were implemented separately. The results were kept in Excel sheets.

By applying IFPUG FPA, the functional size of the project is measured as 925 IFPUG FP. The details of the size measurement are given in Table 14.

Table 14 Case Project IFPUG FPA Size Measurement Details

Number of Elementary Processes	Functional Complexities for Function Types					Total Functional Complexity
	ILFs	EIFs	EIs	EOs	EQs	
123	294	0	262	343	26	925

By applying Mk II FPA, the functional size of the project is measured as 1,330.00 Mk II FP. The details of the size measurement are given in Table 15.

Table 15 Case Project Mk II FPA Size Measurement Details

Number of Logical Transactions	Number of Input DETs	Number of Output DETs	Number of Data Entity Types Referenced	Functional Size (Mk II FP)
123	559	1,679	343	1,330.14

By COSMIC FFP, the functional size of the project is measured as 1,060.0 Cfsu. The details of the size measurement are given in Table 16.

Table 16 Case Project COSMIC FFP Size Measurement Details

Number of Functional Processes	Number of Entries	Number of Exits	Number of Reads	Number of Writes	Functional Size (Cfsu)
123	206	364	334	156	1,060.0

By applying ARCHI DIM FSM, the functional size of the project is measured as 4,476 ADfsu for Interface Component, 1 ADfsu for Control Process Component, 291 ADfsu for Algorithmic / Data Manipulation Process Component and 5,004 ADfsu for Permanent Data Access / Storage Component. The details of the size measurement are given in Table 17.

Interface Component				Interface Functional Size (ADfsu)
Number of Read DETs from Input/Output Device	Number of Write DETs to Volatile Storage	Number of Read DETs from Volatile Storage	Number of Write DETs to Input/Output Device	
559	559	1,679	1,679	4,476
Control Process Component				Control Process Functional Size (ADfsu)
Number of Read DETs from Volatile Storage		Number of Write DETs to Volatile Storage		
1		0		1
Algorithmic / Data Manipulation Process Component				Algorithmic / Data Manipulation Process Functional Size (ADfsu)
Number of Read DETs from Volatile Storage		Number of Write DETs to Volatile Storage		
168		123		290
Permanent Data Access / Storage Component				Permanent Data Access/Storage Functional Size (ADfsu)
Number of Read DETs from Permanent Storage	Number of Write DETs to Volatile Storage	Number of Read DETs from Volatile Storage	Number of Write DETs to Permanent Storage	
1,874	628	628	1,874	5,004

Table 17 Case Project ARCHI DIM FSM Size Measurement Details

5.1.3. Size Measurement of the Case Project Size using EasyEstimate

The measurement of the project using EasyEstimate is performed by the author of this thesis. The total effort utilized for the FSM is 20 person-hours. As we have already identified data groups, transaction and related data elements in the manual measurement, only the data required by the EasyEstimate are entered.

In the next section, the usage of the tool is explained. As the data of the case project are not public, the FURs are not given in the figures. The measurement results of the case study are provided and finally discussion about the manual and tool-based results is presented.

5.1.3.1. Usage Scenario of the Tool in the Case Study

The size measurement by EasyEstimate was conducted by following steps;

1. By using project operations;
 - New project was created with its general information, count information (see Figure 11).
 - Also, the FSM methods to be used were selected by checking all of them.
2. By using data group operations;
 - Data groups were entered one by one as seen in Figure 12.
 - After data groups defined, data element types were added to the data groups as seen in Figure 13.
 - Subgroups were not defined, as we do not identify any subgroup.
3. By using transactions operations;
 - Transactions were imported from the file. As we have already defined and kept them in the Excel file, we copied transaction numbers and names into the csv file and used to import functionality of our tool (see Figure 14).
 - Imported transactions can be seen in the transaction list in Figure 15.
 - The types of the transactions were selected and transactions updated.
 - The transaction contributions for different types were entered. Transaction parts form contains a tab group with 3 tabs; input, processing and output (See Figure

16, Figure 17 and Figure 18). The elements for these parts are changing according to the chosen FSM method. As all of FSM methods had been selected, largest data set were required in this case study.

- The data groups were related to the transactions in these parts. By clicking “...” button, the form for data group assignment to the transaction was opened (see Figure 19).
- After referenced data groups were linked, the related DETs of these data groups were assigned to the transaction as seen in Figure 20.

4. By using Counting Operations

- After describing all parts, the FP was calculated. The measurements for all types of the methods are made using this information. The calculation results can be seen in Figure 21.

5. By using Reporting Operations;

- The reports for IFPUG FPA, Mk II FPA, COSMIC FFP and ARCHI DIM FSM were obtained, using reporting functionality in the tool. The reports are given in Appendix B.

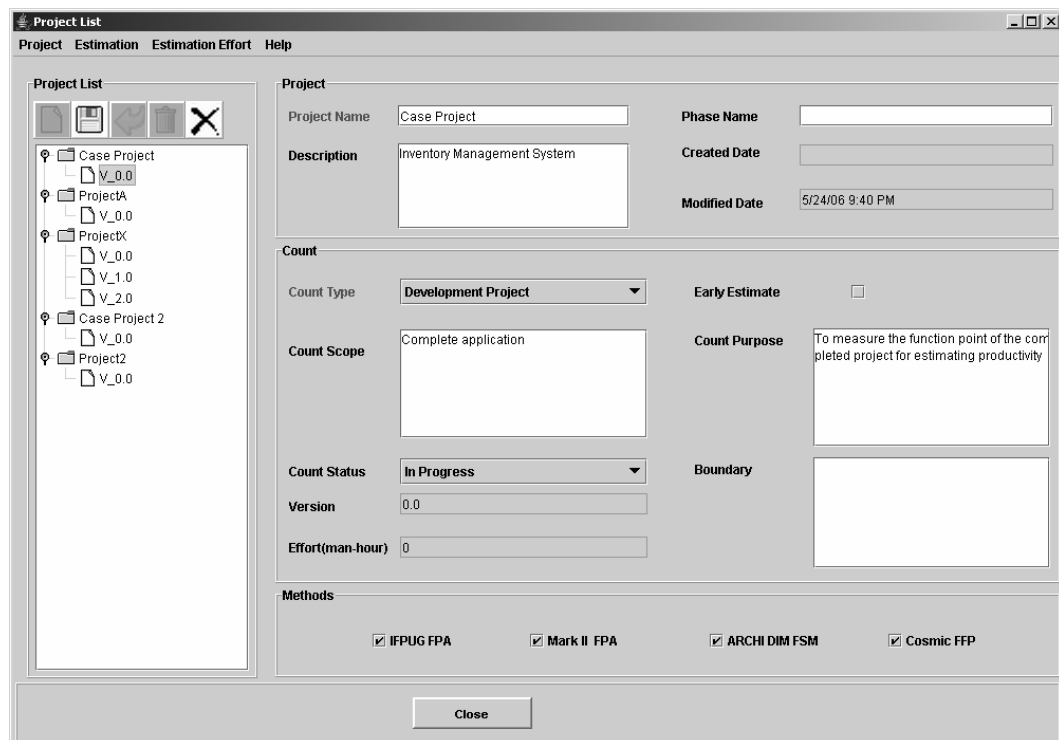


Figure 11 The Project List and Definition

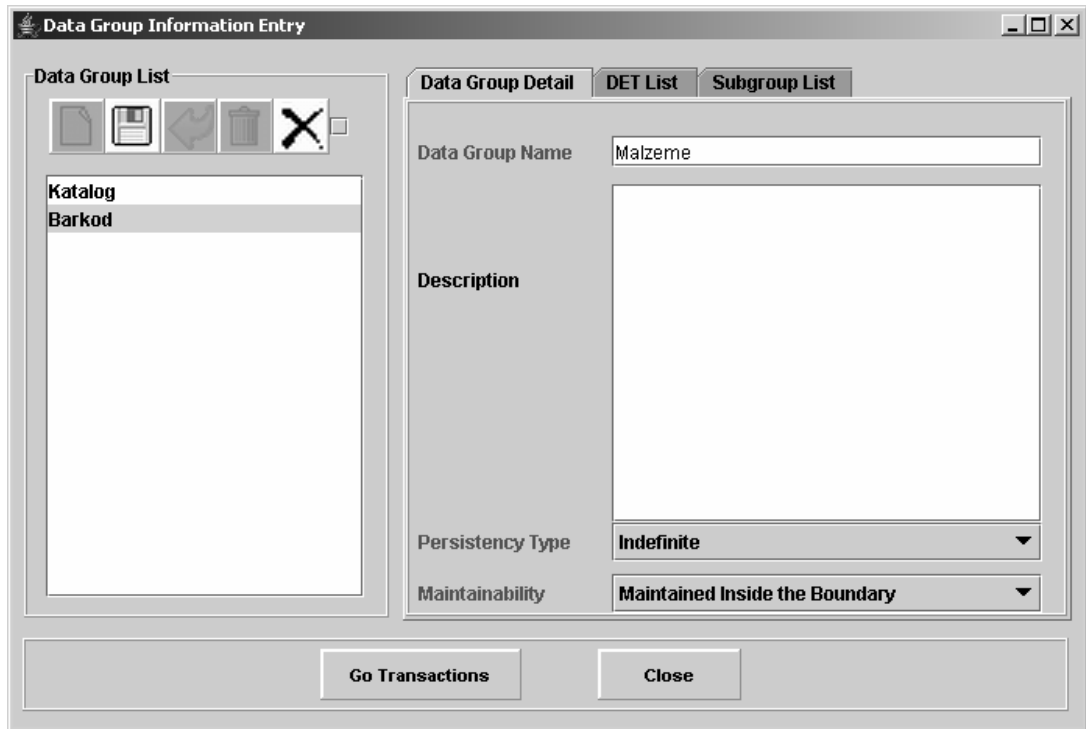


Figure 12 Data Group List and Definition

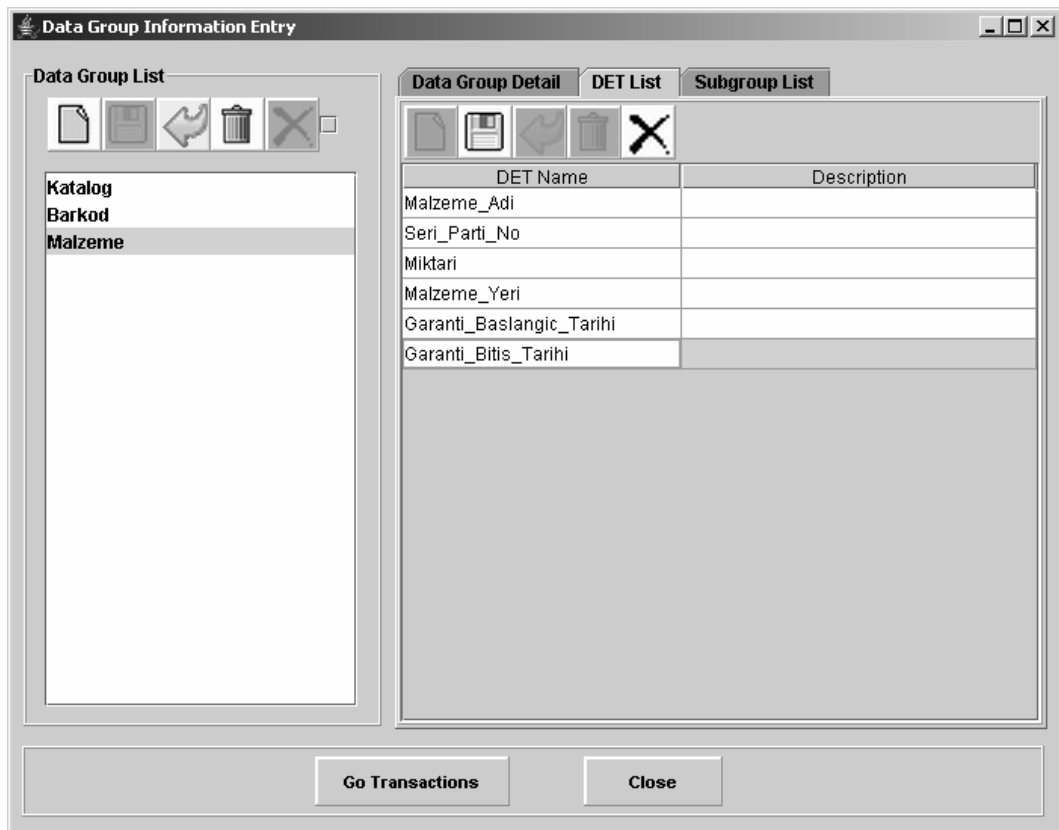


Figure 13 DET List and Definition

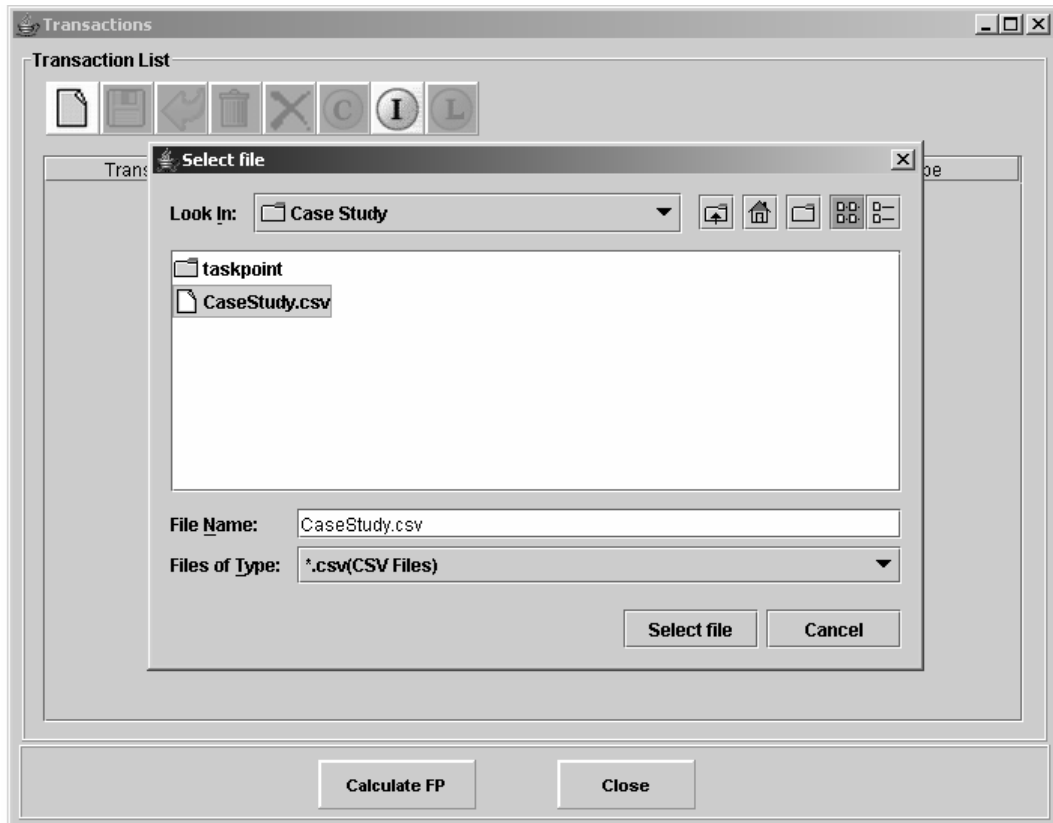


Figure 14 Import of Transactions from CaseStudy.csv file

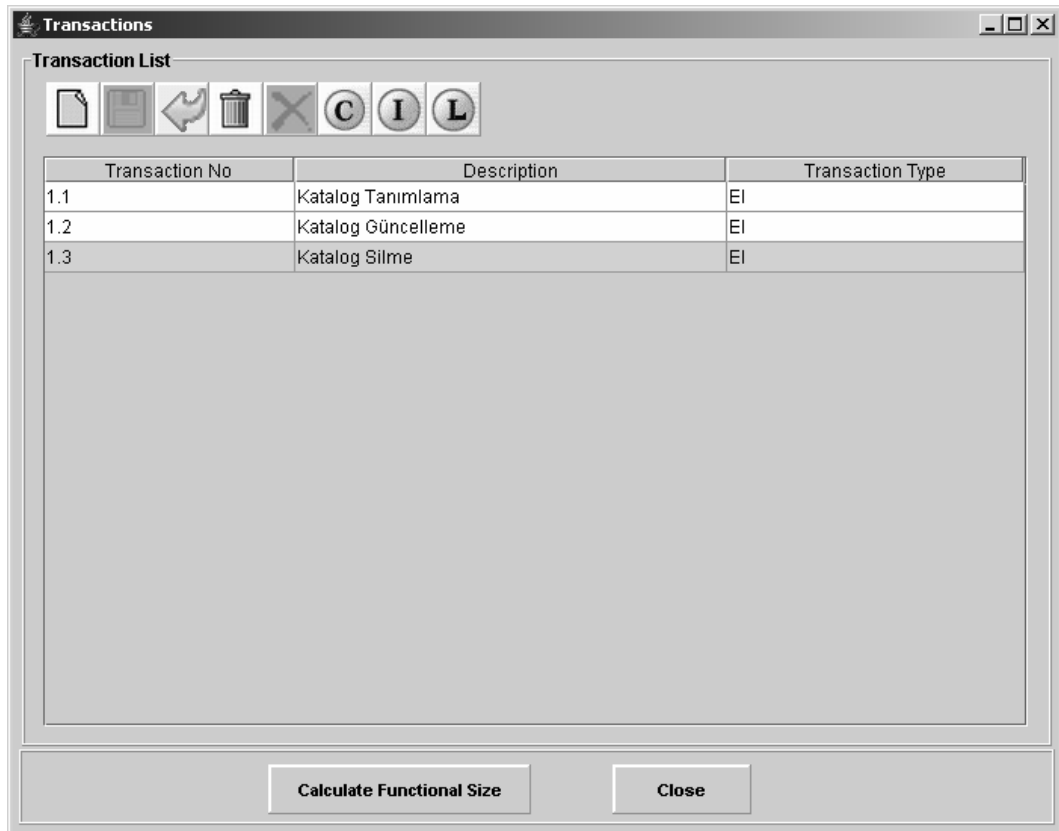


Figure 15 Transaction List

Transaction Parts

Input Processing Output

of DETs Entering Application Boundary 1 ...

of Entering Data Groups 1 ...

Figure 16 Input Part of Transaction (all four FSM methods selected)

Transaction Parts

Input Processing Output

Read Permanent Data Groups

of Read Permanent Data Groups 5 ...

of DETs Read From Permanent Storage 39 ...

of DETs Write to Volatile Storage 39

Maintained Permanent Data Groups

of Maintained Permanent Data Groups 3 ...

of DETs Write to Permanent Storage 31 ...

of DETs Read From Volatile Storage 31

Control Part

of Read Control Data Groups ...

of DETs Read From Volatile Storage ...

of Maintained Control Groups ...

of DETs Write to Volatile Storage ...

Algorithmic Part

of DETs Read From Volatile Storage 8 ...

of DETs Write to Volatile Storage 3 ...

Figure 17 Processing Part of Transaction (all four FSM methods selected)

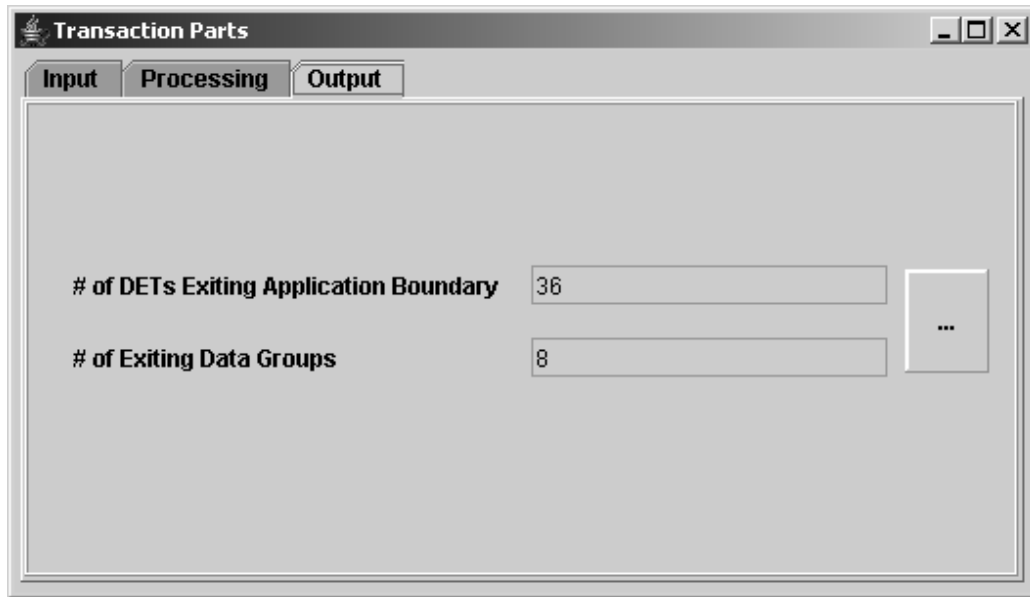


Figure 18 Output Part of Transaction (all four FSM methods selected)

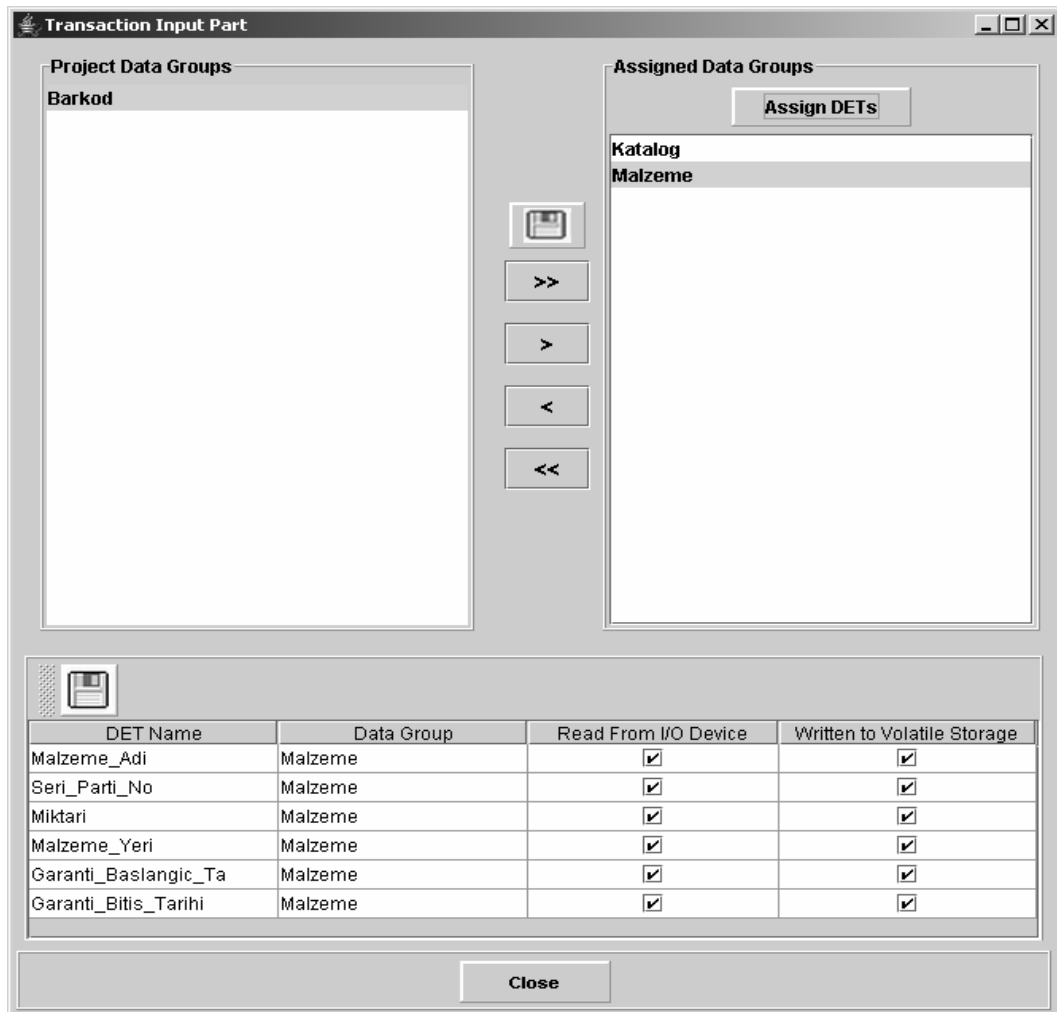


Figure 19 Assigning Data Group to the Transaction (all four FSM methods selected).

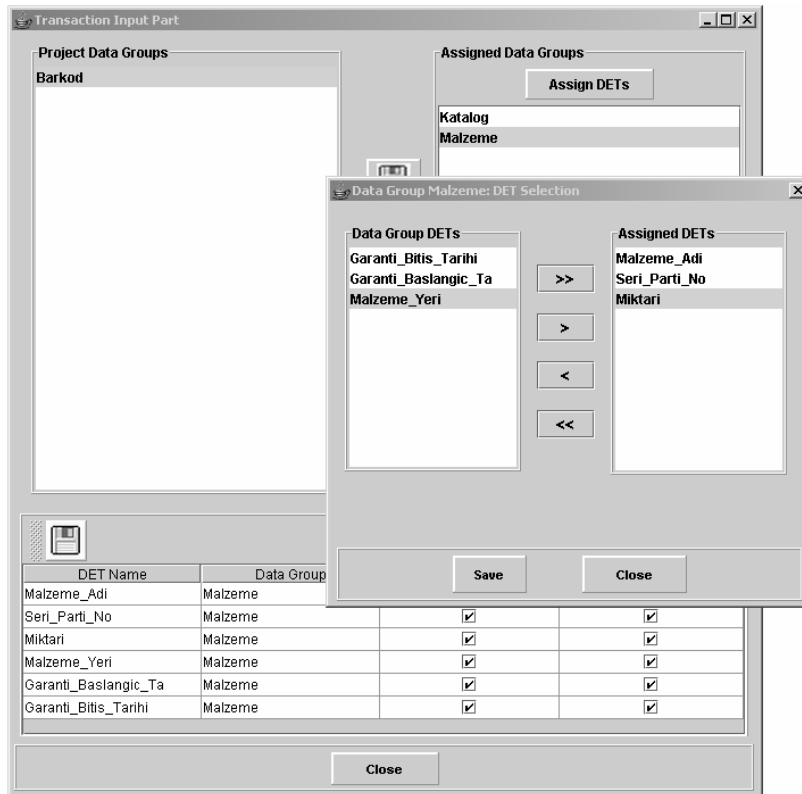


Figure 20 Assigning DET to the Transaction

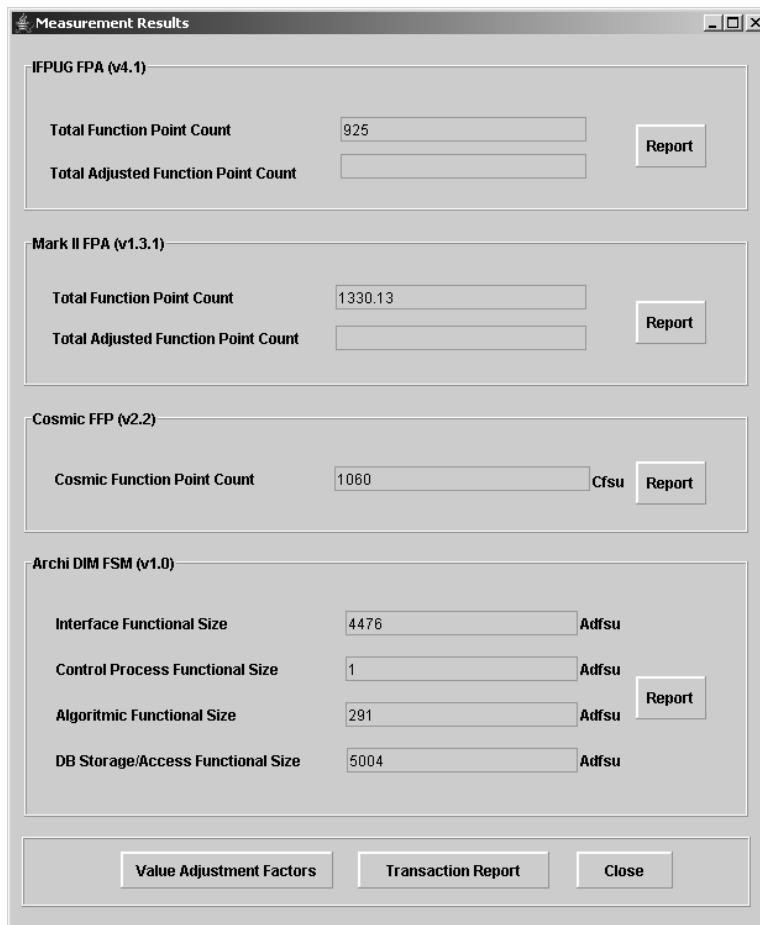


Figure 21 Measurement Results

5.1.3.2. EasyEstimate Measurement Results

The measurement results by IFPUG FPA, Mk II FPA, COSMIC FFP and ARCHI DIM FSM measurements are given in the below tables respectively (see Table 18, Table 19, Table 20, Table 22).

Table 18 Case Project IFPUG FPA Size Measurement Results by EasyEstimate

Number of Elementary Processes	Functional Complexities for Function Types					Total Functional Complexity
	ILFs	EIFs	EIs	EOs	EQs	
123	294	0	262	343	26	925

Table 19 Case Project Mk II FPA Size Measurement Results by EasyEstimate

Number of Logical Transactions	Number of Input DETs	Number of Output DETs	Number of Data Entity Types Referenced	Functional Size (Mk II FP)
123	559	1,679	343	1,330.14

Table 20 Case Project COSMIC FFP Size Measurement Results by EasyEstimate

Number of Functional Processes	Number of Entries	Number of Exits	Number of Reads	Number of Writes	Functional Size (Cfsu)
123	206	364	334	156	1,060.0

5.1.4. Discussion of the Case Study Results

The manual measurement were conducted in 110 person-hours while tool based measurement were performed in 20 person-hours. The first 10 hours of manual measurement was spent on identifying data groups and transactions. The rest 100 hours was spent on measurement. Therefore, manual measurement is 5 times time-consuming than EasyEstimate measurement.

In Section 5.1.2. and 5.1.3., the manual measurement results and tool measurement results of the case study were given consecutively. The comparative total results are given in Table 21. As seen in Table 21, the results are the same. It shows that the unification model is working properly on the data-strong systems.

Table 21 Case Study Results Comparison

	IFPUG FPA	Mk II FPA	COSMIC FFP	ARCHI DIM FSM
Manual Measurement Results	925	1,330.14	1,060	4,476
				1
				291
				5,004
EasyEstimation Measurement Results	925	1,330.14	1,060	4,476
				1
				291
				5,004

When we first performed this measurement, we had slightly different results, e.g. %0.4 error rates. We looked at the details of the measurement using reporting functionality of EasyEstimate, and we understood that these errors were caused by the counting errors in manual measurement. We noticed and corrected 27 errors that we made during counting DETs manually. Moreover, we observed many errors concerning changes in manual measurement. For example, due to a FUR change, a DET was no longer needed but this DET was still existing in some of data groups by mistake. However, in EasyEstimate, it is handled automatically.

5.2. Experimental Study

In this section, the subsystem of a development project is included. The functional size of the subsystem was already measured by Mk II FPA, COSMIC FFP and ARCHI DIM FSM in another study (Gencel, 2005). We performed this experimental study in order to observe how EasyEstimate would behave when measuring a project of another functional domain.

The subsystem is one of the three subsystems of a development project, which is an avionics managements system for small to medium size commercial aircrafts on a Flight Display System. This is a control-strong real-time system which involves intense state transitions, conditional statements, graphical depiction and a number of algorithmic operations.

Table 22 Case Project ARCHI DIM FSM Size Measurement
Results by EasyEstimate

Interface Component				Interface Functional Size (ADfsu)
Number of Read DETs from Input/Output Device	Number of Write DETs to Volatile Storage	Number of Read DETs from Volatile Storage	Number of Write DETs to Input/Output Device	
559	559	1,679	1,679	4,476
Control Process Component				Control Process Functional Size (ADfsu)
Number of Read DETs from Volatile Storage		Number of Write DETs to Volatile Storage		
1		0		1
Algorithmic / Data Manipulation Process Component				Algorithmic / Data Manipulation Process Functional Size (ADfsu)
Number of Read DETs from Volatile Storage		Number of Write DETs to Volatile Storage		
168		123		291
Permanent Data Access / Storage Component				Permanent Data Access/Storage Functional Size (ADfsu)
Number of Read DETs from Permanent Storage	Number of Write DETs to Volatile Storage	Number of Read DETs from Volatile Storage	Number of Write DETs to Permanent Storage	
1,874	1,874	628	628	5,004

The software development organization is a SW-CMM Level 3 company. The project was started in November 2003 and expected to be completed in September 2005. The coding phase was completed and the testing phase has been continuing.

The types of software products and programming language(s) used for the project are Telelogic DOORS for Software Requirements Analysis, Rhapsody for Object Oriented Software Design and Visual Studio C++ for Software Coding.

The project staff consisted of 1 project manager, 1 senior software engineer as a team leader, 6 software engineers, 2 junior engineers, 1 senior software test engineer, 2 junior software test engineers, 1 software quality engineer and 1 software configuration management specialist.

5.2.1. Previous Measurement Results

We have obtained measurement results and detailed measurement data of the subsystem from the study of Gencel (2005). These measurement results by Mk II FPA, COSMIC FFP and ARCHI DIM FSM methods of the subsystem are given in the below tables respectively (see Table 23, Table 24 and Table 28).

Table 23 Mk II FPA Size Measurement Results of Subsystem

Number of Logical Transactions	Number of Input DETs	Number of Output DETs	Number of Data Entity Types Referenced	Functional Size (Mk II FP)
32	112	160	192	425.28

Table 24 COSMIC FFP Size Measurement Results of Subsystem

Number of Functional Processes	Number of Entries	Number of Exits	Number of Reads	Number of Writes	Functional Size (Cfsu)
32	48	32	192	0	272

5.2.2. Subsystem Measurement Results by EasyEstimate

The previously measured subsystem was measured by EasyEstimate. Mk II FPA, COSMIC FFP and ARCHI DIM FSM were selected as FSM methods in EasyEstimate. We entered essential data to EasyEstimate. As the detailed measurement data involves functional sizes of each transaction and explanations for their references, two people completed this measurement in a short time, 5 person-hours. One of them is the author of this thesis study and the other is the person who made the manual measurement that is given in Section 5.2.1. Although both of them are experienced in using the methods, they are not certified by UKSMA and COSMIC.

The results obtained from EasyEstimate measurement are given in Table 25, Table 26 and Table 29.

Table 25 Mk II FPA Size Measurement Results of Subsystem by EasyEstimate

Number of Logical Transactions	Number of Input DETs	Number of Output DETs	Number of Data Entity Types Referenced	Functional Size (Mk II FP)
32	112	160	192	425.28

Table 26 COSMIC FFP Size Measurement Results of Subsystem by EasyEstimate

Number of Functional Processes	Number of Entries	Number of Exits	Number of Reads	Number of Writes	Functional Size (Cfsu)
32	48	32	192	0	272

5.2.3. Discussion of the Results

In this experimental study, we measured the functional sizes of the subsystem that was already measured separately in the traditional way by EasyEstimate. The first measurement was conducted in the scope of another study (Gencel, 2005). We could not find any information about the how much time they utilized during the measurement of subsystem. We utilized 5 person-hours in EasyEstimate measurement.

In Section 5.2.1. and 5.2.2., the manual measurement results and tool measurement results of the subsystem were given consecutively. The comparative total results are given in Table 27.

Table 27 Experimental Study Results Comparison

	Mk II FPA	COSMIC FFP	ARCHI DIM FSM
Manual Measurement Results	425.28	272	544
			128
			336
			400
EasyEstimation Measurement Results	425.28	272	544
			128
			336
			400

As seen in Table 27, the EasyEstimate measurement results are exactly same with the actual measurement results. Therefore, we can say that it is possible to measure a size of real-time software application from a single source of data.

Table 28 ARCHI DIM FSM Size Measurement Results of Subsystem

Interface Component				Interface Functional Size (ADfsu)
Number of Read DETs from Input/Output Device	Number of Write DETs to Volatile Storage	Number of Read DETs from Volatile Storage	Number of Write DETs to Input/Output Device	
112	112	160	160	544
Control Process Component				Control Process Functional Size (ADfsu)
Number of Read DETs from Volatile Storage		Number of Write DETs to Volatile Storage		
128		0		128
Algorithmic / Data Manipulation Process Component				Algorithmic / Data Manipulation Process Functional Size (ADfsu)
Number of Read DETs from Volatile Storage		Number of Write DETs to Volatile Storage		
216		120		336
Permanent Data Access / Storage Component				Permanent Data Access/Storage Functional Size (ADfsu)
Number of Read DETs from Permanent Storage	Number of Write DETs to Permanent Storage	Number of Read DETs from Volatile Storage	Number of Write DETs to Volatile Storage	
200	200	0	0	400

Table 29 ARCHI DIM FSM Size Measurement Results of Subsystem by EasyEstimate

Interface Component				Interface Functional Size (ADfsu)
Number of Read DETs from Input/Output Device	Number of Write DETs to Volatile Storage	Number of Read DETs from Volatile Storage	Number of Write DETs to Input/Output Device	
112	112	160	160	544
Control Process Component				Control Process Functional Size (ADfsu)
Number of Read DETs from Volatile Storage		Number of Write DETs to Volatile Storage		
128		0		128
Algorithmic / Data Manipulation Process Component				Algorithmic / Data Manipulation Process Functional Size (ADfsu)
Number of Read DETs from Volatile Storage		Number of Write DETs to Volatile Storage		
216		120		336
Permanent Data Access / Storage Component				Permanent Data Access/Storage Functional Size (ADfsu)
Number of Read DETs from Permanent Storage	Number of Write DETs to Permanent Storage	Number of Read DETs from Volatile Storage	Number of Write DETs to Volatile Storage	
200	200	0	0	400

CHAPTER 6

CONCLUSION AND FUTURE DIRECTIONS

This thesis focused on the development of the unification model for four selected FSM methods, and a software tool based on the proposed method to overcome the shortcomings of existing functional size estimation/measurement tools.

6.1 Conclusion

Functional size estimation/measurement methods have been studied since their first publication in 1979s. Numerous FSM methods have been proposed up to date.

In this thesis, a comprehensive literature review is performed on the size estimation/measurement methods. The brief information for the well-known FSM methods is presented chronologically. The four selected FSM methods, IFPUG FPA, Mk II FPA, COSMIC FFP and ARCHI DIM FSM, and their measurement processes are explained in detail. The ISO standards for FSM are discussed as well.

A broad literature survey has been conducted on the existing FSM tools. Seven well-known FSM tools, Function Point WORKBENCH, Counter, FP Recorder, PQMPlus, EstimatorPal, MkMAN, µcROSE, are introduced. The short information on these tools and their capabilities are given. The capabilities of these tools are investigated. A checklist that has been established to evaluate and compare the tools is presented as well.

A unification model for the four selected FSM methods is proposed. The concepts for each method, the similarities and differences between the concepts are investigated. The common core concepts and BFC Types; Data Types and Transaction concepts are studied. The counting rules in the measurement processes of the methods are studied comparatively. The mapping for the terminologies of the concepts and the mapping for the constituent part of the methods are presented.

In addition, a software tool, called EasyEstimate, has been developed based on the proposed unification model and the required capabilities of an automated tool derived from the results of the comparison of the existing tools. EasyEstimate has the following capabilities;

- Keeping project information
- Applying one or more FSM methods simultaneously
- Displaying different user interfaces for transaction parts and their constituent parts according to the selected FSM methods
- Keeping data groups, transaction and data element types and their relations
- Keeping project user requirements and their relation with the transactions
- Baselineing /versioning measurement data
- Reporting measurement results
- Keeping measurement effort data

Two empirical studies involving a case study and an experimental study, which are from two different domains; data-strong and control-strong, have been conducted to evaluate the unification model and the usability of the tool. In the case study, a data-strong software system has been measured both manually and using EasyEstimate whereas the functional size of the subsystem of a control-strong avionics project has been measured using EasyEstimate in the experimental study. The manual measurement of this subsystem had been already performed in the scope of another study.

In the case study, firstly the functional size of a real project is measured manually. The results are kept in spread sheets. Afterwards, the size of the project is measured by using EasyEstimate. Then the results of the two measurements are

compared. At first the results seem to be a bit different. When we examined the details of the differences, we observed that we had made some mistakes during manual measurement process even though we spend 5 times more effort with respect to EasyEstimate measurement. We made mistakes especially in counting DETs and changing BFC types. In case one of BFC types change, we had to identify all these BFC types in all related measurement items and change them manually. We spent additional 4 hours for identifying manual measurement errors and correction.

After we corrected these manual measurement mistakes that we noticed, we see that the results are completely same. The total functional size of the case project is 925 IFPUG FP, 1,330.14 Mk II FP, and 1060 Cfsu in both manual measurement and EasyEstimate measurement. Moreover, Applying ARCHI DIM FSM, the project is measured as 4,476 ADfsu for Interface Component, 1 ADfsu for Control Process Component, 291 ADfsu for Algorithmic / Data Manipulation Process Component and 5,004 ADfsu for Permanent Data Access / Storage Component.

In the experimental study, we utilized the detailed manual measurement data of the subsystem. The data was entered to EasyEstimate in a form of unification model. The measurement results of EasyEstimate are exactly the same with the previous results, which are 425.28 Mk II FP and 272 Cfsu for total subsystem and 544 ADfsu for Interface Component, 128 ADfsu for Control Process Component, 336 ADfsu for Algorithmic / Data Manipulation Process Component and 400 ADfsu for Permanent Data Access / Storage Component.

The results of the case study and experimental study show that the unification model and the tool provide multiple measurements at the same time for IFPUG FPA, Mk II FPA, COSMIC FFP and ARCHI DIM FSM methods from single data source.

At the end of this thesis, we see that we have accomplished the four main objectives of this thesis study, which we define in Section 1.3;

- We proposed a unification model in order to measure the software system by any of the selected four methods from the single source of data in a form of unification model.

- We developed a size estimation/measurement tool, EasyEstimate, based on the findings in the unification model and the deficiencies found in the existing tools in order to automate the measurement process of the methods. Before developing this tool, we investigated automation possibilities for the size estimation/measurement processes from this thesis point of view and set up a checklist. Then we evaluated existing commercial estimation/measurement tools according to this checklist.
- The unification model that we proposed and EasyEstimate that we developed allow entering additional data if it is required by one of the methods specifically.
- We validated the unification model and EasyEstimate by conducting two empirical studies. We measured the functional size of a real software project both manually and utilizing EasyEstimate. Also, we entered the measurement data of a subsystem to EasyEstimate, and compared with the actual results.

6.2 Future Work

The proposed model can be extended in several ways. Firstly, the validation of the model can be reinforced by implementing much more case studies using this model. As the case project is data-strong system and the case subsystem is control-strong system, it would be beneficial to implement the projects of different functional domains like algorithmic systems.

The maintenance projects can be considered in the model. Currently, it is proposed only for new development projects.

The model can be linked to the analysis and design elements. If these relations are defined, automation of the multiple FSM methods can be possible from analysis and design elements. Moreover, the tool, EasyEstimate can be integrated with analysis and design tools such as UML modeling tools.

There can be other unification approaches to FSM methods. The proposed model is not the unique unification approach. It is possible to classify BFCs

differently in the unification model and map the BFCs of the FSM models to this BFC Types. For example logical grouping of transactions as Read, Write, Confirm, and Calculate...etc makes sense.

This thesis evaluates unification model and EasyEstimate together. In particular, the unification model must be evaluated on its own, without the EasyEstimate tool, and its validity must be thoroughly investigated.

The comparative manual case studies should be implemented by different persons. In this study, as the author of this thesis and the unification model implemented the manual case studies, the same assumptions of measurement rules in both unification model and the manual measurement were taken into consideration and the results were exactly the same. If another person implemented the manual measurement, it would be very beneficial to improve unification model and their rules.

The tool can be integrated with database management systems. Data groups can be directly retrieved from logical or physical E-R diagrams.

The tool can be improved by adding some query operations, sorting in tables. This would be beneficial and would make estimation much easier for the end-user.

The tool can be integrated with project planning tools. It will provide to use size estimation/measurement results in a project plan.

Online help can be provided.

Finally, additional size estimation methods can be studied in the unification point of view like NESMA, Object-Points...etc.

REFERENCES

- Abran, A., Desharnais, J., Aziz, F., (2005), Measurement Convertibility - From FP to COSMIC-FFP, Proceedings of the 15th International Workshop on Software Measurement, Montreal, Canada
- Albrecht, A.J., (1979), Measuring Application Development Productivity, IBM Applications Development Symposium, Monterey, CA
- Albrecht, A.J. and Gaffney J. E. (1983), Software Function, Source Lines of Code, and Development Effort Prediction: A Software Science Validation, IEEE Transactions on Software Engineering, vol. SE-9, no. 6, November
- ARCHI DIM FSM Measurement Guideline, (2005), Architectural Dimensions Based Functional Size Measurement Method, Measurement Guideline, Version 1.0
- Azzouz, S., Abran, (2004), A. A Proposed Measurement Role in the Rational Unified Process and its Implementation with ISO 19761: COSMIC-FFP, Proceedings of the Software Measurement European Forum - SMEF 2004, Rome, Italy
- Boehm, R. (1997), "Function Point FAQ", Retrieved October 20, 2005 from <http://ourworld.compuserve.com/homepages/softcomp/fpfaq.htm>
- Bundschuh, M., Fabry, A., (2000), Aufwandsschätzung von ITProjekten, MITP-Verlag GmbH, Bonn.
- Candido E., Sanches, R (2004), Estimating the Size of Web Applications by using a Simplified Function Point Method, la-webmedia, pp. 98-105, WebMedia & LA-Web 2004 Joint Conference 10th Brazilian Symposium on Multimedia and the Web 2nd Latin American Web Congress.

COSMIC FFP Measurement Manual, (2003), Cosmic Full Function Points Measurement Manual, Version 2.2

Diab, H., Koukane, F., Frappier, M., St-Denis, R., (2005), μ cROSE: Automated Measurement of COSMIC-FFP for Rational Rose Real Time, Information and Software Technology, 47(3) , pp 151-166.

Fetcke, T., (2001), A Generalized Representation for Selected Functional Size Measurement Methods, International Workshop on Software Measurement (IWSM'01), Montreal, Quebec, Canada

Fetcke, T., (1999), The Warehouse Software Portfolio A Case Study in Functional Size Measurement, Report No. 1999-20

FFP CPM, (1997), Full Function Points Counting Practices Manual, Technical Report 1997-04

Gencel, Ç., (2005), An Architectural Dimensions Based Software Functional Size Measurement Method, A PhD thesis submitted to Middle East Technical University, Ankara, Turkey.

Herron, G. and Garmus, D., (1999), Estimating Software Earlier and More Accurately, Methods & Tools.

IFPUG FP CPM, (2000), International Function Point Users Group (IFPUG) Function Point Counting Practices Manual, Release 4.1.1

ISO/IEC 14143-1, (1998), Information technology, Software measurement, Functional size measurement, Part 1: Definition of concepts

ISO/IEC 14143-2, (2002), Information technology, Software measurement, Functional size measurement, Part 2: Conformity evaluation of software size measurement methods to ISO/IEC 14143-1:1998

ISO/IEC TR 14143-3, (2003), Information technology, Software measurement, Functional size measurement, Part 3: Verification of functional size measurement methods

ISO/IEC TR 14143-4, (2002) Information technology, Software measurement, Functional size measurement, Part 4: Reference model.

ISO/IEC TR 14143-5, (2004) Information technology, Software measurement, Functional size measurement, Part 5: Determination of functional domains for use with functional size measurement.

ISO/IEC FCD 14143-6, (2005) Guide for the Use of ISO/IEC 14143 and related International Standards, 2005.

Jones (1987), A Short History of FP and Feature Points, Software Productivity Research Inc., USA

Lother, M. and 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.

Mendes O., Abran A., Bourque P., (1996), Function Point Tool Market Survey Software Engineering Management Laboratory, Université du Québec à Montréal, Montréal, Canada.

Mk II FPA CPM, (2002), Mark II Function Point Analysis Counting Practices Manual, Version 1.3.1, United Kingdom Software Metrics Association

NESMA Manual, (2005), Nesma Functional Size Measurement Method v.2.1 - Definitions and counting guidelines for the application of Function Point Analysis.

Oligny, S., Abran, A., Desharnais, J.-M., Morris, P., (1998), Functional Size of Real-Time Software: Overview of Field Tests, in Proceedings of the 13th International Forum on COCOMO and Software Cost Modeling, Los Angeles, CA.

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, Los Angeles, CA.

- Özdamar, Türker, (2001), Automating Function Point Analysis in Object Oriented Analysis and Design, A thesis submitted to Middle East Technical University, Ankara, Turkey
- Pressman R., (2001) Software Engineering, A Practitioner's Approach, McGraw-Hill, 5th Edition.
- Rudolph, E., (1997), Understanding Function Point Analysis, Prentice Hall, 1st Edition
- Rule, G. (1999). A Comparison of the Mark II and IFPUG Variants of Function Point Analysis, Retrieved September 10,.2005, from <http://www.gifpa.co.uk/library/Papers/Rule MK2IFPUG.html>.
- St-Pierre D., Maya M., Abran A., Desharnais J.-M., Bourque P., (1997) Full FP: Counting Practices Manual, Technical Report, Université du Québec à Montréal, Montréal, Canada.
- Symons, C. (1999). Conversion between IFPUG 4.0 and Mk II FP, Software Measurement Services Ltd., Version 3.0.
- Symons, C. (2001). Come Back Function Point Analysis (Modernized) – All is Forgiven!, In Proceedings of the 4th European Conference on Software Measurement and ICT Control, FESMA-DASMA 2001, Germany pp. 413-426.
- Whitmire, S.A., (1995) An Introduction to 3D Function Points, Software Development, Volume 3, Issue 4, pp 43-53

APPENDICES

APPENDIX A

ADJUSTMENT FACTORS

Table 30 IFPUG FPA Value Adjustment Factor Calculation Table

General System Characteristics		Degree of Influence(0-5)
1	Data Communications	
2	Distributed Data Processing	
3	Performance	
4	Heavily Used Configuration	
5	Transaction Rate	
6	On-Line Data Entry	
7	End-User Efficiency	
8	On-Line Update	
9	Complex Processing	
10	Reusability	
11	Installation Ease	
12	Operations Ease	
13	Multiple Sites	
14	Facilitate Changes	
Total degree of influence (TDI)		
VAF = TDI * 0.01 + 0.65 =		

Table 31 Mk II FPA Technical Complexity Adjustment Factor Calculation Table

REF	NAME	VALUE
1	Data Communications	
2	Distributed Function	
3	Performance	
4	Heavily Used Configuration	
5	Transaction Rates	
6	On-Line Data Entry	
7	Design for End-User Efficiency	
8	On-Line Update	
9	Complexity of Processing	
10	Usable in Other Applications	
11	Installation Ease	
12	Operations Ease	
13	Multiple Sites	
14	Facilitate Changes	
15	Requirements of Other Applications	
16	Security, Privacy and Auditability	
17	User Training Needs	
18	Direct use by Third Parties	
19	Documentation	
	TOTAL DEGREE OF INFLUENCE	
Total TCA = TDI * 0.005 + 0.65 =		

APPENDIX B

EASYESTIMATE SCREEN SHOTS

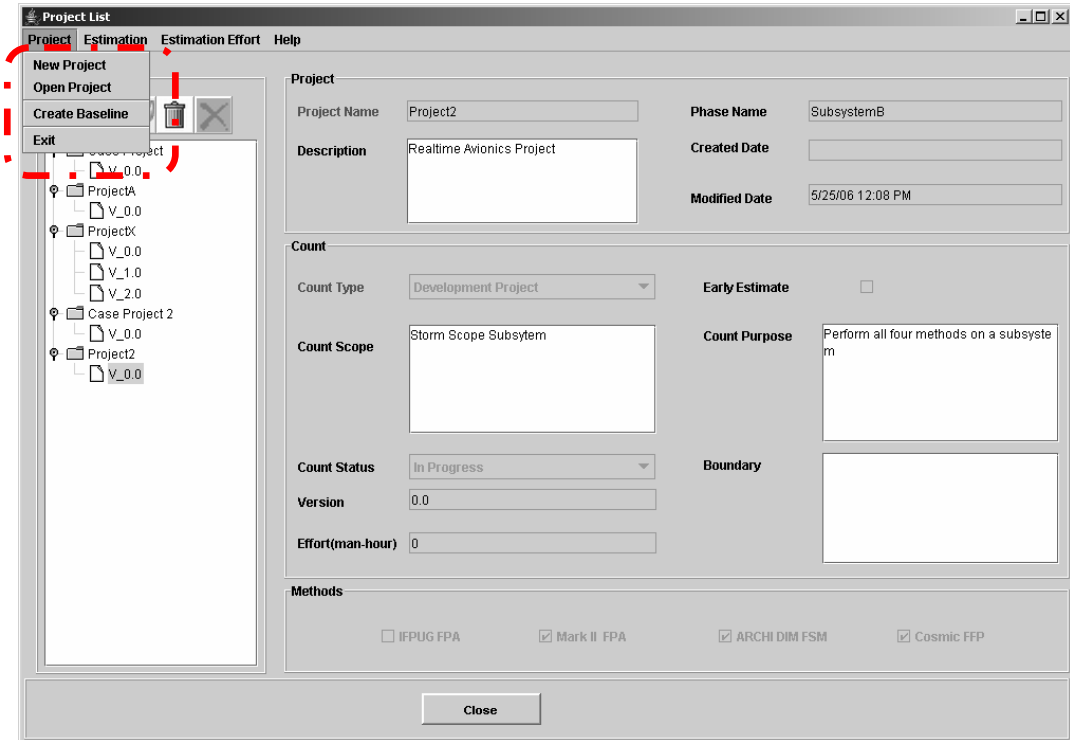


Figure 22 Main Menu and Project Sub Menu

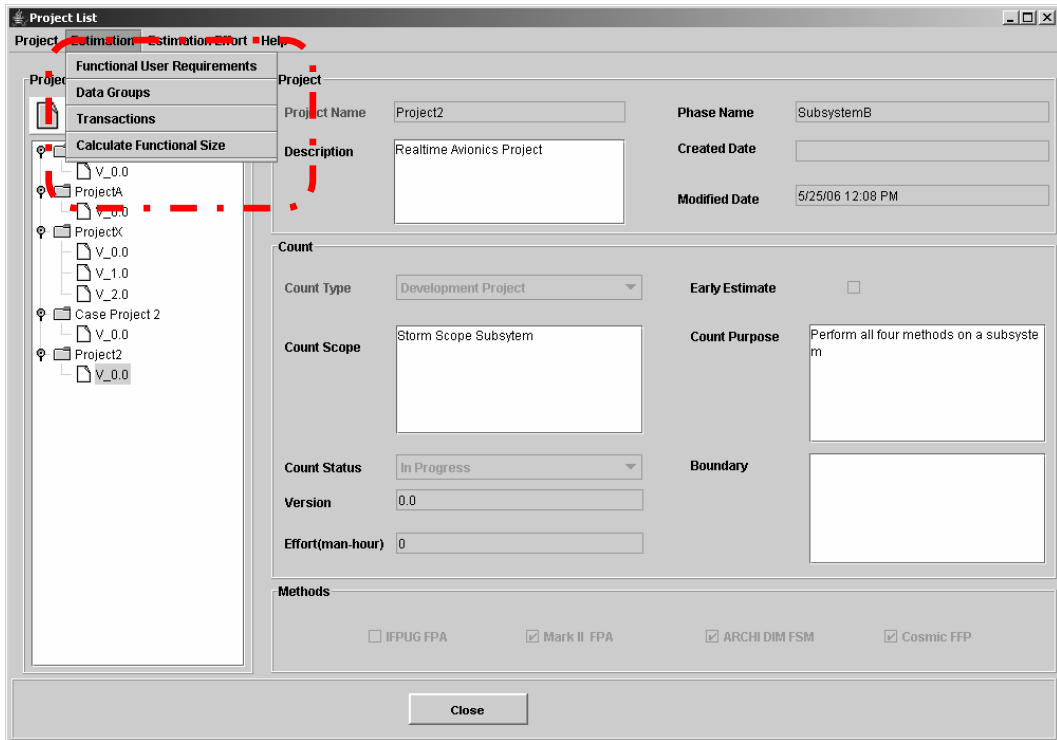


Figure 23 Main Menu and Estimation Sub Menu

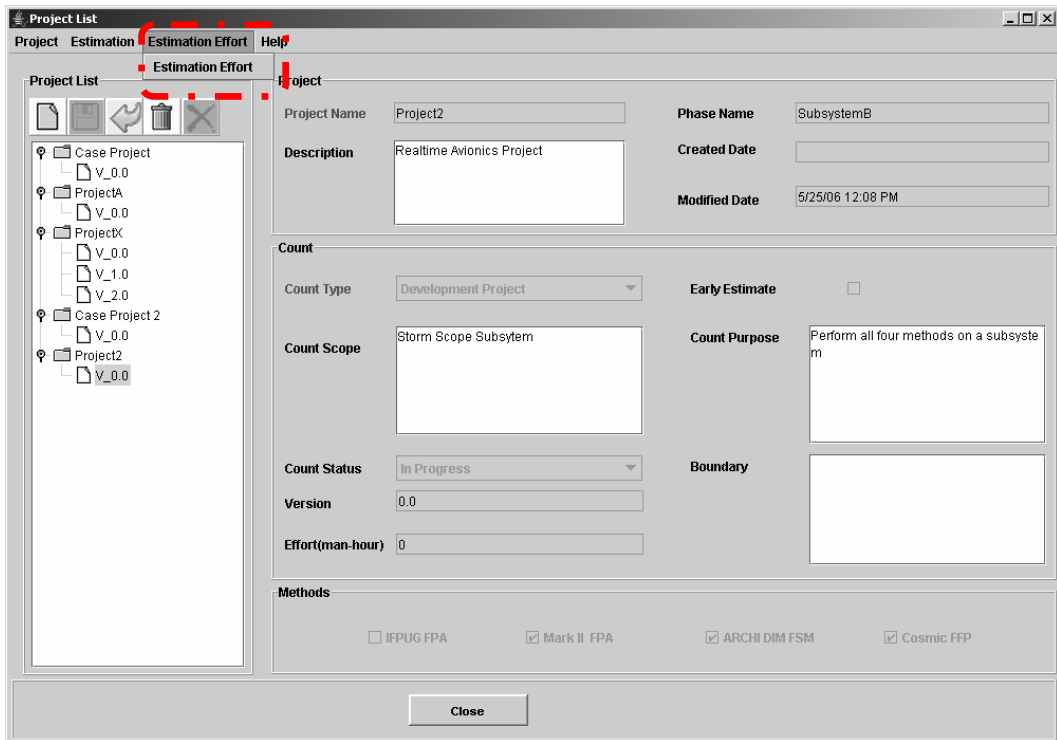


Figure 24 Main Menu and Estimation Effort Sub Menu

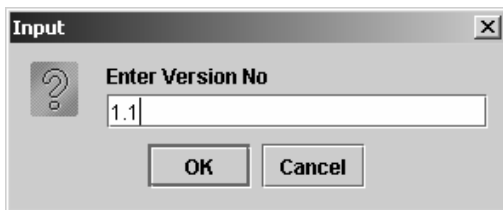


Figure 25 Version Entrance Dialog

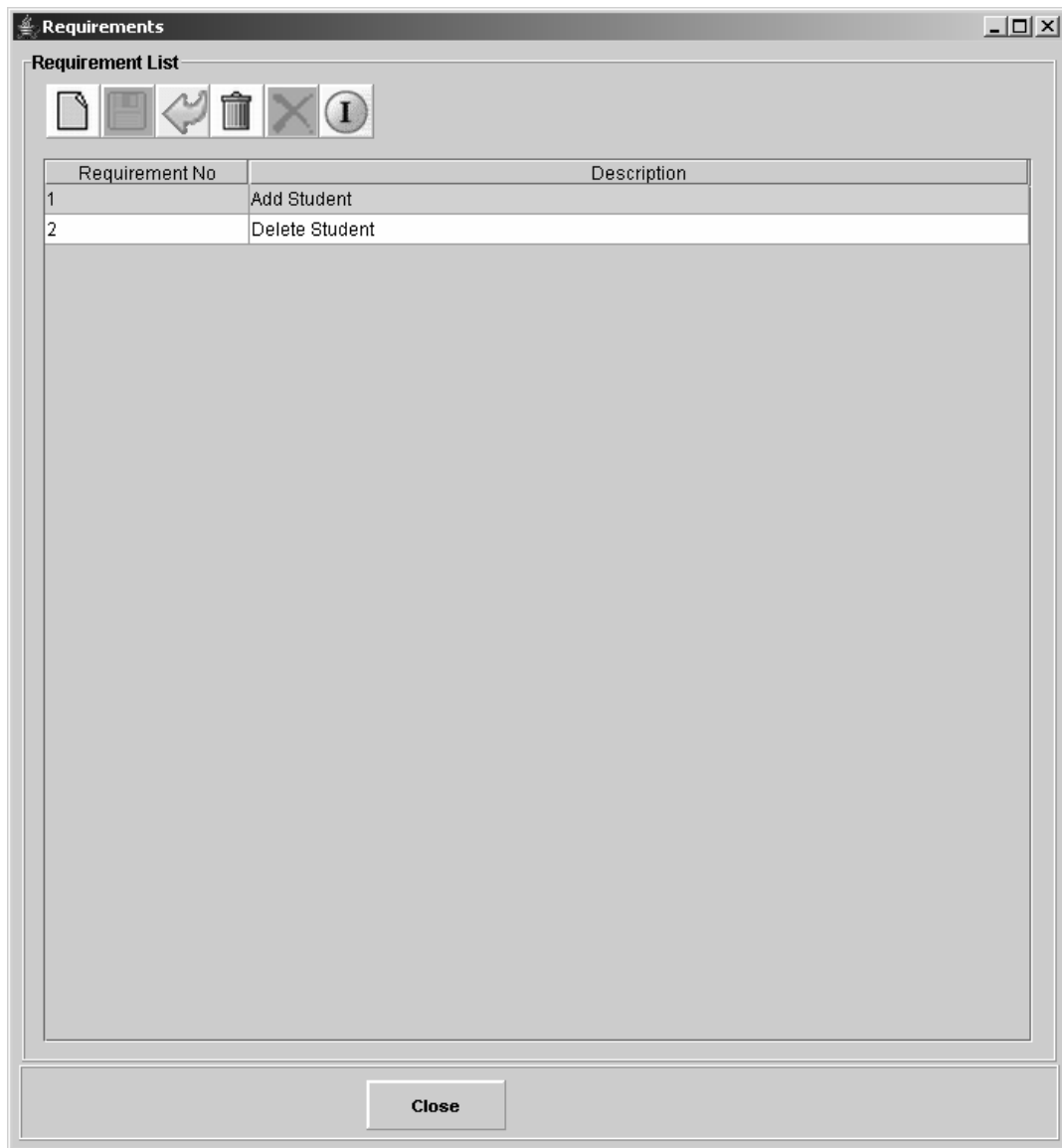


Figure 26 List of Requirements Screen

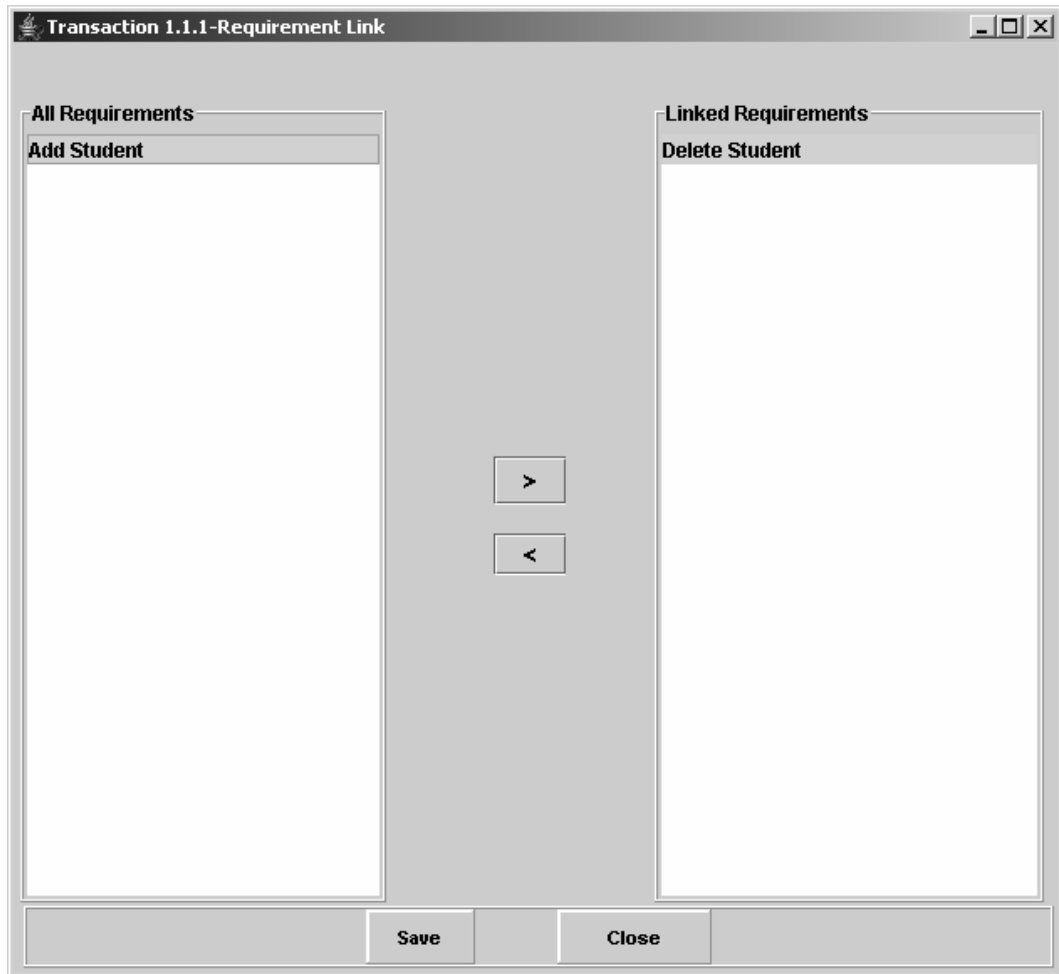


Figure 27 Transaction Link with Requirement Screen

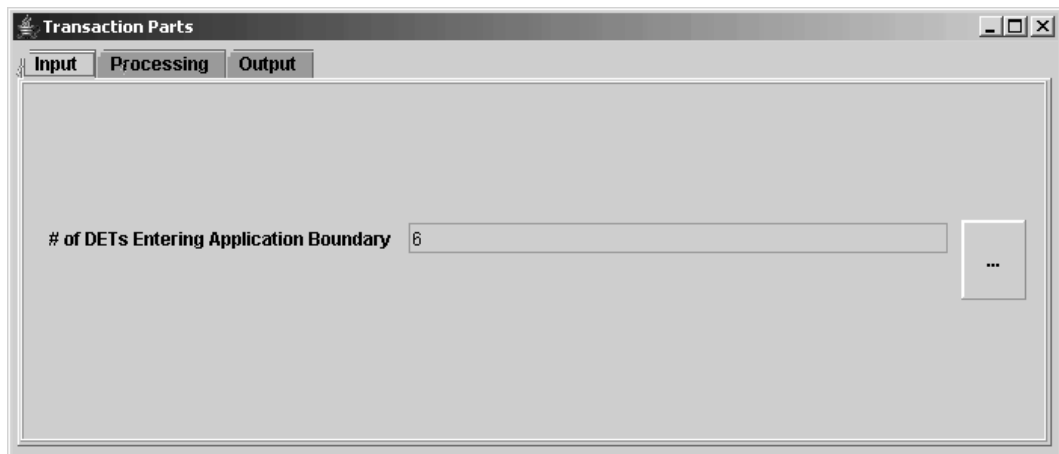


Figure 28 Transaction Input Part (only IFPUG FPA and Mk II FPA selected)

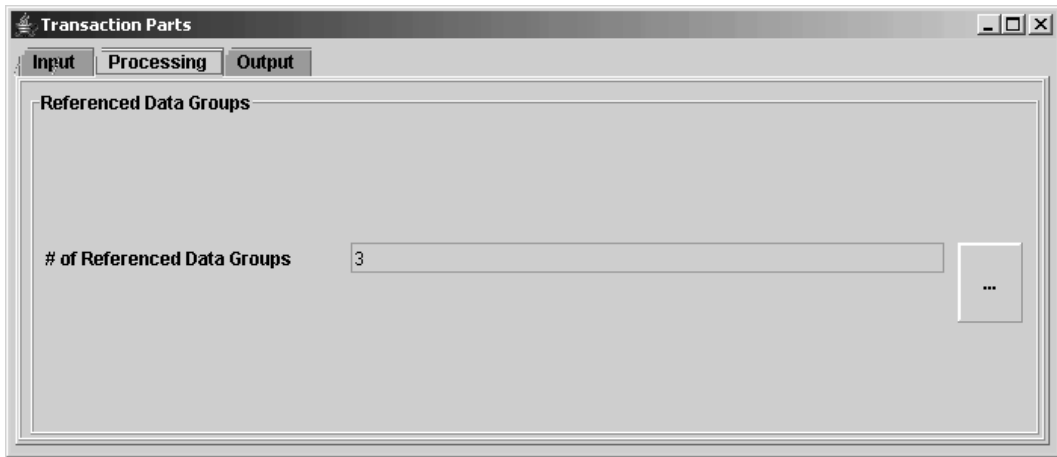


Figure 29 Transaction Processing Part (only IFPUG FPA and Mk II FPA selected)

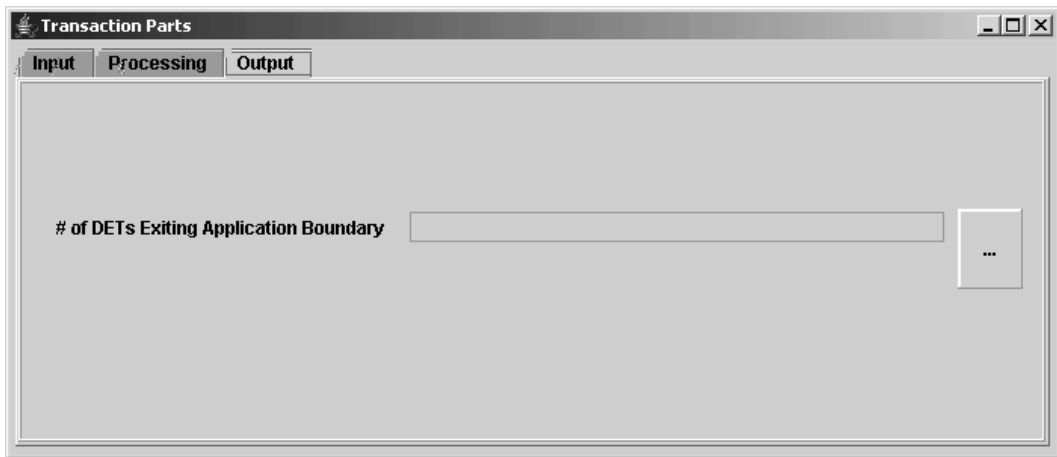


Figure 30 Transaction Output Part (only IFPUG FPA and Mk II FPA selected)

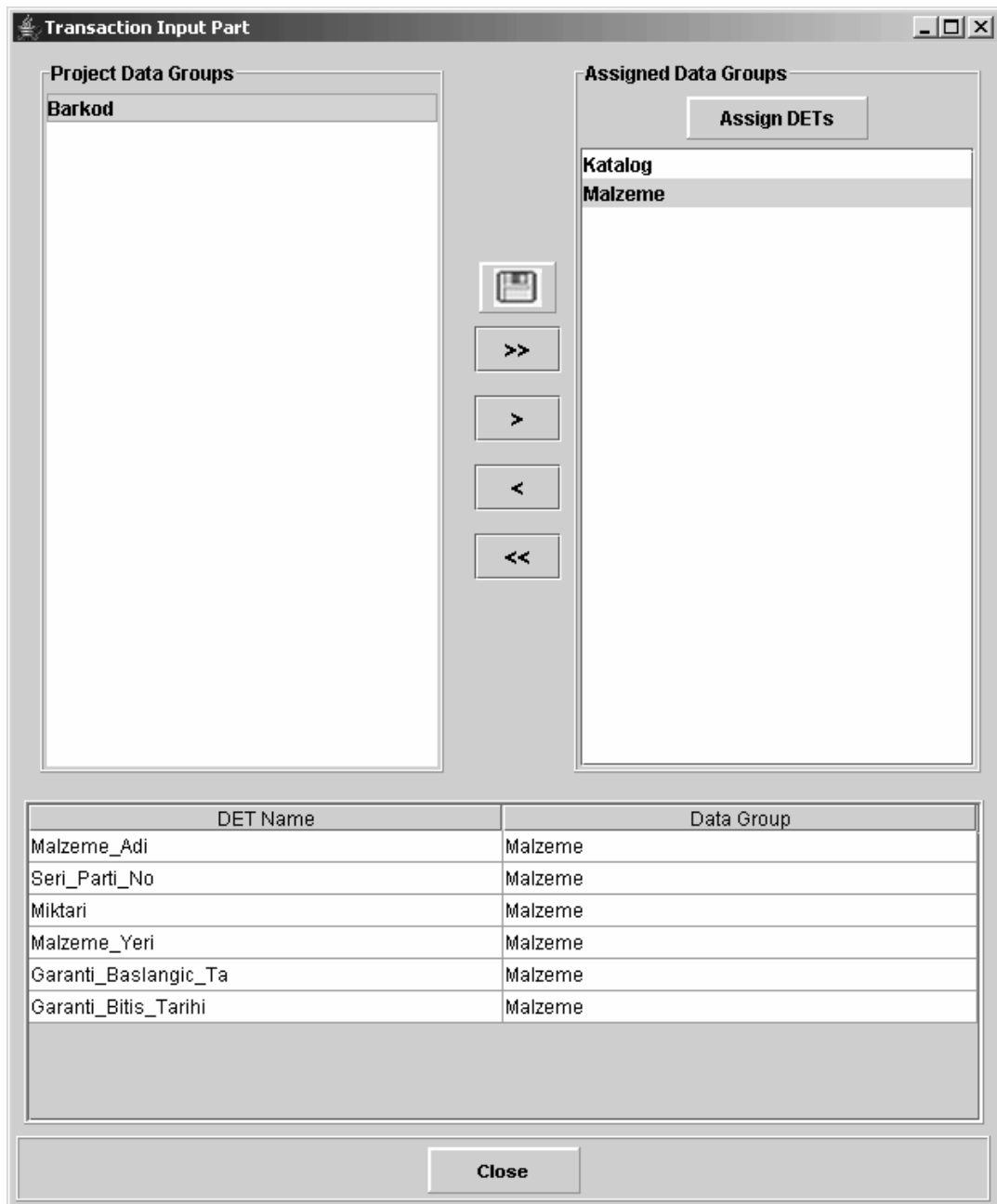


Figure 31 Assigning Data Group to the Transaction (only IFPUG FPA and Mk II FPA selected)

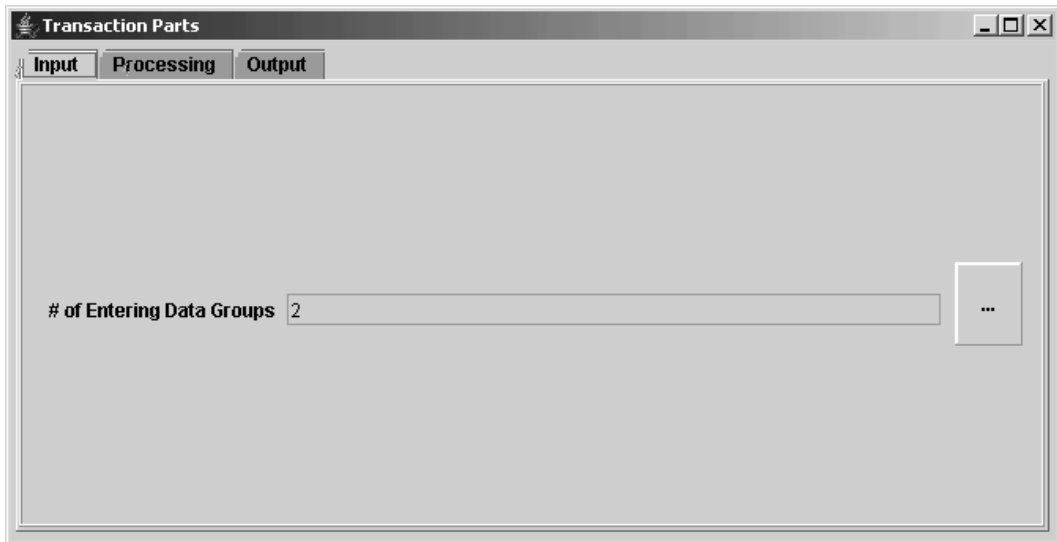


Figure 32 Transaction Input Part (only COSMIC FFP selected)

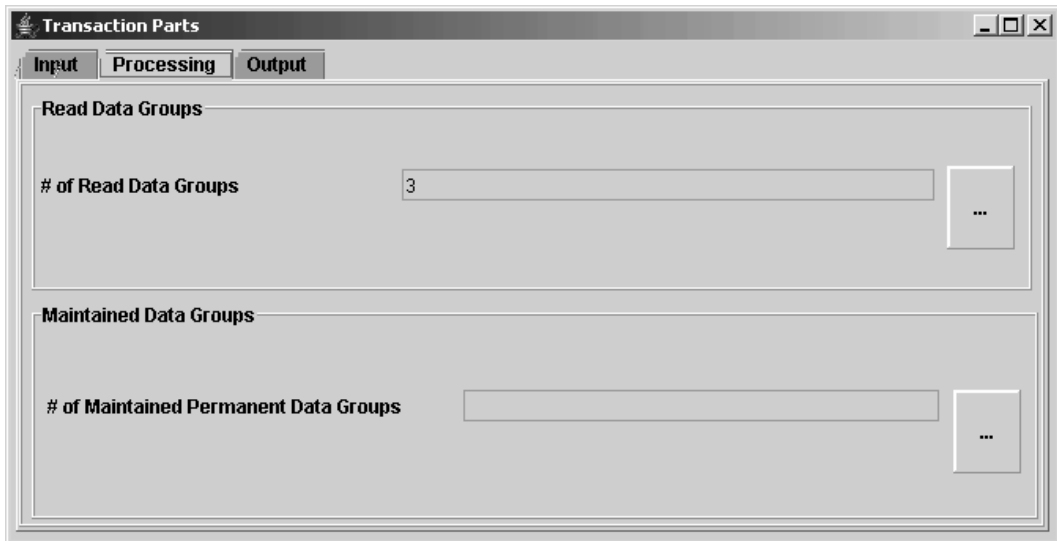


Figure 33 Transaction Processing Part (only COSMIC FFP selected)

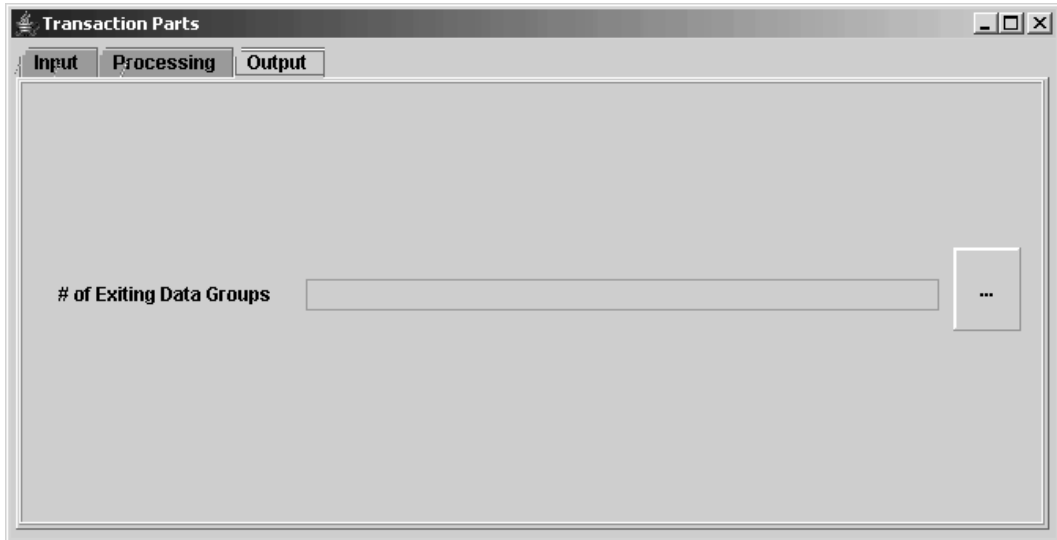


Figure 34 Transaction Output Part (only COSMIC FFP selected)



Figure 35 Assigning Data Group to the Transaction (only COSMIC FFP selected)

Transaction Parts

Input Processing Output

of DETs Entering Application Boundary 6

of Entering Data Groups 2

Figure 36 Transaction Input Part (IFPUG FPA, Mk II FPA and COSMIC FFP selected)

Transaction Parts

Input Processing Output

Read Data Groups

of Read Data Groups 3

Maintained Data Groups

of Maintained Permanent Data Groups

Figure 37 Transaction Processing Part (IFPUG FPA, Mk II FPA and COSMIC FFP selected)

Transaction Parts

Input Processing Output

of DETs Exiting Application Boundary

of Exiting Data Groups

Figure 38 Transaction Output Part (IFPUG FPA, Mk II FPA and COSMIC FFP selected)

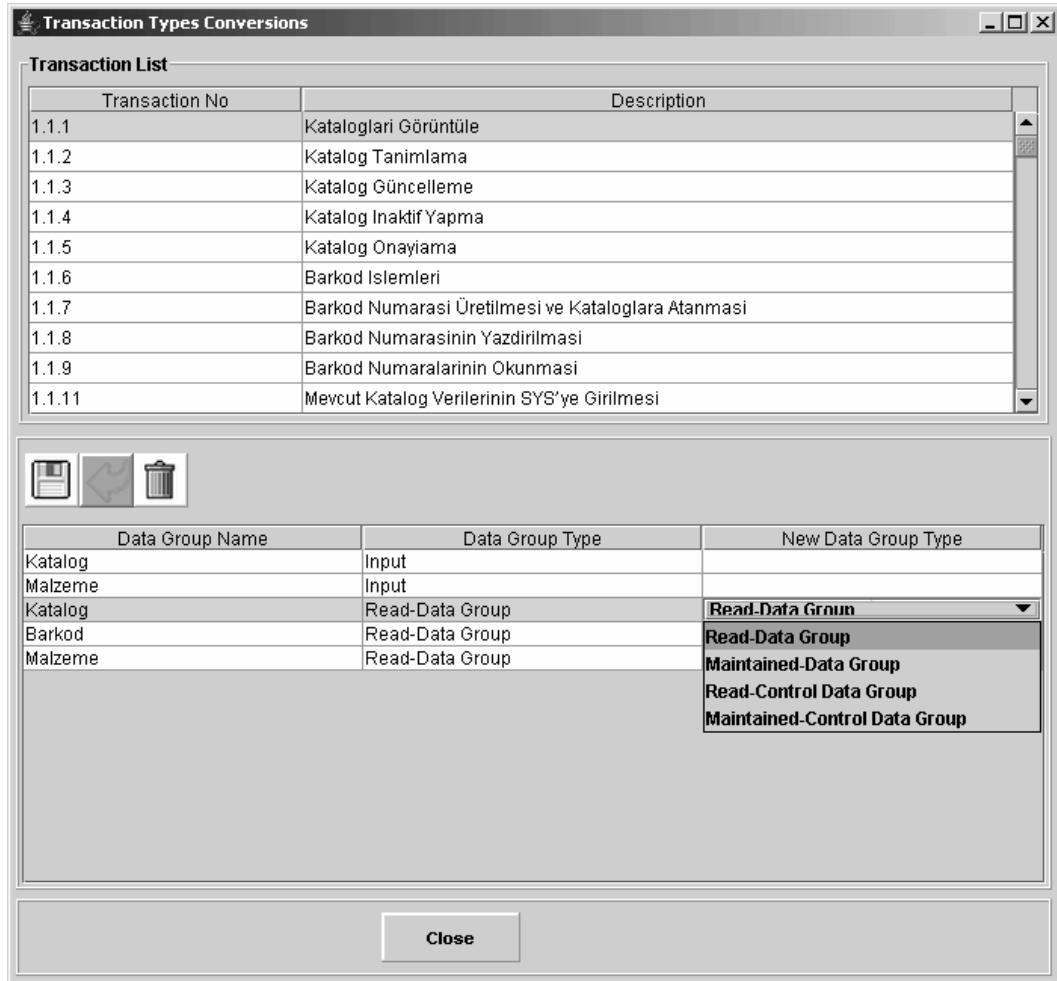


Figure 39 Transaction Data Group Conversions Screen

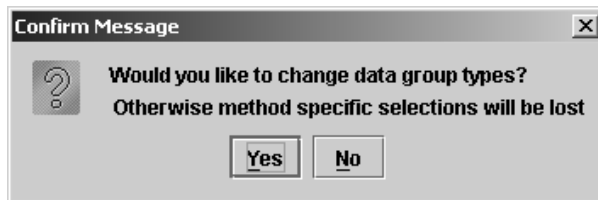


Figure 40 Confirmation Dialog for Transaction Data Group Conversion Screen

General System Characteristics	Degrees of Influence
Data Communications	0 (No influence)
Distributed Data Processing	0 (No influence)
Performance	0 (No influence)
Heavily Used Configuration	0 (No influence)
Transaction Rate	0 (No influence)
Online Data Entry	0 (No influence)
End-User Efficiency	0 (No influence)
Online Update	0 (No influence)
Complex Processing	0 (No influence)
Reusability	0 (No influence)
Installation Ease	0 (No influence)
Operational Ease	0 (No influence)
Multiple Sites	0 (No influence)
Facilitate Change	0 (No influence)
Requirements of Other Applications	0 (No influence)
Security, Privacy, Auditability	0 (No influence)
User Training Needs	0 (No influence)
Direct Use by Third Parties	0 (No influence)
Documentation	0 (No influence)

Buttons: Save, Close

Figure 41 Value Adjustment Factor Screen

Information Message

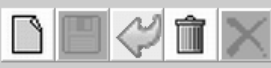
IFPUG Value Adjustment Factor : 0.72
 Mark II Technical Complexity Adjustment : 0.685

OK

Figure 42 Calculated Value Adjustment Factor Dialog

Estimation Effort

Estimation Effort History



# of Estimat...	Work Description	Start Date	End Date	Duration(h)	Effort(mh)
1	Enter Data Groups	4/1/06 8:49 PM	4/1/06 11:55 PM	3.1	3.1
1	Enter Data Groups-cont	4/2/06 9:00 PM	4/2/06 11:59 PM	3.0	3.0
1	Enter Transaction Details	4/3/06 10:15 PM	4/4/06 12:15 AM	2.0	2.0
1	Enter Transaction Details	4/7/06 2:13 PM	4/8/06 2:13 AM	12.0	12.0

Total Effort:20.1 (man hour)

Close

Figure 43 Estimation Effort Screen

APPENDIX C

REPORTS FROM EASYESTIMATE

Microsoft Excel - ifpug.xls

Next Previous Zoom Print... Setup... Margins Page Break Preview Close Help

IFPUG FPA - Summary Report 4/11/2006

Function Type	Functional Complexity	Complexity Totals	Function Type Totals
ILFs	45 Low X 7 =	315	
	0 Average X 10 =	0	
	0 High X 15 =	0	
			315
EIFs	0 Low X 5 =	0	
	0 Average X 7 =	0	
	0 High X 10 =	0	
			0
EIs	16 Low X 3 =	48	
	3 Average X 4 =	12	
	34 High X 6 =	204	
			264
EOs	17 Low X 4 =	68	
	17 Average X 5 =	85	
	90 High X 7 =	210	
			363
EQs	0 Low X 3 =	0	
	2 Average X 4 =	8	
	4 High X 6 =	24	
			32
		Total Unadjusted FP Count	974
		Value Adjustment Factor	0
		Adjusted FP Count	0

1/1

Preview: Page 1 of 1

Figure 44 IFPUG FPA Report

Microsoft Excel - MkII.xls

Next Previous Zoom Print... Setup... Margins Page Break Preview Close Help

Mark II FPA - Summary Report 5/28/2006

Function Added		
Input Types	X 0.58 =	64.96
Data Group References	X 1.66 =	318.72
Output Types	X 0.26 =	41.6
		425.28
	Total Function Point Index	425.28
	Technical Complexity Adjustment	0
	Total Adjusted Function Point Index	0

1/1

Preview: Page 1 of 1

Figure 45 Mk II FPA Report

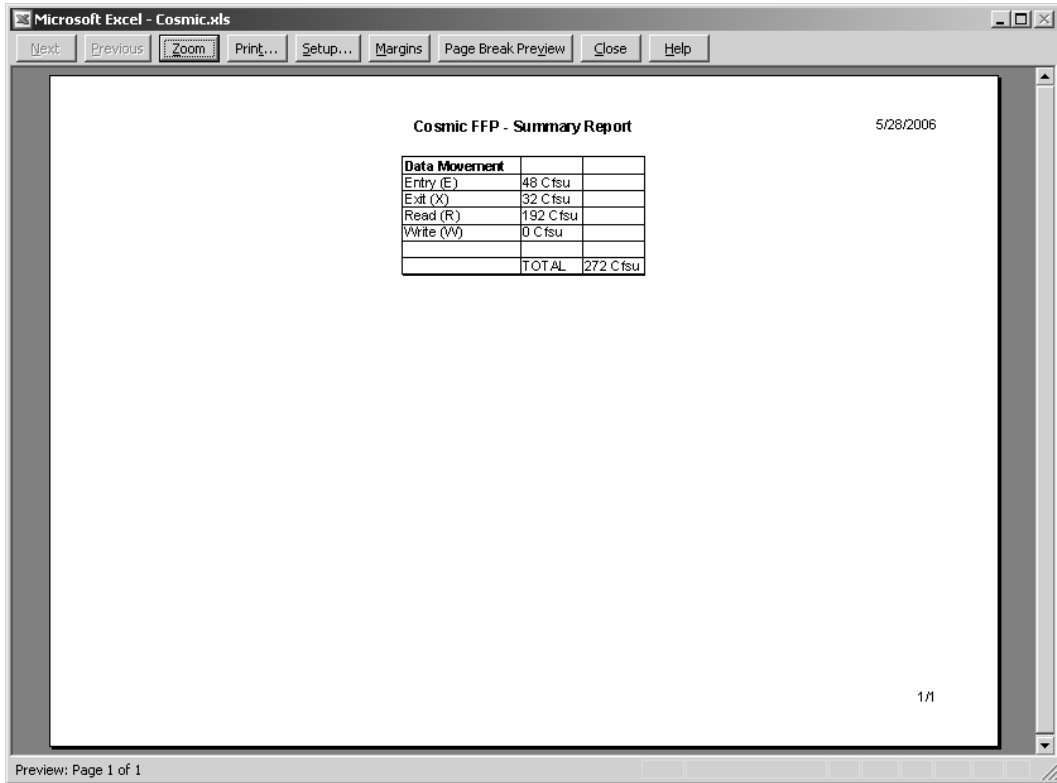


Figure 46 COSMIC FFP Report

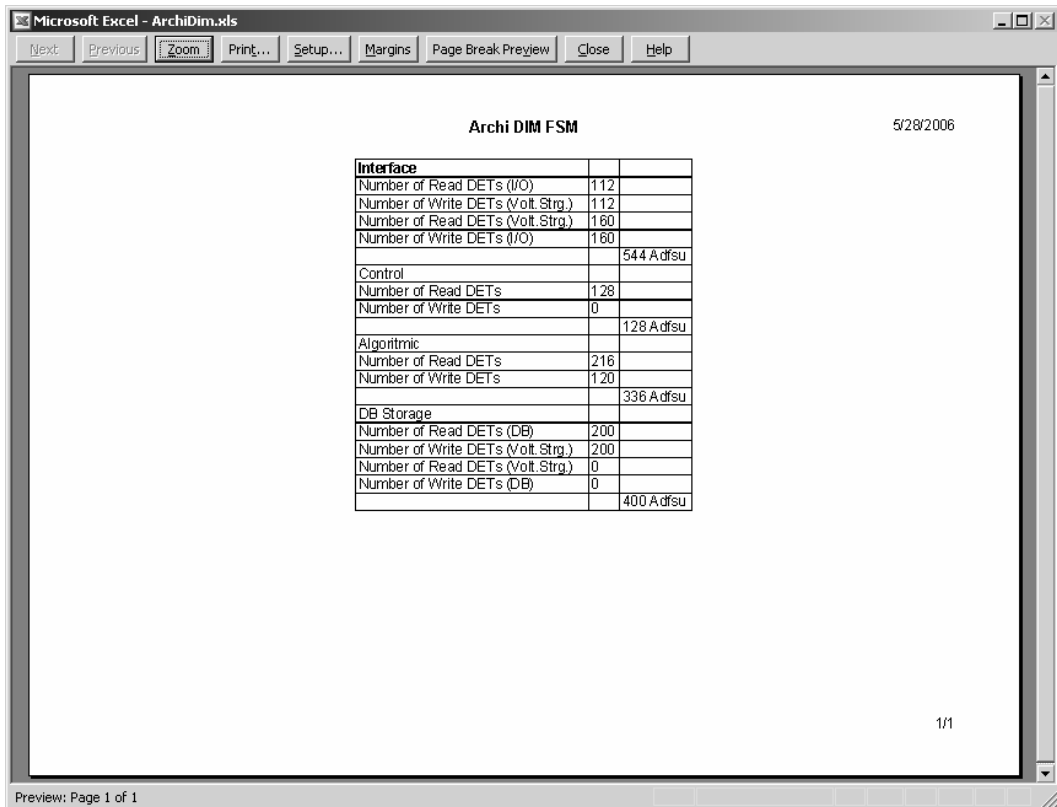


Figure 47 ARCHI DIM FSM Report

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
Transaction Name	Mk II FPA	# of Input	# of F	# of Output	COSMIC FFP	# of Exit	# of Re# of W	ARCHIDIM	# of R# of W	# of Read	# of Write	# of Re# of W	# of Write	# of Re# of W	# of Write	# of Re# of W	# of Write	# of Re# of W
164,321,1632	3	6	5	1	1	6	0	3	3	5	5	4	0	6	3	6	0	0
2064, 2170, 1632	4	6	5	2	1	6	0	4	4	5	5	4	0	7	4	6	0	0
1641, 2169, 1632	3	6	5	1	1	6	0	3	3	5	5	4	0	6	3	6	0	0
2066, 2169, 1632	4	6	5	2	1	6	0	4	4	5	5	4	0	8	5	7	0	0
1642, 2167, 1632	3	6	5	1	1	6	0	3	3	5	5	4	0	6	3	6	0	0
2066, 2167, 1632	4	6	5	2	1	6	0	4	4	5	5	4	0	7	4	6	0	0
1643, 2168, 1632	3	6	5	1	1	6	0	3	3	5	5	4	0	6	3	6	0	0
2067, 2168, 1632	4	6	5	2	1	6	0	4	4	5	5	4	0	8	5	7	0	0
2068, 2170, 1632	3	6	5	1	1	6	0	3	3	5	5	4	0	6	3	6	0	0
2069, 2170, 1632	4	6	5	2	1	6	0	4	4	5	5	4	0	7	4	6	0	0
2070, 2169, 1632	3	6	5	1	1	6	0	3	3	5	5	4	0	6	3	6	0	0
2071, 2169, 1632	4	6	5	2	1	6	0	4	4	5	5	4	0	8	5	7	0	0
2072, 2167, 1632	3	6	5	1	1	6	0	3	3	5	5	4	0	6	3	6	0	0
2073, 2167, 1632	4	6	5	2	1	6	0	4	4	5	5	4	0	7	4	6	0	0
2074, 2168, 1632	3	6	5	1	1	6	0	3	3	5	5	4	0	6	3	6	0	0
2075, 2168, 1632	4	6	5	2	1	6	0	4	4	5	5	4	0	8	5	7	0	0
2076, 2170, 1632	3	6	5	1	1	6	0	3	3	5	5	4	0	6	3	6	0	0
2077, 2170, 1632	4	6	5	2	1	6	0	4	4	5	5	4	0	7	4	6	0	0
2078, 2169, 1632	3	6	5	1	1	6	0	3	3	5	5	4	0	6	3	6	0	0
2077, 2170, 1632	4	6	5	2	1	6	0	4	4	5	5	4	0	8	5	7	0	0
2080, 2167, 1632	3	6	5	1	1	6	0	3	3	5	5	4	0	6	3	6	0	0
2081, 2167, 1632	4	6	5	2	1	6	0	4	4	5	5	4	0	7	4	6	0	0
2082, 2168, 1632	3	6	5	1	1	6	0	3	3	5	5	4	0	6	3	6	0	0
2083, 2168, 1632	4	6	5	2	1	6	0	4	4	5	5	4	0	8	5	7	0	0
2084, 2170, 1632	3	6	5	1	1	6	0	3	3	5	5	4	0	6	3	6	0	0
2085, 2170, 1632	4	6	5	2	1	6	0	4	4	5	5	4	0	7	4	6	0	0
2086, 2169, 1632	3	6	5	1	1	6	0	3	3	5	5	4	0	6	3	6	0	0

Figure 48 Transaction Detailed Report