AN ERROR PREVENTION MODEL FOR COSMIC FUNCTIONAL SIZE MEASUREMENT METHOD

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

BY

MURAT SALMANOGLU

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE
OF
MASTER OF SCIENCE
IN
THE DEPARTMENT OF INFORMATION SYSTEMS

SEPTEMBER 2012

AN ERROR PREVENTION MODEL FOR COSMIC FUNCTIONAL SIZE MEASUREMENT METHOD

Submitted by MURAT SALMANOĞLU in partial fulfillment of the requirements for the degree of **Master of Science in Information Systems, Middle East Technical University** by,

Date: 14.09.2012

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, Last name	**:**	**Murat Salmanoğlu**
Signature	**:**	__________________

ABSTRACT

AN ERROR PREVENTION MODEL FOR COSMIC FUNCTIONAL SIZE MEASUREMENT METHOD

Salmanoğlu, Murat

M.Sc., Department of Information Systems

Supervisor: Prof. Dr. Onur Demirörs

September 2012, 52 pages

Estimation and measurement of the size of software is crucial for project management activities. Functional size measurement is one of the most frequently used methods to measure size of software and COSMIC is one of the popular methods for functional size measurement. Although precise size measurement is critical, the results may differ because of the errors made in the measurement process. The erroneous measurement results cause lack of confidence for the methods as well as reliability problems for effort and cost estimations. This research proposes an error prevention model for COSMIC Functional Size Measurement method to increase the reliability of the measurements. The prevention model defines data movement patterns for different types of the functional processes and a cardinality table to prevent errors. We validated the prevention model with two different case studies and observed that it can decrease errors up to 90% in our case studies.

Keywords: COSMIC, Functional Size Measurement, Error Prevention.

ÖZ

COSMIC FONKSİYONEL BÜYÜKLÜK ÖLÇÇÜMÜ YÖNTEMİ İÇİN BİR HATA ÖNLEME MODELİ

Salmanoğlu, Murat
Yüksek Lisans, Bilişim Sistemleri Bölümü
Tez Yöneticisi: Prof. Dr. Onur Demirörs

Eylül 2012, 52 sayfa

Proje yönetim aktivitelerinde yazılım büyüklük kestirimi ve ölçümü çok önemlidir. Yazılım büyüklüğü ölçümünde en çok kullanılan yöntemlerden birisi fonksiyonel büyüklük ölçümüdür, COSMIC ise en popüler fonksiyonel büyüklük ölçüm yöntemlerindendir. Büyüklük ölçümlerinin kusursuzluğu kritik olmasına rağmen ölçüm sonuçları ölçüm sürecinde yapılan hatalardan dolayı değişiklik gösterebilir. Hatalı ölçüm sonuçları yöntemlere duyulan güveni azaltığı gibi efor ve büyüklük kestirimi ile ilgili güvenirlilik sorunları da yaratır. Bu araştırma ölçüm güvenilirliğini arttırmak için COSMIC fonksiyonel Büyüklük ölçüm yönteminde kullanılacak bir hata önleme modeli önerir. Hata önleme modeli farklı fonksiyonel süreç türleri için veri hareketleri şablonları ve bir bağıntı tablosu tanımlar. Model iki farklı vaka çalışması ile doğrulanmış ve bu çalışmalardaki hataları %90'a kadar azaltabildiği gözlemlenmiştir.

Anahtar Kelimeler: COSMIC, Fonksiyonel Büyüklük Ölçümü, Hata Önleme.

To My Parents, My Brother

And

My Beloved Wife

ACKNOWLEDGMENTS

I would like to express my sincere gratitude to my advisor Prof. Dr. Onur Demirörs for his enlightening suggestions, continuous support and guidance. His patience and kindness are most important factors that help me to encourage myself and make me believe in myself.

I also want to thank to ASELSAN Inc. for supporting my study. I am also grateful to my director İsmet Atalar and my supervisor Aybeniz Yiğit from ASELSAN for their support.

I would like to thank Barış Özkan and Mina Nabi for sharing their experience with Grails framework used in Preventive COSMIC.

Finally, I will never forget the support of my family by inspiring me through all my life and giving me the drive to tackle any task with enthusiasm and determination.

And, I would like to thank my wife for her understanding during my study. She was there whenever I need with her endless support and love.

TABLE OF CONTENTS

ABSTRACT iv
ÖZ v
DEDICATION vi
ACKNOWLEDGMENT vii
TABLE OF CONTENTS viii
LIST OF TABLES x
LIST OF FIGURES xi
LIST OF ABBREVIATIONS xii
CHAPTER
1. INTRODUCTION 1
   1.1 Problem 2
   1.2 Solution Approach 4
   1.3 Validation 5
   1.4 Road Map 6
2. RELATED RESEARCH 7
   2.1 Software Size Measurement 7
   2.2 Functional Size Measurement 8
   2.3 COSMIC Functional Size Measurement Method 9
   2.4 Reliability of FSM Methods 11
3. ERROR PREVENTION MODEL FOR COSMIC FSM 16
   3.1 Common Errors in COSMIC FSM 16
   3.2 Errors Related with Cascading Relations 18
   3.3 Errors Related with FP Types 20
   3.4 Preventive COSMIC 25
4. CASE STUDIES 34

4.1 Case Study Design 35
4.2 Exploratory Case Study Conduct 36
4.3 Validation Case Study Conduct 42
5. CONCLUSION AND FUTURE WORK 46
5.1 Conclusions 46
5.2 Future Studies 49
6. REFERENCES 50

LIST OF TABLES

Table 2.1 Common Observed Errors in COSMIC 14
Table 3.1 Classification of Common Observed Errors in COSMIC 17
Table 3.2 Cardinality Table, Simplified Version of ER Diagram 20
Table 3.3 DM Patterns of Common FP Types 21
Table 4.1 Errors Found in the Case Study 38
Table 4.2 Number of Errors For Teams 39
Table 4.3 Size of Errors For Teams 40
Table 4.4 Number of Errors in Second Case Study 43
Table 4.5 Size of Errors in Second Case Stud 44

LIST OF FIGURES

Figure 2.1 Overview of Data Movements in COSMIC 11
Figure 3.1 ER Diagram of Example Software 18
Figure 3.2 Single FUR of the Example Software 19
Figure 3.3 Representation of Preventive Model 24
Figure 3.4 COSMIC Measurement Page in CUBIT 25
Figure 3.5 Main Page of Preventive COSMIC 26
Figure 3.6 Add Cardinality 27
Figure 3.7 Cardinality View 27
Figure 3.8 Preventive COSMIC Link 27
Figure 3.9 Choose FP Type 28
Figure 3.10 Triggering Entry 28
Figure 3.11 Triggering Entry Warning 29
Figure 3.12 Repetitive Part 30
Figure 3.13 Repetitive Part Warning 31
Figure 3.14 Cascading Relation Warning 32
Figure 3.15 Error / Confirmation 32
Figure 3.16 Preventive COSMIC Main Screen With Measurement Detail 33

LIST OF ABBREVIATIONS

CFP	:	COSMIC Function Points
CMMI-DEV	:	Capability Maturity Model Integration for Development
COSMIC	:	Common Software Measurement International Consortium
DG	:	Data Group
DM	:	Data Movement
ER	:	Entity Relationship
FP	:	Functional Process
FPA	:	Function Point Analysis
FSM	:	Functional Size Measurement
FUR	:	Functional User Requirements
II	:	Informatics Institute
ISO	:	The International Organization for Standardization
METU	:	Middle East Technical University
MIS	:	Management Information System
OOI	:	Object Of Interest
SEI	:	Software Engineering Institute
SLOC	:	Source Lines of Code
SMRG	:	Software Management Research Group
SRS	:	Software Requirements Specification

CHAPTER 1

INTRODUCTION

One of the most important tasks of software project managers is to manage the schedule of the project. Schedule of the project is related with the necessary effort to finish the project on time and within budget and to estimate the effort, size of the software should be known. Measuring the size is a critical task, because the errors in the measured size may create significant discrepancies in the schedule and budget estimation of the project, which as a result may determine whether the project will fail or succeed. Thus correct size measurement has great importance for the software project managers.

There are different measures used for the size of software. Line of codes is traditionally most common measure. Although it is useful with small-scale projects, specialists do not prefer line of code for large-scale projects, as it is hard to measure at the early phases of the project and it changes according to the coding language used (Low and Jeffery, 1990). Another common measure is functional size. Functional size measurement (FSM) has an increasing popularity among software professionals and academicians. It has two main advantages over other methods; it can be used in early stages of software development and it is language independent.

Since Albrecht (1979) first introduced Functional Point Analysis (FPA) method to the software world, several methods has been proposed for functional size measurement. During this research we used COSMIC (COSMIC Method Measurement Manual,

2009) functional size measurement method, which is one of the FSM methods accepted by ISO (ISO/IEC, 2011b). In addition to its acceptance by professionals, the reason to choose COSMIC is that there is already a tool called CUBIT available in METU Informatics Institute to test the solution approach of our research. CUBIT is open for both academia and industry to use for functional size measurement and benchmarking purposes (CUBIT, 2012).

1.1. Problem

The possible errors in the measurement may result in false size measurement for the software and negatively affect project management activities, which in return may result as a failure. Although functional size measurement methods have gained popularity, software professionals have doubts about the reliability of FSM methods. The main criticism is that the measurement process is subjective and the results may change according to the measurer.

Research shows that these critics are partially true, measurements made by different measurers may produce different results (Kemerer, 1993). In their works Türetken, Özcan Top, Özkan, and Demirörs (2008b) found that interpretations and assumptions of different measurers create different measurement results. They suggested that measurement methods need to guide the measurers more closely in some concepts like entity generalization to reduce interpretations differences of measurers and improve measurement results.

Ungan, Demirörs, Özcan Top, and Özkan (2010) made a research to find errors among individual measurements and to find the causes of these errors. They found two main reasons for errors, first reason is that measurers make errors in application of measurement rules, and second reason is measurers' different interpretations and assumptions about project and method rules. In another research Ungan, Demirörs, Özcan Top, and Özkan (2009) applied an improved training program to prevent errors found in their previous work. In the improved training they highlighted common errors to the measurers and before the measurement they applied a pilot study to them. After the pilot they gave feedback to measurers about the errors made, and discussed the results with them. They also prepared the ER diagram for the main project together with the measurers to build a common understanding about the project. Results of the project measurement showed that an improved training approach may reduce measurement errors from 39% to 13%.

Özcan Top, Demirörs, Özkan (2009) conducted a similar research. They applied a pilot study to some of the measurers before measurement to underline common errors and gave feedback after the study. Measurement results showed that measurers attended to pilot study had fewer errors than other measurers.

COSMIC: Guideline for Assuring the Accuracy of Measurements (2011) lists three reasons for the errors in the measurement results. First the errors made by the measurers, second the quality of the measurement artifacts, and third the process used in the measurement activities.

The research demonstrates that there are reliability problems with FSM methods. There are suggestions to decrease the errors and they considerably increase the reliability; however they depend on methods to increase the experience of the measurer. Also these models depend on the interpretation of the measurer.

Although it is shown that errors may be prevented by guiding measurers, there isn’t any defined model to guide the measurers during measurement and prevent possible errors.

In this research, we aim to find a structured model to guide the measurers during measurement to prevent the errors. We think that by using a preventive model it is possible to guide the measurer with any experience level through measurement process and prevent common errors.

1.2. Solution Approach

It is shown that measurement result may be improved by training the measurers about the common error types. However there is still an improvement opportunity as these methods depend to the measurers. We aimed to find a solution which does not depend on experience level or attention of the measurer. There should be a solution to prevent the errors in a structured way, this solution should guide the measurer through critical points giving him minimum initiative.

Inspection of the common errors showed that common errors are clustered around some functional process (FP) types and we may prevent the errors by structuring systematic rules for the FP types. We analyzed the FP types and observed that every FP type have specific data movement (DM) patterns. The fundamental rules of the preventive method depend on these DM patterns.

In addition to the errors about the FP types, there are errors related with the cardinality relations among entities in the software, measurers tend to forget cascading processes in measurements. To prevent these types of errors, we include entity relationship (ER) diagram of the software in our preventive model, measurers should check ER table for the functional processes to determine whether a cascading process is required or not.

After building the preventive model, we also integrated it into preexisting software CUBIT and build preventive COSMIC tool (CUBIT, 2012). This tool helps the measurers through measurement and guide them according to the rules defined in preventive model.

Our solution approach aims to prevent possible errors with minimum initiative given to the measurer. Errors should be prevented by guiding the measurer through measurement process. The approach is formed by a two step prevention model, at first step user should enter functional processes according to their type and should follow the rules defined for every type. At the second step user should check the ER diagram according to the rules defined in the model to prevent errors related with cascading relations.

1.3. Validation

Our aim was to prevent errors by guiding the measurer according to the FP types. We conducted an initial exploratory case study to determine common errors that can be prevented by defining structured rules and suggest the rules. After extracting and inspecting the errors, we defined basic DM patterns for main FP types according to our observations and defined a set of rules to guide the measurer. In the case study, we reviewed the results of the measurements of the students of Information Systems Departments. Conducting the rule set we observed that the majority of the errors could be prevented if the measurers were using the new preventive rule set.

After the first case study showed that our model has potential to reduce the number of errors, we integrated the model into CUBIT, and build “Preventive COSMIC” software. Preventive COSMIC guides the measurers during measurement through predefined rule set created for each FP type.

We conducted the second case study by using Preventive COSMIC. A measurer with COSMIC measurement certificate measured a software requirement specification (SRS) document of a Management Information Systems (MIS) software project by using CUBIT before we defined preventive model. Experienced measurers analyzed the results and classified the errors. After we built our preventive model, same measurer measured the same software by using new Preventive COSMIC. Experienced measurers again analyzed the results and classified the errors after the second measurement. We compared the results of the two measurements and found out significant decrease in the number of errors.

1.4. Road Map

We performed a thorough literature review at the beginning of the research about software size measurement, functional size measurement, COSMIC, and reliability of FSM methods. Second section summarizes our findings in related research.

In third section we explain the details of proposed prevention model: how we classified the errors, what the details of the model are, and how we structured Preventive COSMIC software.

In section 4 we summarized the details and results of two case studies conducted to validate the preventive model.

Section 5 concludes the thesis with the summary of the works, and explores possible future work about error prevention in COSMIC FSM.

CHAPTER 2

RELATED RESEARCH

In this section literature review conducted for the thesis is summarized. The review is divided into four sections; software size measurement, functional size measurement, COSMIC FSM, reliability of FSM methods.

2.1. Software Size Measurement

Size measurement is an important input for software project management practices (“CMMI® for Development”, 2010). It is main input for effort and schedule estimation, which in turn determine project budget and duration.

The oldest and widely used measure for size measurement is source lines of code (SLOC). This measure is in use from the beginning of software development activities as main measure, it is easy to understand, easy to measure, and objective. However, it is language dependent and the size of software written in different coding languages cannot be compared directly. Moreover, it is not possible to measure SLOC accurately in the early phases of software development. Considering its disadvantages, SLOC lose its acceptability in software engineering discipline to new methods (Gencel, Buglione, Demirörs, and Efe, 2006).

There are various measures for software size. In Capability Maturity Model Integration for Development version 1.3 (CMMI-DEV, V1.3) Software Engineering Institute (SEI) defines a wide variety of measures such as; source line of codes, function points, number and complexity of requirements, number and complexity of interfaces, number of database tables (CMMI® for Development, Version 1.3, 2010)…

Among other measures, functional size is second commonly used measure for software size. Functional size is calculated by measuring the functionality of the software provided to the user. In the next section, we explain functional size in detail.

2.2. Functional Size Measurement

Albrecht (1979) first introduced the idea of the functional size in 1979. First model introduced by Albrecht was called function point analysis (FPA). FPA gained significant interest because its focus was to measure the functionality of the software from users' point of view without considering the software itself. Albrecht has improved the method by Albrecht and Gaffney (1983) and Albrecht (1984).

Following Albrecht, various authors suggested different method to measure the functionality of the software with different techniques in different domains. To organize different proposed techniques of software functionality, ISO started a working group in 1996 to build common principles (ISO/IEC, 2007), (ISO/IEC (2011a), ISO/IEC (2003), ISO/IEC (2002a), ISO/IEC (2004), ISO/IEC (2006)). Currently ISO approved 5 methods to become international standard, these are: IFPUG (ISO/IEC, 2009), MkII (ISO/IEC, 2002b), COSMIC (ISO/IEC, 2011b), NESMA (ISO/IEC, 2005), FISMA (ISO/IEC, 2010).

Among these methods we used COSMIC in this research. There are several reasons to work with COSMIC; first reason is that software management research group of Informatics Institute (II) in Middle East Technical University (METU) has a knowledge base about COSMIC. Secondly, there is an available tool in METU II to be used for COSMIC research (CUBIT). Lastly, COSMIC is gaining popularity among both researchers and professionals of software development.

In next section COSMIC is explained in detail.

2.3. COSMIC Functional Size Measurement Method

COSMIC is one of the FSM methods that are accepted by ISO and we worked with COSMIC in our research because of the know-how in METU II and its popularity.

COSMIC is introduced in 1999, and since then it has been improved and new versions have been released. By the time of this research COSMIC version 3.0.1 is in use (COSMIC Method Measurement Manual, 2009).

COSMIC method is applicable in two main domains: Business application software like banking or insurance management, and real time software like telephone exchange or message switching. Measurers can also use COCMIC with hybrids of these systems, however it is not applicable for algorithm rich software like expert systems or simulation software.

COSMIC measurement process has three main phases; in measurement strategy phase the measurer should define purpose and scope, then in mapping phase measurer should apply generic software model, and lastly in measurement phase the functional size of the software is obtained.

In measurement strategy phase, measurer should determine the purpose and scope of the measurement. COSMIC applies set of rules to functional user requirements (FUR) of the software. Other requirements like operational or performance requirements are not included in the scope of COSMIC. In the measurement strategy phase, also the functional users of the software should be assigned.

In mapping phase, the functional processes are derived from functional user requirements. Functional process is defined as “an elementary component of a set of Functional User Requirements comprising a unique, cohesive and independently executable set of data movements.” (COSMIC Method Measurement Manual, 2009)

Functional process starts when a functional user of the software sends a triggering entry to the software for an event and finishes when the software completes all requirements for the event.

After functional processes are derived, measurer determines Object of interests (OOI) and data groups (DG). OOI can be anything that is identified from the point of view of the FUR, about which the software is process or store data and a DG is a distinct, non-empty, non-ordered and non-redundant group of attributes related with one OOI (COSMIC Method Measurement Manual, 2009).

In measurement phase, measurer identifies data movements in the FUR and applies measurement function. Data movements represent data that cross the boundary between the functional user and the software, or the boundary between the software and persistent storage (Figure 2.1). There are four types of data movements:

- Entry (E): Moves a data group from functional user to the functional process.
- Exit (X): Moves a data group from the functional process to the functional user.
- Read (R): Moves a data group from the persistent storage to the functional process.
- Write (W): Moves a data group from the functional process to the persistent storage.

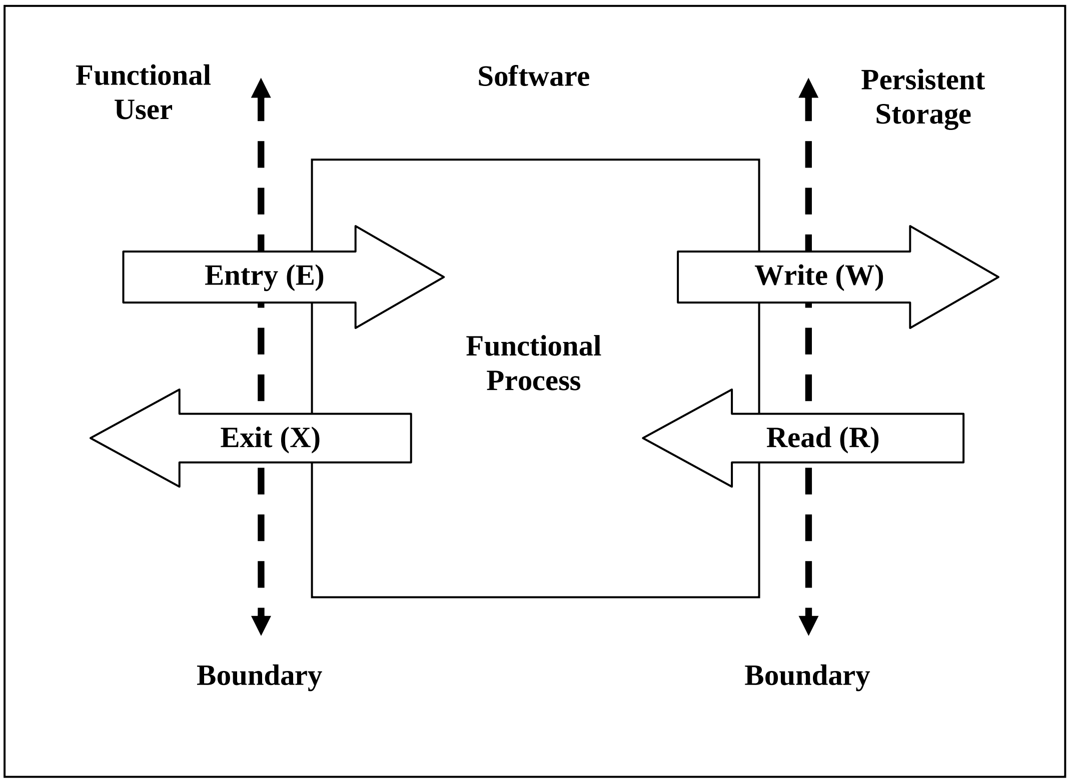


Figure 2.1 Overview of Data Movements in COSMIC

After determining the data movements, measurer should aggregate measurement results by applying measurement function (Equation 2.1). Any functional process should include at least one triggering entry (E) and at least one exit (X) or write (E) movement, as a result of that rule any FP should have at least two DM.

$$\text{Size(functional process}_i\text{)}=\sum\text{Size}(E_i)+\sum\text{Size}(X_i)+\sum\text{Size}(R_i)+\sum\text{Size}(W_i) \quad \text{(Equation 2.1)}$$

2.4. Reliability of FSM Methods

Early works about the reliability of the FSM were mainly about the inconsistency between measures of different specialists. In one of their research about the reliability of functional size measurement, Low and Jeffery (1990) stated that function point is a more consistent measure than SLOC and should be preferred over SLOC. They state that although the results of measurements within an organization may change, these differences may be results of analyst’s interpretations and judgment. They also found heterogeneity in the measurements of different companies. Inconsistencies among different companies doe not create a thread to the validity of function points measurement as long as measurements within an organization are consistent.

Kemerer (1993) stated that although there are doubts about the inter-rater reliability, “both the inter-rater and inter-method reliability of functional processes are high” he also concluded in his early work in 1992 that even though functional size measurement is rightful accepted by the majority of the industry, by improving some factors the reliability of the FSM can be increased (Kemerer and Porter, 1992).

In 2004 Abrahao, Poels, and Pastor (2004) compared IFPUG FPA, which was most common FSM method for that time, with OOmFP, which redefines IFPUG counting rules for object oriented systems, for their reproducibility and accuracy. They found that OOmFP is significantly more consistent and accurate than IFPUG FPA (in 5% significance level). This research shows that by modifying FSM methods for the needs of the measurer we may get more accurate and consistent results.

Low and Jeffery (1990) also compared measures of experienced and inexperienced measurers and found out that experienced measurers have results that are more consistent. They also conclude that not only measurement experience but also software development experience helps measurers to achieve consistent results. Their conclusion helps us to understand that by increasing the experience level of the measurer we can increase the reliability of the measurement. However; our purpose is to find a method that works with measurers in all experience levels.

Different measurers may interpret the rules of measurements differently, which in result may cause considerable differences in measurement results. Türetken, Demirörs, Özcan Top, and Özkan (2008a) studied entity generalization concept in two FSM methods; IFPUG and COSMIC. They observed in a multiple-case study that assumptions and interpretations of the measurers differ not only in different methods but also in different measurers of same method, which result in significantly different measurement results. Interpretations effected measurement up to 67% in IFPUG and up to 80% in COSMIC. They suggest that to overcome interpretations errors in FSM, rules for entity generalization should be improved in method manuals. In another research, Türetken, Özcan Top, Özkan and Demirörs (2008b) tested MkII with IFPUG and COSMIC and found similar results. In MkII, interpretations in measurements affected the results up to 90%. These studies show that some rules are not clearly described in measurement manuals, and they may have significant effects on the measurement results. The rules should be clarified for the users, or as in our prevention method, measurers should be guided through these rules to overcome interpretation errors.

Ungan, Demirörs, Özcan Top, and Özkan (2010) divided the reasons of the errors in COSMIC measurement into two groups; first group of errors are results of wrongly applied rules of measurement. Second group includes errors that are results of assumptions and interpretations of the measurer, even though the rules are applied correctly. They suggest to include guidelines to the measurement manual to inform the measurers for most common error types. These finding are in line with the classification of the COSMIC: Guideline for Assuring the Accuracy of Measurements (2011). The guideline divide the errors in three groups which are; the errors made by the measurers, the quality of the measurement artifacts and the process used in the measurement activities.

COSMIC: Guideline for Assuring the Accuracy of Measurements (2011) is official guideline of COSMIC that lists factors affecting the quality of a FSM to increase accuracy of a measurement. Guideline divides quality control efforts into two; error-prevention and defect-detection efforts. Moreover; error-prevention efforts have three main factors: quality of the measurer, quality of software artifacts, and quality of measurement process.

Although COSMIC: Guideline for Assuring the Accuracy of Measurements (2011) gives main factors for increasing the accuracy of the measurement, it does not specify a structured method to increase reliability. For the quality of the measurer guideline suggests to increase level of experience and for the quality of measurement process gives hints for organization to build their measurement processes. Our purpose is to prepare a structured model, which guides the measurer independent from their experience levels through critical points.

A study observed that even in very basic concepts measurers may make mistakes. They listed these mistakes as in Table 2.1 and suggested to emphasize these mistakes in COSMIC trainings and measurement manual with examples (Özcan Top et al., 2009). The works of Ungan et al. (2009) and Özcan Top et al. (2009) about the reliability of the COSMIC FSM results are also the base for this research.

Table 2.1 Common Observed Errors in COSMIC

ID	Error Type
1	List Before Update is ignored
2	Retrieve Before Update is ignored
3	List is defined as a part of update FP
4	Retrieve is defined as a part of update
5	Sub-Type concept is not considered
6	Cascading Delete is not performed
7	Transient data group concept is not ignored

Ungan et al. (2010) in another research applied the methods suggested in previous work, and successfully decreased some error types by modifying the training program to emphasize advanced rules of COSMIC. They also increased inter-measurer reliability by enabling them better understand software requirements.

Yılmaz, Ungan, and Demirörs (2011) inspected the effects of the software requirement specifications document on the software size measurement and build a guideline to help the measurers to prepare the measurement process.

The related research shows that it is quite possible to make errors during a measurement by using COSMIC FSM. Although there are methods to reduce the number of errors, they depend on increasing the measurers experience level and awareness about possible errors. In our knowledge there is not any comprehensive research to provide a prevention model that is applicable during the measurement process giving minimum initiative to the measurer. The aim of this research is to define a preventive method to reduce the errors made by the measurers using COSMIC FSM method during the measurement process with structured rules.

In the next chapter preventive model and preventive COSMIC software are explained in detail.

CHAPTER 3

ERROR PREVENTION MODEL FOR COSMIC FSM

This chapter explains the prevention model for COSMIC FSM method. In the first section the common errors in COSMIC FSM is classified, in second and third sections errors related with cascading relations and errors related with FP types area explained and suggested rules to prevent the errors are given, in the last section Preventive COSMIC software and its capabilities are provided.

3.1. Common Errors in COSMIC FSM

The purpose of this research is to define a model to reduce the errors during measurement in COSMIC FSM method. To define a model, we collected common errors made in the measurements. Özcan Top et al. (2009) and Ungan et al. (2009) listed common errors in their works. We combined these errors as listed in Table 3.1. As seen in the table, we put errors into two groups; first group of errors are related with the measurement process, whereas second group of errors are results of misunderstanding the artifacts used in measurement process. The reasons of the errors in the second group are mainly the quality of the measurement artifacts and how the measurers interpret these artifacts. These errors may be prevented by training the users and increasing their experience level. However; using a preventive process during measurement may decrease the number of errors in the first group, and in this research we aimed to prevent the errors in the first group.

Table 3.1 Classification of Common Observed Errors in COSMIC

Errors Related with Measurement Process	Different error messages are considered as separate exits
	Missing Exits for Error/Confirmation
	Redundant Exits for Error/Confirmation
	List before update/delete is ignored
	Retrieve before update/delete is ignored
	Retrieve is defined as a part of another FP
	Measurement of other types of operations as a separate FP, such as close, save
	Attributes are considered data groups
	Missing Triggering Entry
	Cascading Delete is ignored
	Redundant FPs/DMs for conditional cases
	No "exit" for query results
	List & Retrieve Combined
	Retrieve and Update Combined
	Query & Detail Listing Combined
	Assumed a retrieve FP before delete/update FP
	Assumed read before write
Errors Related with Requirement Understanding	Used only Sub-Types
	Used only parent types
	Exits for populating dropdown form boxes are ignored
	Transient data group concept is ignored
	Multi pages are considered as separate FPs
	Parameter tables are considered as OOIs

Some of the errors in the first group populated around some FP types, like "Redundant Exits for Error/Confirmation", "Missing Exits for Error/Confirmation", "Assumed a retrieve FP before delete/update FP" and some of the errors are related with cascading processes, like "Cascading Delete is ignored", "Read"s for cascading deletes are missing". We approach these two different groups of error with a two step model: At the first step user should provide the ER diagram of the software, which will be used to prevent cascading process related errors. At the second step, users should enter FP's according to the FP types defined in the model checking the data movements according to predefined DM patterns of the FP types.

3.2. Errors Related with Cascading Relations

While measuring software, measurers often forget to consider cascading processes. To explain the concept of cascading process we can use the example ER diagram in Figure 3.1. In the example software there are two entities: "company" and "employee", which are in a relation called "works in". As one company may have more than one employee, the company entity has a cardinality of "N" and as one employee may only work in one company, employee entity has a cardinality of "1". This simple software has one FUR that is given in Figure 3.2.

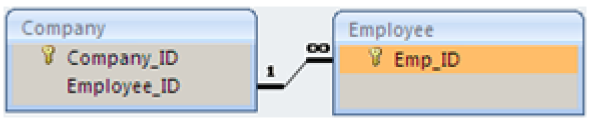


Figure 3.1 ER Diagram of Example Software

FUR-1

- Add Company
    - User enters a name for the company
    - Software saves the company
- Add Employee
    - User enters a name for the employee and choose the company that employee works for
    - Software saves the company
- Delete Employee
    - User choose the employee to delete
    - Software deletes the employee
- Delete Company
    - User choose the company to delete
    - Software deletes the company

Figure 3.2 Single FUR of the Example Software

In this simple software, delete company FP should include a cascading relation; as the employees may only work in one company, user should also delete users that work in the deleted company. To prevent measurers to make cascading process related errors, an ER diagram is expected from the user before she begins measurement. In the ER diagram entities represent OOI’s of the software. With the help of ER diagram, the measurer can follow the relations between OOI’s, and can decide whether she should include other movements for other OOI’s that belongs to the same relation. To control the possibility of cascading relations measurer should basically check the cardinalities, if one entity has a “1” cardinality in a relation, than the other entity in this relation may need a cascading process.

While measuring manually, measurers may use the E-R Diagram in any format they wish, however; in this research, we defined a cardinality table, this is a simplified version of ER diagram, to easily integrate into preventive COSMIC, which will be given in Table 3.2 and will be explained in section 3.4 in detail. For the example software, measurer will see in cardinality table (Table 3.2) that company has “1” cardinality in “Works In” relation with employee object, in this point measurer should decide according to the requirements whether to delete the employee OOI or not.

Table 3.2 Cardinality Table, Simplified Version of ER Diagram

Entity 1	**Cardinality 1**	**Entity 2**	**Cardinality 2**	**Relation**
Employee	N	Company	1	Works In
Company	1	Projects	N	Owns
Projects	1	Users	N	Employs

3.3. Errors Related with FP Types

For FP type related errors we extracted DM patterns for common FP types. We think that during the measurement these errors may be prevented by guiding the measurer with these DM patterns.

The main FP types a typical COSMIC measurer uses are: List, Retrieve, Add, Update, and Delete. Typical DM patterns for these common FP types are given in Table 3.3.

Table 3.3 DM Patterns of Common FP Types

FP Type	Data Movements		
	Triggering Entry	Repetitive Part	Error / Confirmation
List	E	R-X	X*
Retrieve	E	R-X	X*
Add	E*	E-W	X*
Update	E*	E-W	X*
Delete	E	W	X*
Other	E	…	X*

*Measurer may omit this movement

As seen in the Table 3.3 data movements of a FP are divided into three parts:

- First part is the triggering entry; every FP should include one triggering entry movement.
    - This part expected to remind the user that there should be a triggering entry, and the FP should start with an Enter DM.
    - For Add and Update FP's, triggering entry is optional, as they already have an entry movement in the repetitive part user may omit triggering entry part.
- Second section is the Repetitive part, where the main DM's of the related FP type are. Measurer may repeat repetitive part as much as necessary for cascading processes.
    - List and Retrieve FP's have "Read-Exit" movements.
    - Add/Update FP's have "Enter-Write" movements.
    - Delete FP's have "Write" movement.
    - OOI's expected to be same per repetitive part.

    - This part should be repeated as many times as necessary.
    - While filling this part, measurer expected to use the E-R diagram, which should be prepared before the measurement process.
- Last part is for Error and Confirmation (E/C) messages in the FP
    - If there is E/C message in the FUR, there should be only one entry for both positive and negative messages
    - This part will remind this to the user that she supposed to enter only one movement for the E/C message, if there is any.
    - If an E/C message is not defined user may omit this part.

We defined one last FP type, “other”, for uncommon FP’s in the software. If a FP does not fit into the defined types, user should use the “other” FP type, where only triggering entry and E/C parts are guided. User is free to for the repetitive part.

Figure 3.3 summarizes the preventive model. Model has four main controls: triggering entry, repetitive movements, cardinality relations, and error/confirmation movement. Measurer starts the measurement according to the FP type. After users choose the type of the FP, she should enter a triggering entry, then user should enter data for repetitive part according to the defined DM patterns and then check the ER diagram for cardinality. In the last control user should enter error / confirmation movement if necessary.

To illustrate the process we can trace a hypothetical measurement. The measurer wants to enter a “List” FP. After she chooses “List” as the FP type, she should first enter a triggering entry with an “Entry” DM. Then she should enter one “Read” and one “Exit” DM’s with same OOI’s. In this point, the ER diagram should be controlled to check the relations of the entered OOI’s. If there is a relation with the same OOI with cardinality of “1”, then the measurer may enter another repetitive part with same rules. If there is not any relation or the cardinality is “N”, the measurer should enter error/confirmation movement with an “E” DM at last if required.

To structure the model, we conducted an exploratory case study. We inspect the common errors in the case study and use them to build expected DM patterns for FP types. The case study showed that this process helps to reduce the errors if followed by the measurers in a COSMIC measurement. Details of the case study are explained in chapter 4 in detail.

The model integrated into CUBIT to build “Preventive COSMIC”. In the next section the details of Preventive COSMIC are given.

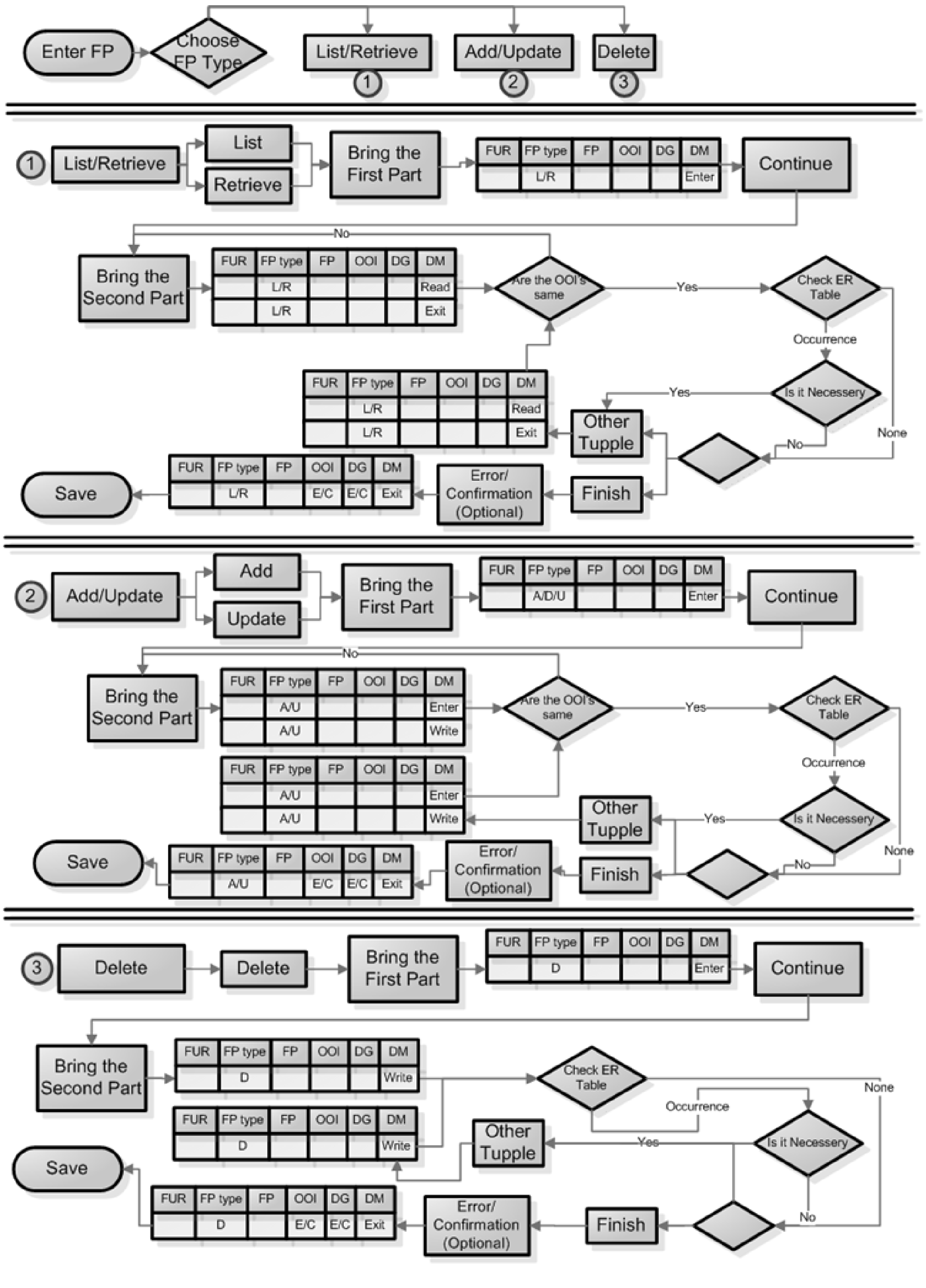


Figure 3.3 Representation of Preventive Model

3.4. Preventive COSMIC

We conducted an exploratory case study, which explained in chapter 4, to build and test the error prevention model and found out that the model may help to prevent the errors in COSMIC measurement. After having evidence for the model, to automatize the use we integrated it into CUBIT and build “Preventive COSMIC” (Figure 3.4). Preventive COSMIC has the capability of guiding the users through each rule in the preventive model. We also used the software in a case study to validate the model. The details of the case study are explained in chapter 4.

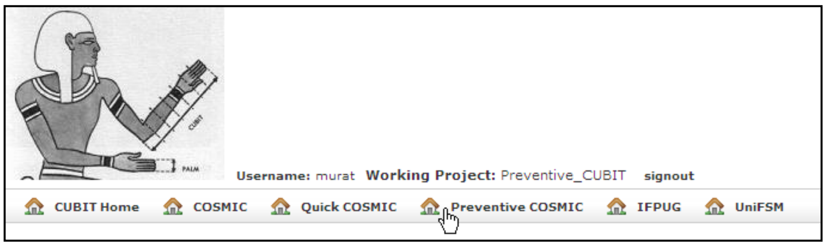


Figure 3.4 COSMIC Measurement Page in CUBIT

We developed preventive COSMIC as an expansion of CUBIT software, which is coded by Software Management Research Group (SMRG) of METU II and is open to use for both academic and professional users. CUBIT software has capabilities such as; COSMIC Measurement, IFPUG Measurement, and Benchmarking.

Preventive COSMIC is available in CUBIT, where measurers can use it to measure their software according to the rules defined in Error Prevention Model. (Figure 3.4) Software guides the measurers according to the FP type they are enter and also controls the cardinality information.

One of the main advantages of Preventive COSMIC is that it checks the cardinality table entered from the ER diagram while the measurer enters the measurement and warns the measurer if there is a possible cascading process. Cardinality table is a simplified version of ER diagram and should be filled by the measurer before the measurement process. Measurer should enter the name of the entities, their cardinality, the name of the relation, and choose the related project from the drop down menu. Measurer chooses the cardinalities of the entities from a drop down menu, there are two choices; “N” and “1”. As seen in Table 3.2, in a single “Works in” relation there could be more than one “Employee” but only one “Company” entity. To reflect this situation, cardinality of the company is “1” and cardinality of Employee is “N”.

Whenever the measurer saves a relation in cardinality table, software saves the entities as OOI for the chosen project and will later lists in drop-down menus while user enters the data movements.

After going into Preventive COSMIC, measurers see the form to start measuring the software and a link, “Edit Cardinality Information”, to enter the cardinality table for ER diagram. (Figure 3.5)

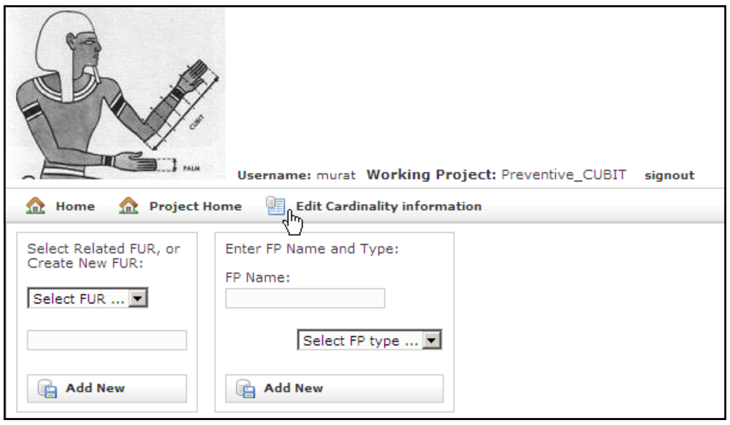
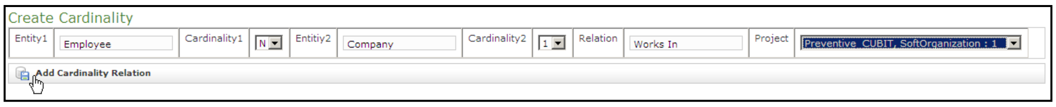


Figure 3.5 Main Page of Preventive COSMIC

Measurer should fill the cardinality table before beginning the measurement to make it possible for the software to control cascading processes. In cardinality page, user should enter entities, relations, and cardinality information (Figure 3.6) and then click “Add Cardinality Relation” button to save the relation. Software will save the relation and list in the page (Figure 3.7). If the measurer wants to edit or delete a relation, she can do it in this page. After the measurer completed all the relations, she can proceed to the measurement page by clicking Preventive COSMIC link (Figure 3.8).

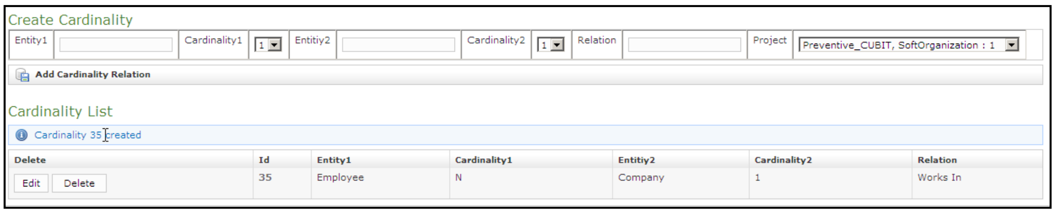


Create Cardinality

Entity1	Employee	Cardinality1	N	Entitiy2	Company	Cardinality2	1	Relation	Works In

Project: Preventive_CUBIT, SoftOrganization : 1

Figure 3.6 Add Cardinality



Create Cardinality

Entity1		Cardinality1	1	Entitiy2		Cardinality2	1	Relation	

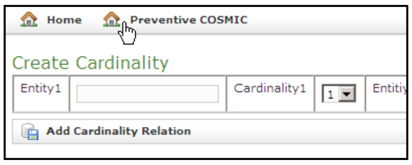
Project: Preventive_CUBIT, SoftOrganization : 1

Cardinality List

Cardinality 35 created

Delete	Id	Entity1	Cardinality1	Entitiy2	Cardinality2	Relation
Edit Delete	35	Employee	N	Company	1	Works In

Figure 3.7 Cardinality View



Create Cardinality

Entity1		Cardinality1	1	Entitiy

Figure 3.8 Preventive COSMIC Link

In measurement page, measurer starts the process by choosing FUR, FP and FP type (Figure 3.9). Software will bring first DM screen for the triggering entry (Figure 3.10). To continue the measurement, user should enter a triggering entry with an Enter DM in this screen. For Add and Update functional processes user may skip this part as they have an Enter DM in their repetitive parts. If user makes an error, for example choose a DM other than Enter, software will warn her and request to correct the error (Figure 3.11).

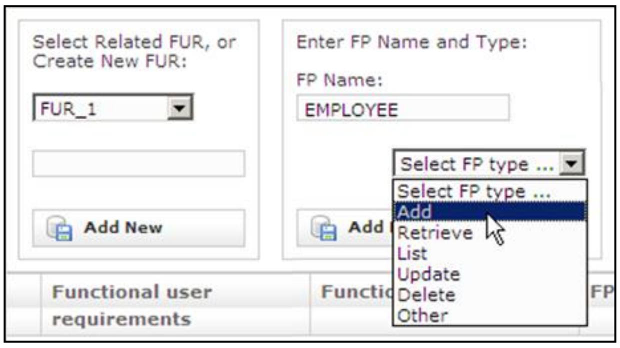
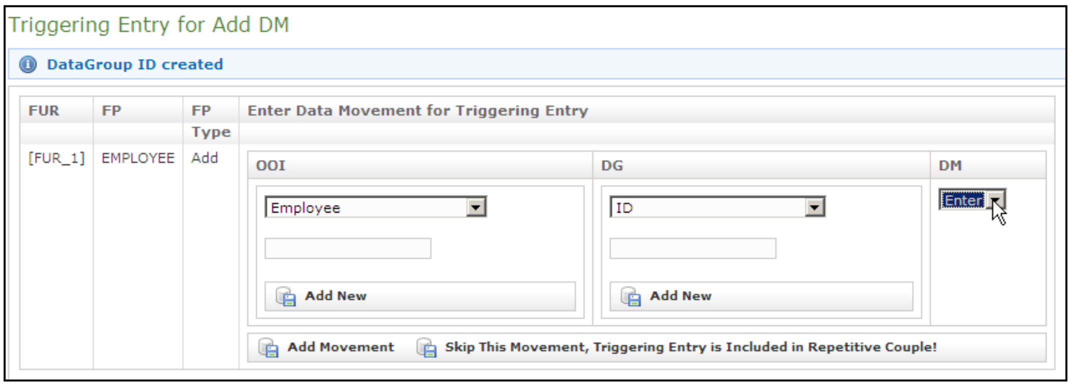


Figure 3.9 Choose FP Type

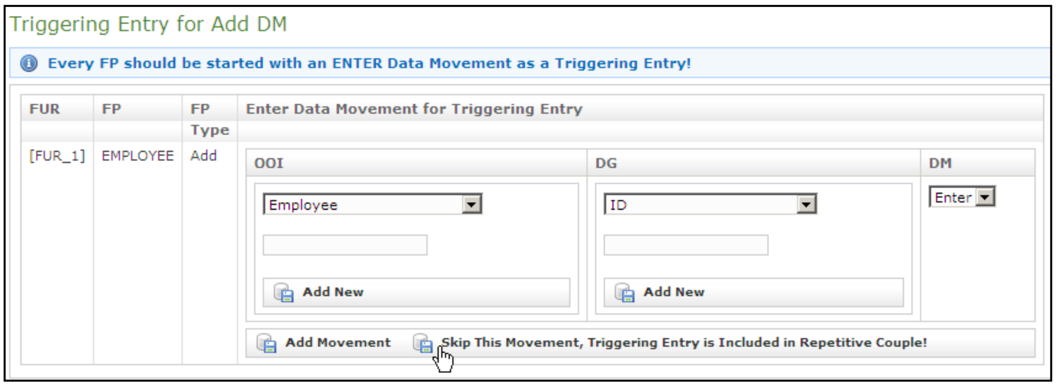

Triggering Entry for Add DM

DataGroup ID created

FUR	FP	FP Type	Enter Data Movement for Triggering Entry
[FUR_1]	EMPLOYEE	Add	OOI: Employee; DG: ID; DM: Enter

Add New
Add New
Add Movement
Skip This Movement, Triggering Entry is Included in Repetitive Couple!

Figure 3.10 Triggering Entry


Triggering Entry for Add DM

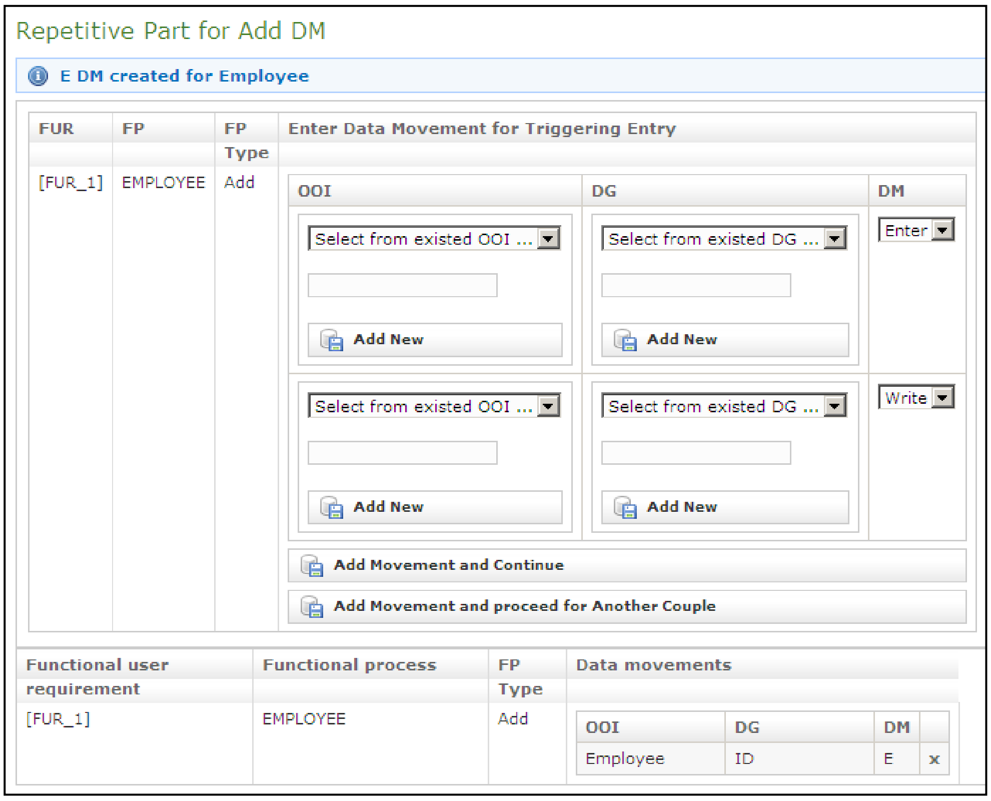
Every FP should be started with an ENTER Data Movement as a Triggering Entry!

FUR	FP	FP Type	Enter Data Movement for Triggering Entry
[FUR_1]	EMPLOYEE	Add	OOI: Employee; DG: ID; DM: Enter

Add New
Add New
Add Movement
Skip This Movement, Triggering Entry is Included in Repetitive Couple!

Figure 3.11 Triggering Entry Warning

When the triggering entry is completed correctly, user will be directed to the repetitive part (Figure 3.12), where user should enter two DM's (in Delete FP there is one DM). The OOI's and DG's should be same, and DM's should be correct, if the measurer makes an error, software will warn her and request to correct the error (Figure 3.13). If the measurer knows that there is a cascading relation, she can just add another part, if she wants to continue she can just click continue button.


Repetitive Part for Add DM

E DM created for Employee

FUR	FP	FP Type	Enter Data Movement for Triggering Entry
[FUR_1]	EMPLOYEE	Add	

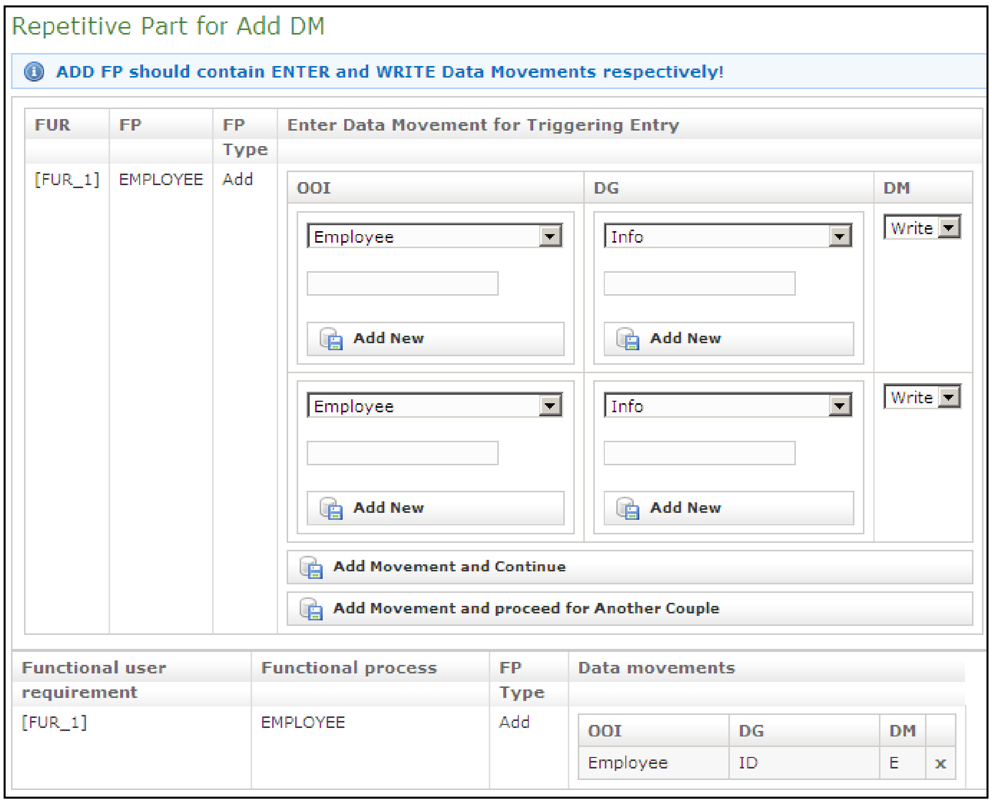
OOI	DG	DM
Select from existed OOI ... / Add New	Select from existed DG ... / Add New	Enter
Select from existed OOI ... / Add New	Select from existed DG ... / Add New	Write

Add Movement and Continue
Add Movement and proceed for Another Couple

Functional user requirement	Functional process	FP Type	Data movements
[FUR_1]	EMPLOYEE	Add	

OOI	DG	DM	
Employee	ID	E	x

Figure 3.12 Repetitive Part


Repetitive Part for Add DM

ADD FP should contain ENTER and WRITE Data Movements respectively!

FUR	FP	FP Type	Enter Data Movement for Triggering Entry
[FUR_1]	EMPLOYEE	Add	

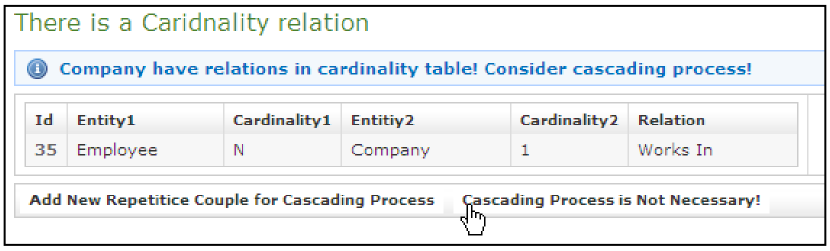
OOI	DG	DM
Employee	Info	Write
Employee	Info	Write

Add New
Add Movement and Continue
Add Movement and proceed for Another Couple

Functional user requirement	Functional process	FP Type	Data movements		
			OOI	DG	DM
[FUR_1]	EMPLOYEE	Add	Employee	ID	E

Figure 3.13 Repetitive Part Warning

When the measurer clicks the continue button, software checks the cardinality table. If there is a relation where the OOI of the DM has a cardinality of 1, then software asks the user whether she wants to add a cascading process or not (Figure 3.14). If the user wants to add another repetitive movement, software will return to previous screen, if she wants to continue, she will be directed to the last part to enter Error/confirmation movement (Figure 3.15).



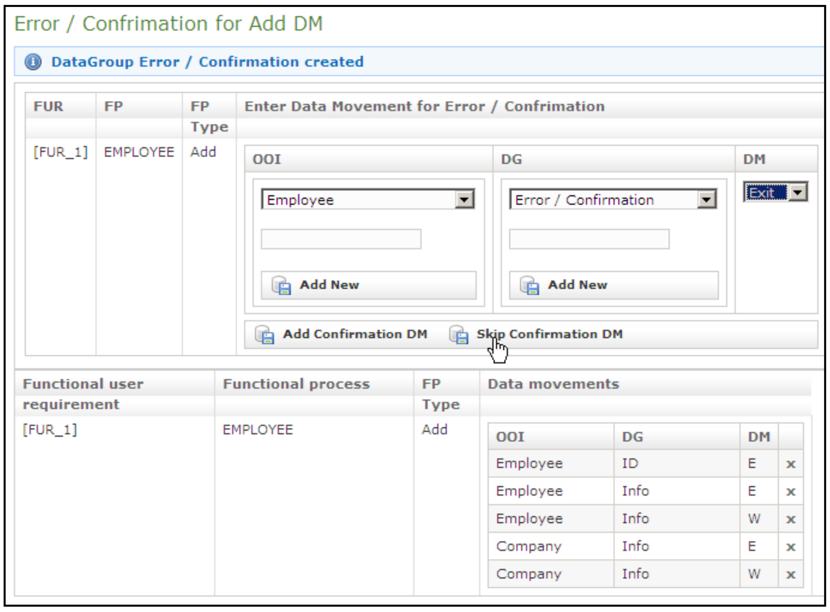
There is a Caridnality relation

Company have relations in cardinality table! Consider cascading process!

Id	Entity1	Cardinality1	Entitiy2	Cardinality2	Relation
35	Employee	N	Company	1	Works In

Add New Repetitice Couple for Cascading Process
Cascading Process is Not Necessary!

Figure 3.14 Cascading Relation Warning



Error / Confrimation for Add DM

DataGroup Error / Confirmation created

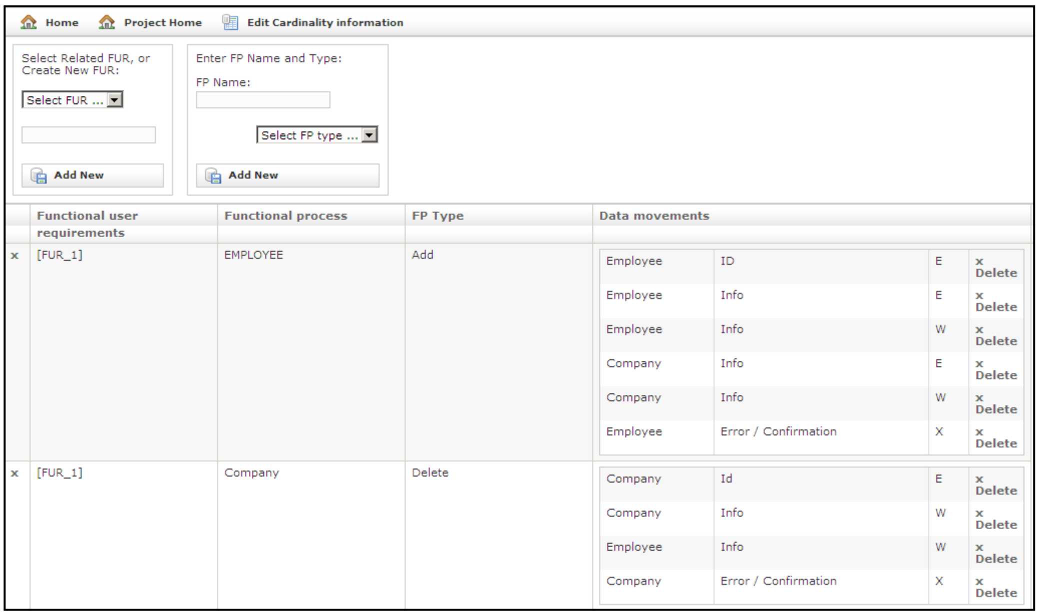
FUR	FP	FP Type	Enter Data Movement for Error / Confrimation
[FUR_1]	EMPLOYEE	Add	OOI: Employee; DG: Error / Confirmation; DM: Exit

Functional user requirement	Functional process	FP Type	Data movements
[FUR_1]	EMPLOYEE	Add	

OOI	DG	DM	
Employee	ID	E	x
Employee	Info	E	x
Employee	Info	W	x
Company	Info	E	x
Company	Info	W	x

Figure 3.15 Error / Confirmation

In Error/Confirmation screen measurer should enter OOI and DG names and choose Exit DM. User may also skip this part, she should decide according to the software artifact she is measuring. When this part is completed, measurer will be directed to the main screen where she can see all the FP's entered and continue to measuring the software (Figure 3.16).



	Functional user requirements	Functional process	FP Type	Data movements			
x	[FUR_1]	EMPLOYEE	Add	Employee	ID	E	x Delete
				Employee	Info	E	x Delete
				Employee	Info	W	x Delete
				Company	Info	E	x Delete
				Company	Info	W	x Delete
				Employee	Error / Confirmation	X	x Delete
x	[FUR_1]	Company	Delete	Company	Id	E	x Delete
				Company	Info	W	x Delete
				Employee	Info	W	x Delete
				Company	Error / Confirmation	X	x Delete

Figure 3.16 Preventive COSMIC Main Screen With Measurement Detail

Measurers should enter every FP in the software with this method. They can track the functional processes in the main screen of repetitive COSMIC.

After the software is completed, we conducted a second case study by using preventive COSMIC to validate both the model and the software. In the next chapter both the exploratory case study and the case study to validate the model will be discussed in detail.

CHAPTER 4

CASE STUDIES

To build and test our model we designed two single-case studies. The purpose of the first case study was to propose a model that has potential to prevent errors during measurement. Second case study was conducted to validate the model.

We conducted an exploratory case study to build and verify our preventive model. The case study showed that it is possible to build a model that might help measurers during the measurement process by reducing the errors. To validate this result, we building preventive COSMIC using the model and conducted another case study and observed that the model helps the measurer to reduce the errors during the measurement.

In the first exploratory case study, we used measurement results of three student groups. In these 3 measurements we found the errors, classified them, and inspect to form DM patterns. We observed that with the proposed model we can reduce the errors up-to 98%.

In the second case study, we took measurement result of a COSMIC measurer prior to development of the model and identified the errors. In the second phase same measurer measured the same software, with Preventive COSMIC. Second measurement is examined to identify the errors and we compared the results of two measurements. Results showed that Preventive COSMIC helps the measurers to reduce the errors during measurement.

4.1. Case Study Design

In this research we aimed to find a structured model to prevent errors during measurement process. We started with the first question: “What are the common errors in COSMIC measurement?” We collected common errors from the works of Özcan Top et al. (2009) and Ungan et al. (2009), and after collecting common errors we asked our second and third questions: “Can we prevent common errors with a structure model?” and “What is the mechanism to prevent the errors with a structured model?” Two find the answers of these two questions we conducted our first case study. In the case study we identified errors which can prevented by a structured model, we analyzed the errors and found that these errors are grouped around some basic FP types. After evaluating the errors and FP types, we proposed a two step model. At first step measurer use FP type as base unit during the measurement and follows defined DM patterns for chosen FP type. At the second step measurer should utilize ER diagram of the software to decide cascading processes.

For the first case study we chose to work with measurement results of METU II students, because measurements of students expected to include common errors and we could easily observe error mechanisms. Moreover the measurements were conducted as a part of their class and as they will get credit from these measurements they expected to pay attention to the case study. In the case study we found the errors, classified them build the model and observed that using a preventive model with a structured rule set would prevent the errors up to 98%. We started to build the software to be used in the second case study.

We designed second case study after having positive results from the first one. Our purpose in the second case study was to validate that preventive model actually reduce errors in a measurement. To reach this purpose same measurer measured same software with and without preventive model with one year interval. Measurer first measured a software nearly one year prior to the development of preventive model and then she measured again by using preventive model. We compared the results of two measurements and observed that the model helps the measurer to reduce the number of errors by 90% in the case study. As a result our structured model helped the measurer to prevent errors during measurement process.

In the next sections details of the case studies are given.

4.2. Exploratory Case Study Conduct

We conducted the first case study to examine common errors and build the prevention model. In this study, we used measurement results of student groups that took the Software Project Management class in METU Information Systems department. Students taking the class were graduate students and working in software related jobs. All of the students taking the class had attended COSMIC lessons and recitations given by same lecturers. Most of the students applied for COSMIC measurer certificate exam.

The students were given a software requirement specifications document to measure as homework. Students formed groups of two or three and measured the software as groups. The software was an MIS application and it is measured as 463 COSMIC function points (CFP) by expert measurers before the homework.

For the case study we randomly chose three groups. There were two students in each group and each group measured the same artifact. Two experienced COSMIC measurers evaluated the reports and identified the errors by comparing the measurements to an answer key that is prepared before the homework.

After the errors are identified; we analyzed them and divided into three main groups; errors related with data movements in FP, errors related with cascading processes, and errors related with the interpretation of the measurer. First group of errors include forgotten movements in functional processes such as; forgetting triggering entry, forgetting confirmation, forgetting a Read movements in a list process… An error in one FP is counted as one error even though it may result more than one missing or unnecessary data movements. Second group of errors includes ignoring cascading processes in an FP such as forgetting to delete employee information while deleting company information. In second group omitted OOI's counted as 1 error, again one error may result more than one DM miscount. Last group of errors are related with measurers' interpretation of artifacts such as ignoring a retrieve process even though it is required in the artifact.

The errors at the first and second group are our target to prevent with the preventive process. The quality of the artifacts or experience level of the measurer may result in the errors in the third group and these errors are not our target in Preventive COSMIC.

In Table 4.1 the errors from first and second groups found in case study are listed. In measurements there are five main FP type. Errors can be related with the FP's or with the DM's in them. For example, the error "Missing exits for error/confirmation" is result of a missing DM in a FP, or the error "List is defined as a part of update FP" is result of combining two different FP type.

To prevent these errors we proposed a method in which measurers should choose FP type for each FP and then enter related DM according to predetermined DM pattern. While entering each FP measurer should check ER diagram to determine cascading processes. The way of preventing each error by the method is presented in the Table 4.1 For the last four errors marked with stars, the method may fail because these errors also related with the interpretation of the measurer.

Table 4.1 Errors Found in the Case Study

ERROR	Prevention Method
Different error messages are considered as separate exits	DM Pattern
Missing exits for error/confirmation	DM Pattern
List is defined as a part of update FP	FP Type Option
Retrieve is defined as a part of another FP	FP Type Option
Missing triggering entry	DM Pattern
Cascading delete is ignored	Cardinality Reminder
List & retrieve combined	FP Type Option
Retrieve and update combined	FP Type Option
Query & detail listing combined	FP Type Option
Assumed read before write	DM Pattern
Redundant exits for error/confirmation	DM Pattern*
Assumed a retrieve FP before delete/update FP	FP Type Option*
List before update/delete is ignored	FP Type Option*
Retrieve before update/delete is ignored	FP Type Option*

* These errors are related also with measurers' interpretations.

After we determined the method to prevent common errors, we analyzed the measurements used in the case study to analyze the effects of the model. Table 4.2 shows the number of errors for each team. Line A lists the number of total errors, Team 1 has 38 errors, Team 2 has 45 errors and Team 3 has 115 errors. Line B shows the errors related with data movements, line C shows the errors related with cascading processes and line D shows the errors related with artifact interpretation. Our preventive model expected to prevent the errors in line B and line C. Line E shows the expected improvement rate that is calculated by dividing numbers in B and C by the total number of errors (Equation 4.1).

Expected improvement rates for teams are 68%, 98%, and 90% respectively. These rates show us that if the measurers have been using our model during their measurement, the model would prevent around 90% of the errors for these cases.

Table 4.2 Number of Errors For Teams

		Team 1	Team 2	Team 3
	Total Error Counts (A)	38	45	115
Expected Improvement	**DM Errors (B)**	6	29	97
	Cascading Errors (C)	20	15	7
	Interpretation Errors (D)	12	1	11
	Expected Improvement Rate (E)	68%	98%	90%

Expected Improvement Rate= (B+C) / A (Equation 4.1)

In addition to the rates calculated with the number of errors, we calculated with the size of the calculated software of the teams. In Table 4.3 the size of the software given to the teams is in line A, it is 463 CFP. In line B the sizes, which are calculated by the teams, are given. The difference between calculated size and expected size is also given in the Table 4.3 (B-A). These numbers are proportional with the numbers of errors given in Table 4.1. Team 1 has least numbers of errors and calculated size is the closest one to the software size. Team 3 has most errors and has the highest size difference.

The size of the errors is also given in the Table 4.3 in line C, line D and line E. Minus numbers represent the missing movements. Team 1 has 6 DM related errors and these 6 errors result in total of 7 missing movements. This number is calculated by subtracting missing movements from redundant movements.

Expected size after the model is given in line F and the difference between line A and line F is given in the last line. As seen from the last line, absolute values of the size differences (F-A) are improved when compared with calculated size difference (B-A), except for team 1. The reasons may be that Team 1 has the highest numbers of interpretation errors.

Table 4.3 Size of Errors For Teams

		Team 1	**Team 2**	**Team 3**
	Expected Size (A)	463 CFP	463 CFP	463 CFP
	Calculated Size (B)	456 CFP	628 CFP	643 CFP
	Calculated Size Difference (B-A)	-7 CFP	165 CFP	180 CFP
Expected Improvement	**Total DM Errors Size (C)**	-3 CFP	96 CFP	181 CFP
	Total Cascading Errors Size (D)	-47 CFP	73 CFP	24 CFP
	Total Interpretation Errors Size (E)	43 CFP	-4 CFP	-25 CFP
	Expected Size After Expected Improvement (F)	509 CFP	459 CFP	438 CFP
	Size Difference After Improvement (F-A)	46 CFP	-4 CFP	-25s CFP

In the exploratory case study, even though all the group members were trained by the same instructors and by the same material, there is a clear difference among the number of errors. The education background and the level of experience may be the result of the difference in the number of errors.

We did not include the interpretation errors as expected improvements, because it is not possible to guide the user about the quality of the artifacts with our approach in the preventive model. A different approach may be suggested for future study to improve these errors.

The case study helped us to compose our propositions for the research questions. We classified the common errors and build a model to prevent them. The case study also showed that the preventive model has substantial potential to improve COSMIC measurement process by reducing the errors made by the measurers during measurement.

Although case study provided evidence for the success of the preventive model, there are some validity threads of the case study. We used an MIS software for the measurements; therefore the study may be repeated for software in other domains like real time software. The measurers used SRS document of the software, which is provided at the beginning of the project. Results of a measurement conducted in other phases of software development process may give different results; measurements should be conducted for other phases with related artifacts. Also the size of the measured software is only 463 CFP, in following studies software in different sizes may be used. We use only one software with three measurer groups, both the number of software and measurer groups should be increased to make detailed statistical analyses.

To further test the model and improve our findings, we conducted a second case study after we build preventive COSMIC. In the next section our second case study is explained in detail and the findings are given.

4.3. Validation Case Study Conduct

After completing the exploratory case study by building the preventive model with positive evidence that the model may reduce the errors, we integrated the model into CUBIT and build "Preventive COSMIC" as explained in Chapter 3. Preventive COSMIC is a tool to help user measure software, moreover it also guides the measurers according to the rule set defined in line with the preventive model. If the user tries to enter a possibly wrong movement, it warns the user. It also helps the user to follow cardinality relations among OOI's that are entered in the movements.

We conducted a validation case to validate our findings in the exploratory case study. For the case study, 1 year before the development of preventive model, a measurer measured an SRS document for an MIS software, which has 337 CFP. The measurement evaluated by a certified COSMIC measurer and errors are identified.

After the preventive model is developed and integrated into CUBIT, same measurer measured the same software this time by using Preventive COSMIC. The results are evaluated and errors are compared with the previous measurement.

The results of two measurements are compared and they are shown in Table 4.4 and Table 4.5. At the first measurement there were 62 errors with 9 different error types and measured size were 365 CFP. After the second measurement with preventive COSMIC there were 6 errors with 2 error types and measured size were 335. Total number of errors reduced by 56 and total difference by the expected size is reduced from 28 CFP to 2 CFP. When we measured the improvement ratio with the Equation 4.2, improvement in the number of errors is 90% and Improvement in the size difference is 93%.

When we examine the results of the case study, we observed that its results are similar to the exploratory case study. We have a 90% improvement in the number of errors that proves the preventive COSMIC helps the measurer to reduce the errors during measurement.

Table 4.4 Number of Errors in Second Case Study

	Total Number of Errors	**Size Difference with the Expected Size**
First Measurement (A)	62	28
Second Measurement (B)	6	2
Improvement (C)	90%	93%

$$C = (A-B)/A \qquad \text{(Equation 4.2)}$$

Table 4.5 Size of Errors in Second Case Study

	Total Size (CFP)	Error Definition	Total Effected Size	Count of Error
First Measurement	365	Missing "List" FP	-30	9
		Combined Retrieve FP	0	11
		Missing "Retrieve" FP	-3	1
		Missing "Add" FP	-19	2
		Missing DM	-3	2
		Combined FP	-36	3
		Unnecessary DM Before Confirmation	+20	10
		Unnecessary DM Before Delete	+24	12
		Other Unnecessary DM	+75	12
Second Measurement	335	Missing Retrieve Before Update	-6	2
		Unnecessary Error/Confirmation	+4	4
Expected Size	337			

The measurer still made some errors in the second measurement; she forgot 2 retrieve FP's before Update FP's and she add 4 redundant confirmation DM's even though they weren't defined in the SRS document. However; these errors affect the measurement result only by 2 CFP. These errors can be explained by the attention level of the measurer and her interpretation of the software artifacts.

The case study showed that we can use Preventive COSMIC to prevent measurement errors during measurement. It reduced the errors by 90% in our case studies and helped to give more accurate size measurements according to our evaluation. The software is ready to use and it might be improved by the results of future research.

Although case study reduce the errors successfully by 90%, there are some validity threats of the case study and future studies should consider these points. We used SRS document of an MIS software and only one measurer measured this software. The study may be repeated for software in other domains like real time software, and results may change in other phases of software development process. Measurement should be repeated for other software development phases with related artifacts. Also the size of the measured software is only 337 CFP; in following studies software in different sizes may be used. We used only one software with only one measurer to observe the decrease of the errors of same measurer with and without the preventive model; however the study may be repeated with different group of measurers to make detailed statistical analyses.

Both of the case studies in our research is conducted with students in the same organization, this creates external validty threats for our research. As the case studies were used student measurements, an industry based case study should be conducted and results should be discussed to reach a generalization. The measurers for both case studies were from the same organization, to reach a statistically meaningful results, inter-organizational measurement results should be analyzed.

In the conclusion section we will discuss the significance of the study, results of this research, and possible future work in the area.

CHAPTER 5

CONCLUSION AND FUTURE WORK

In this chapter the results of the thesis study and its contributions are summarized. At the end of the chapter possible future studies are listed.

5.1.Conclusions

Measuring the size of a software project has crucial importance for both project management and software development activities and errors in size measurement process may have large impacts in other phases of software development activities. In this thesis study, we searched for a method to prevent the errors that are made by the measurers during a COSMIC functional size measurement activity.

In the literature review we found research about the reliability of COSMIC measurement, common errors in a measurement, and suggestions to reduce the errors. The suggested improvements successfully reduced the number of errors by increasing the knowledge and experience level of the measurers, however; in our knowledge there isn’t any research about a structured way to prevent the errors during measurement by guiding the measurer.

We started with the errors defined in papers of Özcan Top et al. (2009) and Ungan et al. (2009), and tried to identify ways to prevent the errors. We came out with a two-step model. At the first step measurer uses an ER diagram to prevent errors related with cascading processes. At the second step user follows predefined DM patterns for different types of FP's.

We conducted an exploratory case study to build and test our model. The case study with measurements of METU II students revealed that the model might prevent possible errors up to 98% in those cases. The errors in the measurements used in the case study might be reduced in rates between 68% and 98% if they were using our preventive method. However, some error types cannot be prevented with our model. These errors are usually related with measurers' interpretation of software artifacts and they could be prevented by increasing the quality of the artifacts or increasing the experience level of the measurers.

After we observed the potential improvement as a result of the preventive model, we build software called Preventive COSMIC. Preventive COSMIC guides the measurer to use the rules defined by the preventive model. The software is open to use for academicians and professionals who have interest in the field. Finding of future studies could be implemented into Preventive COSMIC to improve its capabilities.

We conducted a second case study to validate our findings from first case study. A measurer used preventive COSMIC to measure a software that she measured nearly one year ago without having any information about preventive model. When the results of her first and second measurement are compared the findings were parallel with the findings of our exploratory case study.

In the second case study, preventive COSMIC helped to reduce the number of errors by 90% and the measured size of the software is improved by 93%. The results showed us that the model prevents most of the errors during measurement process.

Although we observed in two different case studies that preventive COSMIC helps to reduce the number of errors, further studies should be conducted to test the model with different software artifacts and with different measurers from both academia and industry. In the both case studies, we used measurements of students from the same organization. For generalizability purposes the studies may be repeated with inter-organizational and industry-wide measurement results. Also we used artifacts of an MIS application; the model should be tested by using artifacts of real time application software.

Preventive model significantly improve the reliability of measurement, in our case studies the number of errors are reduced by 90%. The model also improved measured size of the software, in second case study first measurement gave 8% greater size and the model reduced this difference to 1%. Although 8% difference in size may not be significant for small scale projects, in larger projects %8 difference in size may result a %8 difference in budget planning, which can lead to the denial of the project.

Although the model prevents most of the errors we aimed, it does not provide an error free measurement. One group of errors that could not be prevented by the model is the errors associated with the relations between FP types, for example "missing retrieve before update/delete". Model reminds the user "retrieve" FP type, however there isn't any direct way to prevent this type of errors in the model as these types of errors are also associated with the interpretation of the measurers. Model can be improved in future to control the relation between FP types and warn the user about possible errors.

While building the model we assumed that ER diagram is available to the user during measurement, because not having an ER diagram ready in the early stages means that relations between entities may not be clarified and it may not be possible to determine cascading relations with or without the prevention model.

In preventive COSMIC, measurer should enter the ER table before the measurement and it may take nearly 10% of measurement time according to our measurements in the case study. Considering that in the first case study 24% of the errors in average were related with cascading processes, spending 10% of the required time for ER table is not a redundant effort. Moreover, it is planned to add the ability to accept other ER diagram format instead of manual entry of ER table.

As a result, this study has two considerable contributions to the field of software project management; defining a model to prevent the errors during COSMIC functional size measurement activities; and developing preventive COSMIC software, which guides the measurers during measurement activities, eases the measurement, and decreases the time spend for measurement with preventive model.

5.2.Future Studies

We defined preventive model for COSMIC functional size measurement and validate the model with two case studies, however we used MIS software in our case studies. The model should also be tested by using software in different domains.

We used requirement specification documents to test the model in our research, the model should be tested in different phases of software development with different artifacts.

We build and examined the preventive model for only COSMIC FSM, similar models might be defined for other FSM methods such as IFPUG and MkII.

In preventive COSMIC software we defined a simplified version of ER diagram for simplicity however software might be improved to extract cardinality information from other formats such as ER diagram graphics.

We developed the preventive COSMIC software to help the user through measurement process, however after the measurement user may want to edit the results. Software has the capability to edit the results, but it does not guide the user according to the preventive model while editing. The capability may be added to the software in the future.

REFERENCES

Abrahao, S., Poels, G., & Pastor, O. (2004). Assessing the reproducibility and accuracy of functional size measurement methods through experimentation. *Proceedings of Empirical Software Engineering, ISESE '04,* 189- 198.

Albrecht, A. (1979). Measuring Application Development Productivity. *Proceedings of IBM Application Development Symposium,* 83-92.

Albrecht, A.J., & Gaffney J.E., Jr. (1983). Software function, source lines of code, and development effort prediction: A software science validation. *IEEE Transactions on Software Engineering, 9(6),* 639-647.

Albrecht, A.J. (1984). AD/M Productivity Measurement and Estimate Validation. *IBM Corporate Information Systems, IBM Corp., Purchase, N.Y.*

CMMI Product Team (2010) *CMMI® for Development, Version 1.3*. Addison-Wesley Professional

CUBIT, Retrieved March 18, 2012, from http://smrg.ii.metu.edu.tr/cubit

Gencel, Ç., Buglione, L., Demirors, O., & Efe, P. (2006). “A Case Study on the Evaluation of COSMIC-FFS and Use Case Points”. *3rd Software Measurement European Forum (SMEF 2006),* 121-138.

Gencel, Ç., & Demirors, O. (2008). Functional size measurement revisited. *ACM Transactions on Software Engineering and Methodology, 17(3).*

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

ISO/IEC (2002b). 20968: Software engineering -- Mk II Function Point Analysis -- Counting Practices Manual

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

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

ISO/IEC (2005). 24570: Software engineering -- NESMA functional size measurement method version 2.1 -- Definitions and counting guidelines for the application of Function Point Analysis

ISO/IEC (2006). 14143-6: Information technology -- Software measurement -- Functional size measurement -- Part 6: Guide for use of ISO/IEC 14143 series and related International Standards

ISO/IEC (2007). 14143-1: Information Technology – Software Measurement - Functional Size Measurement - Part 1: Definition of Concepts.

ISO/IEC (2009). 20926: Software and systems engineering -- Software measurement -- IFPUG functional size measurement method 2009

ISO/IEC (2010). 29881: Information technology -- Systems and software engineering -- FiSMA 1.1 functional size measurement method

ISO/IEC (2011a). 14143-2: Information technology -- Software measurement -- Functional size measurement -- Part 2: Conformity evaluation of software size measurement methods to ISO/IEC 14143-1

ISO/IEC (2011b). 19761: Software engineering -- COSMIC: a functional size measurement method

Kemerer, C.F. (1993). Reliability of function points measurement: a field experiment. *Communications of the ACM 36 (2),* 85-87.

Kemerer, C.F., & Porter, S. (1992). Improving the Reliability of Function Point Measurement: An Empirical Study. *IEEE Transactions on Software Engineering, 18(11)*, 1011-1024.

Low, G.C., & Jeffery, D.R. (1990). Function points in the estimation and evaluation of the software process. *Software Engineering, IEEE Transactions, 16(1),* 64 – 71

Özcan Top, O., Demirörs, O., & Özkan, B. (2009). Reliability of COSMIC Functional Size Measurement Results: A Multiple Case Study on Industry Cases. *Proceedings of 35th Euromicro Conference on Software Engineering and Advanced Applications, IEEE Computer Society,* 327-334

The Common Software Measurement International Consortium (COSMIC): Guideline for Assuring the Accuracy of Measurements Version 1.0. (2011)

The Common Software Measurement International Consortium (COSMIC): COSMIC Method Measurement Manual Version 3.0.1. (2009)

Türetken, O., Demirörs, O., Özcan Top, O., & Özkan, B. (2008a). The Effect of Entity Generalization on Software Functional Sizing: A Case Study. *Product-Focused Software Process Improvement 5089/2008*, 105-116

Türetken, O., Özcan Top, O., Özkan, B., & Demirörs, O. (2008b). The Impact of Individual Assumptions on Functional Size Measurement. *IWSM / MetriKon / Mensura 2008, 5338*, 164–178.

Ungan, E., Demirörs, O., Özcan Top, O., & Özkan B. (2010). Evaluation of Reliability Improvements for COSMIC Size Measurement Results. *IWSM / MetriKon / Mensura 2010.*

Ungan, E., Demirörs, O., Özcan Top, O., & Özkan, B. (2009). An Experimental Study on the Reliability of COSMIC Measurement Results. *IWSM/Mensura 2009,* 321-336.

Yılmaz, G., Ungan, E., & Demirörs, O. (2011). Yazılım Gereksinim Dokümanı Kalitesinin İşlevsel Büyüklük Ölçümüne Etkisi [Effects of the Quality of Software Requirement Specifications Document to Functional Size Measurement]. *5. Ulusal Yazilim Mühendisliği Sempozyumu - UYMS'11.*