EVENT POINTS: A SOFTWARE SIZE MEASUREMENT MODEL

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

TUNA HACALOĞLU

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

SEPTEMBER 2021

Approval of the thesis:

EVENT POINTS: A SOFTWARE SIZE MEASUREMENT MODEL

Submitted by **TUNA HACALOĞLU** in partial fulfillment of the requirements for the degree of **Doctor of Philosophy in Information Systems Department, Middle East Technical University** by,

Date: 10.09.2021

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 : Tuna Hacaloğlu

Signature : _____

ABSTRACT

EVENT POINTS: A SOFTWARE SIZE MEASUREMENT MODEL

Hacaloğlu, Tuna
Ph.D, Department of Information Systems
Supervisor: Assoc. Prof. Dr. Aysu BETİN-CAN
Co-Supervisor: Prof. Dr. Onur DEMİRÖRS

September 2021, 118 pages

Software Size Measurement is a critical task in Software Development Life Cycle (SDLC). It is the primary input for effort estimation models and an important measure for project control and process improvement. There exist various size measurement methods whose successes have already been proven for traditional software architectures and application domains. Functional size measurement (FSM) being one of them attracts specific attention due to its applicability at the early phases of SDLC. Although FSM methods were successful on the data-base centric, transaction-oriented stand-alone applications, their applicability on the new generation software architectures are not studied well. Today software is frequently service based, highly distributed, message driven, scalable and having unprecedented levels of availability. In these architectures, 'event' concept largely replaces the 'data' concept. In this thesis, considering the significance of the event concept in today's software systems, we explored the potential of an event-based software size measurement method. For this aim, we collaborated with 5 software organization and conducted multiple case studies. As a result of this research, it is seen that the proposed model produce promising results; "Event points" correlates well with effort. According to the findings of our study, it can be concluded that event as base counting unit can be used for measuring software size for both traditional and novel architectures, it is possible to perform a measurement without considering the data as a counting base, event-based effort estimation models yield acceptable error rates and prediction performance in the effort estimation models.

Keywords: Software size measurement, event, COSMIC, case study, effort estimation

ÖZ

OLAY PUANI: YAZILIM BÜYÜKLÜK ÖLÇÜM MODELİ

Hacaloğlu, Tuna
Doktora, Bilişim Sistemleri Bölümü
Tez Yöneticisi: Doç.Dr. Aysu BETİN-CAN
Eş Danışman: Prof.Dr. Onur DEMİRÖRS

September 2021, 118 sayfa

Yazılım Büyüklük Ölçümü çeşitli kestirimler için birincil girdi olmasının yanı sıra, proje yönetimi ve süreç iyileştirme için önemli bir gösterge olması açısından Yazılım Geliştirme Yaşam Döngüsü (YGYD) için kritik bir iştir. Geleneksel yazılım mimarileri ve uygulama alanları için başarısı kanıtlanmış birçok büyüklük ölçüm yöntemi bulunmaktadır. Bu yöntemlerden biri olan İşlevsel Büyüklük Ölçümü (İBÖ), YGYD'nin erken fazlarında uygulanabilirliği sayesinde özel olarak dikkat çekmektedir. İBÖ yöntemleri veri tabanı odaklı, işlem yönelimli, yekpare uygulamalar üzerinde başarılı olmuş olsalar da, yeni nesil yazılım mimarileri üzerinde ne kadar uygulanabilir oldukları yeterince araştırılmamıştır. Günümüzün yazılımlarının sıklıkla servis-odaklı, oldukça dağıtık, mesaj-yönelimli, ölçeklenebilir olmalarının yanı sıra ve örneğine rastlanmamış bir biçimde her an hazır olmaları beklenmektedir. Bu mimarilerde "olay" kavramı genellikle "veri" kavramının yerine geçmektedir. Bu tezde olay kavramının günümüz yazılımlarındaki önemini dikkate alarak, olay-tabanlı bir ölçüm yöntemi oluşturulması amaçlanmıştır. Bu amaçla, 5 yazılım organizasyonu ile işbirliği yapılmış ve çoklu vaka çalışmaları gerçekleştirilmiştir. Araştırmanın sonucunda, önerilen modelin anlamlı sonuçlar ürettiği: "Olay Puanı" ölçüm biriminin efor ile ilişkili olduğu görülmüştür. Elde edilen bulgulara göre, olay kavramının temel sayma birimi olarak umut vadettiği, veri odaklı bir ölçüm temelini dikkate almadan hem geleneksel hem de yeni nesil mimarilerde ölçüm yapma imkanı sağladığı ve olay ile oluşturulan efor kestirim modellerinin kabul edilir hatalar ve kestirim başarısı sağladıkları saptanmıştır.

Anahtar Sözcükler: Yazılım büyüklük ölçümü, olay, COSMIC, durum çalışması, efor kestirimi

To my mother Nazan Hacaloğlu and
my father Tuncay Hacaloğlu,

ACKNOWLEDGMENTS

I would like to thank to my supervisor Assoc.Prof. Dr. Aysu Betin-Can, who supported me in this long process, for her encouragement and all the helps to overcome this challenge.

I would like to express my gratitude to my co-supervisor Prof. Dr. Onur Demirörs who encouraged me to conduct this thesis study, even in the times when I lost my hope on what to do. He was always positive, encouraging, and available whenever I was in trouble with my questions. In addition to his positive attitude, his innovative point of view to the problems inspired me a lot. Apart from knowledge acquisition, I took lessons from him about how to be a good advisor while I experience this long and challenging process. Being his student is an honor for me, I will always be grateful to him.

I would like to thank the jury members Prof. Ali Doğru and Assoc. Prof. Dr. Altan Koçyiğit for their valuable comments during the thesis monitoring meetings.

I am grateful to Selami Bağrıyanık, Tuğba Hafızoğulları, Aylin Deveci, Zehra Gül Çabuk, and Ali Yıldız for their invaluable supports.

I express my special thanks to Assoc. Prof. Dr. Çiğdem Turhan, Assist. Prof. Dr. Bilge Say, Assoc. Prof. Dr. Yavuz İnal, Assoc. Prof. Dr. Deepti Mishra, Aylin Akça Okan, Hüseyin Ünlü, Neslihan Küçükateş Ömüral, Ezgi İlhan and Tolga Metin for their helps, encouragements and motivations during this period.

I am also thankful to my colleagues at Atilim University Information Systems Engineering Department for their helps and understanding they provided during this process.

Last but not least, no words can explain my gratitude to my mother Nazan, my father Tuncay, my sister Tuğçe, my uncle Mustafa and cousin Murat; who were the actual witnesses of this long process and whom I owe all the best things I have in this life.

TABLE OF CONTENTS

ABSTRACT iv
ÖZ v
DEDICATION vi
ACKNOWLEDGMENTS vii
TABLE OF CONTENTS viii
LIST OF TABLES xi
LIST OF FIGURES xii
LIST OF ABBREVIATIONS xiii
CHAPTERS
1. INTRODUCTION 1
1.1. Problem Statement 2
1.1.1. Size Measurement Challenge from Architectural Perspective 3
1.1.2. Size Measurement Challenge from SDLC perspective 5
1.2. Objectives, Contributions and limitations 6
1.3. Research Strategy 8
1.4. Structure of the thesis 12
2. METHODOLOGY 13
2.1. How DSR fits our research? 14
2.1.1. Guideline 1: Design as an artifact 14
2.1.2. Guideline 2: Problem Relevance 15
2.1.3. Guideline 3: Design Evaluation 16
2.1.4. Guideline 4: Research Contribution 18
2.1.5. Guideline 5: Research Rigor 18
2.1.6. Guideline 6: Design as a Search Process 21
2.1.7. Guideline 7: Communication of Research 21
2.2. Applying the Design Science Research Process 21
2.2.1. Problem Identification 22

2.2.2. Solution Design 23
2.2.3. Evaluation.....24
2.3. Summary of the Chapter.....24
3. PROBLEM IDENTIFICATION.....27
3.1. Literature Search I: Systematic Literature Review 27
3.1.1. Review Planning.....27
3.1.2. Conduct of the searches.....29
3.1.3. Reporting the results.....29
3.2. Exploratory Case Study 1.....33
3.2.1. Case Study Design.....33
3.2.2. Description of the Cases.....34
3.2.3. Execution of the Exploratory Multiple Case Study.....35
3.2.4. Results and Discussions 36
3.2.5. Concluding Remarks for Multiple Exploratory Case Study 1 40
3.3. Summary of the Chapter.....40
4. SOLUTION DESIGN 43
4.1. Literature Search II.....43
4.2. Multiple Exploratory Case Study 2 45
4.2.1. Case Study Design.....46
4.2.2. Execution of Case Study 50
4.2.3. Results 50
4.2.4. Implications from the Multiple Exploratory Case Study 2 54
4.3. Literature Search III 54
4.4. Multiple Exploratory Case Study 3 57
4.4.1. Case Study Design.....57
4.4.2. Execution of Case Study 59
4.4.3. Results 60
4.4.4. Implications from the Multiple Exploratory Case Study 3 66
4.5. Summary of the Chapter.....67
5. EVENT POINT: EVENT-BASED SIZE MEASUREMENT MODEL 69
5.1. Description of the Event- Driven Software Model 70

5.2. Description of the Event-based size measurement method 70
5.2.1. Method Principles 71
5.2.2. Mapping roadmap 72
5.2.3. Measurement Stage: Definition of the measurement unit 73
5.2.4. Discussion about COSMIC FSM and Event-based measurement model 73
5.3. Summary of the Chapter 74
6. EVALUATION CASE STUDIES 75
6.1. Case Study Design 75
6.1.1. Case Descriptions 75
6.2. Conduct of the Case Studies in Organization X 77
6.2.1. Measurement Results of Project 1 77
6.2.2. Measurement Results of Project 2 78
6.2.3. Measurement Results of Project 3 79
6.2.4. Analysis of the results 79
6.3. Conduct of the Case Studies in Organization Y 84
6.3.1. Measurement Results 84
6.3.2. Analysis of the Results 85
6.3.3. Implications from the Evaluation Case Study 88
6.4. Summary of the Chapter 89
7. LIMITATIONS AND VALIDITY THREATS 91
8. CONCLUSIONS AND FUTURE WORKS 95
8.1. Conclusions 95
8.2. Future Works 96
REFERENCES 99
CURRICULUM VITAE 117

LIST OF TABLES

Table 1 Distribution of Articles per Database (Hacaloğlu & Demirörs, 2018) 29
Table 2 Size measures/ estimates and articles (Hacaloğlu & Demirörs, 2018) 30
Table 3 Project demographics adopted from (Hacaloglu & Demirors, 2019) 36
Table 4 Measurement results of Project 1 51
Table 5 Measurement results of Project 2 52
Table 6 Measurement results of Project 3 53
Table 7 Event types mentioned in the literature 55
Table 8 Events considered and examples 58
Table 9 Description of Cases 59
Table 10 Measurement Results of Project 1 61
Table 11 Measurement Results of Project 2 63
Table 12 Measurement Results of Project 3 65
Table 13 Summary of Measurement Results 67
Table 14 Measurement Results of Projects of Organization Y 77
Table 15 Data collected in Project 1 78
Table 16 Data collected in Project 2 78
Table 17 Data collected in Project 3 79
Table 18 Descriptive Statistics of Projects 79
Table 19 Correlation Analysis Results of Project 1, 2 and 3 81
Table 20 Linear Regression Results 82
Table 21 Project 3 Regression Results after excluding the outlier 83
Table 22 Accuracy Assessment 83
Table 23 Model Validation using LOOCV 84
Table 24 Measurement Results 85
Table 25 Correlation Analysis Results 86
Table 26 Linear Regression Results 86
Table 27 Accuracy Assessment 87
Table 28 Model Validation using LOOCV 88

LIST OF FIGURES

Figure 1 Research Roadmap 9
Figure 2 Multiple case study process by (Yin, 2017) 17
Figure 3 Research Activities adapted from (Offermann et al., 2009) 22
Figure 4 Size measures/ estimates types 30
Figure 5 COSMIC Functional Size Measurement Phases (COSMIC, 2017b) 35
Figure 6 Model Version 1 48
Figure 7 Syntax Check Facility of Bflow Toolbox 49
Figure 8 Scatter Plots of Project 1.....52
Figure 9 Scatter plots of Project 2.....53
Figure 10 Synthesis of events from the literature 55
Figure 11 Model Version 2 56
Figure 12 Scatter Plots of Project 1.....62
Figure 13 Scatter Plots of Project 2.....63
Figure 14 Scatter Plots of Project 3.....64
Figure 15 Model Version 3 70
Figure 16 Categorization of Events.....72
Figure 17 Scatter Plots of Project 1, 2 and 3.....80
Figure 18 Scatter plots of Project 3 after excluding the outlier 82
Figure 19 Scatter plots of Projects of Organization Y 86

LIST OF ABBREVIATIONS

ACID	Atomicity, Consistency, Isolation, and Durability
BFC	Base Functional Components
BPMN	Business Process Management
CFP	COSMIC Function Points
DDD	Domain-Driven Design
DSR	Design Science Research
EDA	Event-Driven Architecture
EPC	Event Driven Process Chain
eEPC	extended Event Driven Process Chain
FSM	Functional Size Measurement
JSF	Java Server Faces
MRE	Magnitude of Relative Error
MVC	Model View Controller
LOC	Line of Code
LOOCV	Leave-One-Out-Cross Validation
MMRE	Mean Magnitude of Relative Error
MdMRE	Median Magnitude of Relative Error
OOI	Object-of-Interest
OOP	Object-Oriented Programming
PRED(x)	Percentage Relative Error Deviation within x
SDLC	Software Development Life- Cycle
SLR	Systematic Literature Review
SMTP	Simple Mail Transfer Protocol
UML	Unified Modelling Language

CHAPTER 1

INTRODUCTION

Software measurement is one of the most critical practices in software engineering. Measurement enables organizations to understand, control and improve their projects and processes (Fenton & Pfleeger, 1997). In this context, perhaps, one of the most significant measurement attributes of a software is its size. Software size is such a critical attribute that, in its absence, it becomes very difficult to plan, estimate, and control large-scale projects objectively (Gencel & Demirors, 2008). In addition, size measurement provides managers opportunities for managing, maintaining, improving projects and also chances to make comparisons among them (Abran, Symons, Ebert, Vogelezang, & Soubra, 2016).

According to Fenton & Pfleeger (1998), software size can be described with three main attributes; namely length, complexity and functionality. Among these, when it is compared to others, measuring the size of a software in terms of its functionality, in other words, "functional size" has special prominence. Especially the applicability of functional size measurement (FSM) at the early stages of the software development life cycle (SDLC), due to its independence of implementation tools and languages (Ren & Yun, 2013) render it a powerful technique. Moreover, since its measurement procedure is made up of systematic rules (Abran, Desharnais, Zarour, & Demirörs, 2015) and it is objective and repeatable (Commeyne, Abran, & Djouab, 2016); it is therefore less prone to measurers' subjective judgements. Thus, by using functional size, organizations can build consensus with supplier and acquirer, monitor the scope changes of their projects, normalize their performance and quality measures for process assessment and improvement (Ozkan, Turetken, & Demirors, 2008).

Functional size measurement (FSM) related studies have been significantly evolved during the last two decades. Today, there exist five FSM methods which are already ISO/ IEC (ISO, 2007)– compliant (Czarnacka-Chrobot, 2009). However, as a result of the technological improvements, the bases of some of today's software systems differ significantly from the ones for which available FSM methods were developed. Many of today's software systems are characterized by being interconnected, online, programmable and showing a continuous shifting behavior from the traditional single device computing, to decentralized multi-device architecture (Mongiello, Nocera, Di Sciascio, & Di Noia, 2018). New generation software is considerably different from the previous one, specifically monoliths, which were designed to use the resources such as database, memory and files of the same machine (Dragoni et al., 2017). This traditional

concept of a single program that is responsible for everything such as displaying the user interface, accessing data etc. Stephens (2015) has been left behind in today's application ecosystem, such that Andriole (2017) describes this situation as "the entire world of big software design, development, deployment and support is dead." (p.32). In addition, a progressive shift is being observed towards distribution, modularization and loose coupling (Mazzara, Bucchiarone, Dragoni, & Rivera, 2020).

(Dragoni et al. (2017) emphasize the difference of these new architectures such as microservices being independent and deployed in isolation; from the monoliths which use the resources of the same machine. Especially, Cloud Computing created a revolution on contemporary software architecture in terms of behavior, structure, style and topology due to its dynamic resource access, resource pooling, rapid elasticity, dynamicity and multi tenancy characteristics (Bahsoon, Ali, Heisel, Maxim, & Mistrik, 2017).

1.1. Problem Statement

The paradigm shift has reflections in software size measurement perspective. Despite their maturity and benefits the available FSM methods bring, their applicability is no longer straight-forward in the new generation of software architectures. There exist different logics behind both dimensions. Available FSM methods are transaction-based and designed to measure the functional size of a given software artifact by binding the functionality with data. In systems for which the FSM methods are designed, there exists a single data model belonging only to the application. However, new generation software by being highly distributed, loosely-coupled and service-oriented is completely different from the monoliths. In these new architectures, there is not an endeavor to fit all data into a single data model (Mistrík, Bahsoon, Ali, Heisel, & Maxim, 2017).

Consequently, relational databases are no more favored in designing distributed systems since they have static schema and rigid structure (Zaki, 2014), they do not scale horizontally, and they have join and lock characteristics as negative effector in terms of distributed systems performance (Hecht & Jablonski, 2011). Therefore, in contemporary software development it is an observed fact that software developers are not creating traditional data analysis techniques such as creating Entity-Relationship Diagrams, Relational data analysis etc. (Hacaloglu & Demirors, 2019). Even though available functional size measurements are still relevant, we also believe that their philosophy of focusing on transaction and data becomes incompatible and sometimes challenging with new architectural considerations.

Today's applications Facebook, Netflix, Uber and others communicate with the help of lightweight protocols such as REST API etc. (Sucaciu, 2020). Contemporary giant service providers such as LinkedIn, Netflix and Amazon are using event-oriented microservice based architectures (Riggins, 2017). In new architectures, such as microservices, events provide the communication through system services, rather than functions which convey data flowing in the system. The working mechanism of these systems can be broadly

explained as follows: upon updating its data, a service publishes an event where other services receive that event because they are subscribed to it. This way of transitioning from data to events is promising to explore, identify and describe the improvement opportunities for size measurement in the context of the new generation software architectures. By motivating from the importance of events (Boner, 2017; Parulkar, 2020a; Sucaciu, 2020) in nowadays famous software applications, for example, eBay, PayPal and LinkedIn (Zhu, Richins, Halpern, & Reddi, 2015), we explore the relevance of events as base measurement components.

The problems in this context are presented in detail in two-categories in details in the following sub-sections 1.1.1 and 1.1.2.

1.1.1. Size Measurement Challenge from Architectural Perspective

Today's software systems are described as the applications operating on multitude of platforms including mobile devices and cloud-based clusters having thousands of servers comprising multi-core processors where users expecting responses in milliseconds and 100% accessibility requiring the use of reactive systems (Bonér, Farley, Kuhn, & Thompson, 2014). Shifting applications to the cloud and building microservice architecture are shown as the facts dominating the IT market and causing applications to no longer share the common computational space but to communicate using lighter protocols (Sucaciu, 2020). Conventional design approaches such as Object-Oriented Programming (OOP) and Domain-Driven Design (DDD) which are focusing on the things, in other words, the structural aspects of system should be replaced with behavioral perspective, in terms of the flow of events (Boner, 2017). For this reason, newer patterns such as Event-Driven Architecture, Microservices, Space-Based Architecture become a natural choice by addressing specifically to the scalability issue which is a must in today's systems (Richards, 2015).

Event-Driven Architecture (EDA) has attracted attention recently (Bonér et al., 2014; Laigner et al., 2020) especially in cloud-based applications where high availability and data throughput are needed (Zhelev & Rozeva, 2019). EDA is a distributed architectural pattern composed of decoupled single-purpose asynchronous event processing components (Richards, 2015) where the communication among the components are provided with events' production and consumption (Parulkar, 2020a; *The Reactive Manifesto v1.0*, 2013).

The architectural paradigm shift poses a challenge for functional sizing in which transaction and data type play a significant role in the measurement process. Before explaining the challenge, it is important to describe the foundations of these methods. Even though they use different counting schema, IFPUG and COSMIC, being widely used FSM methods, follow a common similar philosophy to measure the functional size of a given software artifact; by identifying functional user requirements, extracting base

functional components (BFCs) and aggregating the results (Demirors & Gencel, 2009). In these methods, functional user requirement corresponds to a set of transactions and data-types. Data type in these methods is an attribute of an object of interest (OOI) related to a transaction (Demirors & Gencel, 2009). Although each method defines their own concepts for BFC- that is the elementary measurement unit, in fact, they have a similar philosophy. These methods are transaction based and highly data-centric which bind the functionality with data and consider the system as a flow data. They were suggested to measure the functional size of previous generation architecture -monoliths-. Monoliths were designed in a way to use the database, memory and files of a single machine (Dragoni et al., 2017).

Relational databases which were highly used in Monoliths, cannot accommodate new architectures because they have strict relational schema, a normalized data model, and ACID property yielding lack of horizontal scaling and join and lock concepts affecting the performance in a negative way in distributed systems (Hecht & Jablonski, 2011). Google, Amazon, Facebook and LinkedIn are the first companies to discover the limitations of relational databases in fulfilling the requirements of big users and big data (Zaki, 2014). Specifically, with the proliferation of Cloud Computing and systems involving big data analytics, non-relational database models are being used frequently (Banerjee & Sarkar, 2016; Han, Haihong, Le, & Du, 2011). Mistrík et al. (2017) draw attention to the organizations recognizing the need for information management techniques which have to be data platform and data type agnostic as much as possible.

In the old generation architectures functions are called as data transfer where in new generation architectures these communications are provided using event-queues. As a result, in contemporary software development it is an observed fact that software developers are not conducting traditional data analysis techniques such as creating Entity-Relationship Diagrams, Relational data analysis etc. (Hacaloglu & Demirors, 2019).

Therefore, relational, data-centric approach which constitutes the basis of functional size measurement philosophy has become less used in new architectures. Therefore, available functional size measurement methods because of being data-centric face difficulties to adapt to these new software architectures, yields in less accurate results and it remains unclear how to tackle the size measurement in these systems. This situation causes a pursuit of a new approach for software size measurement that is compatible with the elements the new architectures bring.

Gartner anticipated that event-based and real time solutions will be requested in 80% of all digital applications used in the industry by 2020 (Chima, 2018). Naturally, there exist organizations which are World's pioneers such as eBay, PayPal and LinkedIn that have already noticed the importance of accommodating this change and adopted event-driven philosophy (Zhu et al., 2015).

Considering the important place of events in contemporary architectures we focused on approaching the software size measurement problem from the event-driven perspective.

In addition to the three main attributes of software size namely length, complexity, and functionality (Fenton & Pfleeger, 1998). In this thesis study, we aimed to explore the appropriateness of events as another software size attribute by conforming to today's architectural paradigms.

1.1.2. Size Measurement Challenge from SDLC perspective

In addition to the architectural paradigm shift, SDLC adopted in contemporary software development pose challenge in software size measurement. Even though the process model adopted and the technological paradigm used are two different concepts, in practice, they are closely related. Today's expectation is to deliver a working software product as quickly as possible, accommodating to the changing requirements and conditions in order to both satisfy the needs of customers and beat the competitors. Consequently, market demands for faster delivery of working software products encourage the adoption of Agile Methods as an SDLC model in software industry (Rodríguez et al., 2019). For example, microservice architecture as being an example of the latest trend in contemporary software development; increases the software agility need by being developed, deployed, versioned and scaled as independent units (Jamshidi, Pahl, Mendonça, Lewis, & Tilkov, 2018).

In this aspect, the characteristics of Agile methodologies which made it specifically prominent for contemporary software can further be explained as follows: Agile methods are people-oriented, welcome change requests in every steps (Javdani, Zulzalil, Ghani, Sultan, & Parizi, 2013; Popli & Chauhan, 2014b), focus on product development, offer fast delivery in an iterative manner, are simple to follow with self-organizing teams, improve quality continually (Javdani et al., 2013).

Since it was first introduced in 2001 as a manifesto to the traditional software development methodologies (Fowler & Highsmith, 2001), Agile practices changed the way software is developed (Dingsøyr, Falessi, & Power, 2019) and are being well accepted in the community so that it became more mainstream in industry (Barroca, Sharp, Salah, Taylor, & Gregory, 2018) and academia have strong interest on it (Hoda, Salleh, Grundy, & Tee, 2017).

On the other hand, despite the fact that Agile can be considered as a mature paradigm at the present time, it still possesses challenges from the size measurement and effort estimation perspective. Although objective size measurement have many benefits such as constructing estimation models, benchmarking, improving internal processes and project governance (Buglione & Trudel, 2010), Agile practitioners still mostly prefer and adopt subjective estimates such as use case points and story points (Usman, Mendes, Weidt, & Britto, 2014) in place of constructing effort models based on objective size measurement. Although, in Agile community, story points are well accepted estimates allowing the comparison of actual and estimated effort, they criticized by not being a size measure; thus not allowing to analyze the productivity of a project and to compare across projects and organizations (Commeyne et al., 2016).

Furthermore, there are evidences in the literature proving that FSM-based effort estimations outperforming story points (Commeyne et al., 2016; Salmanoglu, Hacaloglu, & Demirors, 2017; Ugalde, Quesada-López, Martínez, & Jenkins, 2020; Ungan, Cizmeli, & Demirors, 2014). Still, there is a resistance among Agile practitioners to learn an unfamiliar FSM method which they do not believe that it will improve the quality and schedule of project; instead of story points which they think they are already familiar and successful (Buglione & Trudel, 2010).

On the other hand, Agile practices encourage iterative and incremental evolution of projects, therefore, requirements are not complete at the initial phases of SDLC but come to a mature state when they are actually implemented. In addition, they are written for main functions alone (Paetsch, Eberlein, & Maurer, 2003), in a manner containing fewer details, generally kept as brief as possible.

This situation specifically complicates the utilization of standardized Functional Size Measurement (FSM) methods. This problem has been identified in a literature review study by (Hacaloğlu & Demirörs, 2018) where it is reported that, for example, (Hussain, Kosseim, & Ormandjieva, 2013) emphasize specifically for COSMIC FSM that, lack of details and formal expression prevent the measurers from identifying the measurement components for instance "data groups" (Hussain et al., 2013) thus limit the measurement capabilities. In addition, especially, varying level of abstractions posing the necessity of making an adjustment to bring the analysis documents in an equal abstraction level for measurement (Salmanoğlu, Öztürk, Bağrıyanık, Ungan, & Demirörs, 2015) which is perceived as a costly and undesirable effort (Hussain et al., 2013). Lack in detail in requirement documentation can be shown as one of main reason for this situation (Usman, Mendes, & Börstler, 2015).

Based on the study by Hacaloğlu & Demirörs (2018) the obstacles on the adoption of functional size measurement methods can be summarized as the difficulty in procedure, mismatch between agility, lack of detailed requirement documentation, and Agile teams' desire of being decision making authority in software development.

1.2. Objectives, Contributions and limitations

All the problems stated in section 1.1 motivated us to explore a software size measurement method which suits better in changing software systems' characteristics and which can also be in accordance with Agile practices.

Accordingly, we aim to handle the size measurement problem in contemporary software from a behavioral perspective rather than a data-centric or structural point of view which is adopted by the available functional size measurement methods. In this context, events are increasingly attracting the attentions due to cloud, multicore architectures, micro-

services and distributed systems, and customer demands (Boner, 2017). Therefore, we envision that event-driven size measurement can be useful to size the contemporary software systems and we aim to define the software size from behavioral perspective, in terms of events where we propose an event-based measurement model.

As the main motivation of this study, we envision that "event" -as a candidate of size attribute- can match with our objectives to overcome software size measurement challenges from architectural perspectives. Since "event" can appear in various levels of abstraction in computer science, we primarily are in the pursuit of how to use the event concept in sizing software requirements. In other words, we aim to assess whether "event" is relevant and convenient as a software size measurement unit by exploring the appropriate granularity level of events for size measurement which will form the baseline of event-based size measurement models.

Within the scope of this thesis, an event-based size measurement model for new-generation software architectures is developed. The applicability and effectiveness of the model is assessed through multiple case studies. The paradigm shift is expected to diffuse day by day into software development domain. Hence, providing such a measurement model's success in both data-centric- traditional software architectures and in new generation projects can be valuable for the practitioners. In this context, proposed measurement model will be the first considering new generation projects and an innovative approach in this scope. It will be beneficial for the organizations which develop both types of projects.

Event-based size measurement model is composed of the identification of events which will be candidate for measurement and aggregation of them. The success of the system is assessed by creating effort estimation models and by comparing the deviation of predicted efforts from the actual ones and also by comparing with COSMIC FSM methods' performance.

With this thesis, we aim to contribute to the literature by describing the characteristics of new generation software that are conflicting with the functional size measurement philosophy and by discovering the improvement opportunities for prospective software size measurement method through an exploratory case studies. Accordingly, this thesis aims following contributions:

- To reveal current state of the art of the measurement in contemporary software projects,
- To identify challenges related with the functional size measurement in these type of projects,
- To discover improvement opportunities regarding the characteristics of contemporary software projects,
- To develop a new size measurement model with events that is suitable both for new generation architectures,

- To classify events for the purpose of measurement which is not yet done in the literature to the best of our knowledge,
- To evaluate the suitability of the proposed model in contemporary projects in software development industry.

With this model we envision that organizations,

- can measure the event-based size of the software without being dependent on the data model,
- can create estimations by using event-based size,
- can create estimations such as multiple regression analysis by further categorizing computation events

1.3. Research Strategy

By motivating with the emergent technology shift to the event driven systems, we hypothesize that a size of an application can be specified in terms of events and event-based sizing can be relevant and can provide more precise effort estimation. For this aim, following research questions (RQ)s are derived:

RQ1. What are the challenges encountered in functional software size measurement on new generation architectures?

RQ2. How events can be used to measure software size?

RQ2.1. What can be the event related measurable components in these new architectures?

RQ2.2. Is event-based size measurement relevant?

RQ2.3. How event-based size measurement compares with FSM?

RQ3. How useful are events in software effort estimation?

To answer these research questions, we planned multiple case studies and literature searches. As the research strategy, Design Science Research (DSR) method is chosen in this thesis. While conducting the research, we adopted the guidelines for DSR proposed by Hevner et al. (2004) and the DSR process model suggested by Offermann et al. (2009). Details concerning the methodology is explained in depth in Chapter 2. To be in line with the objectives of this thesis, roadmap visualized in Figure 1.

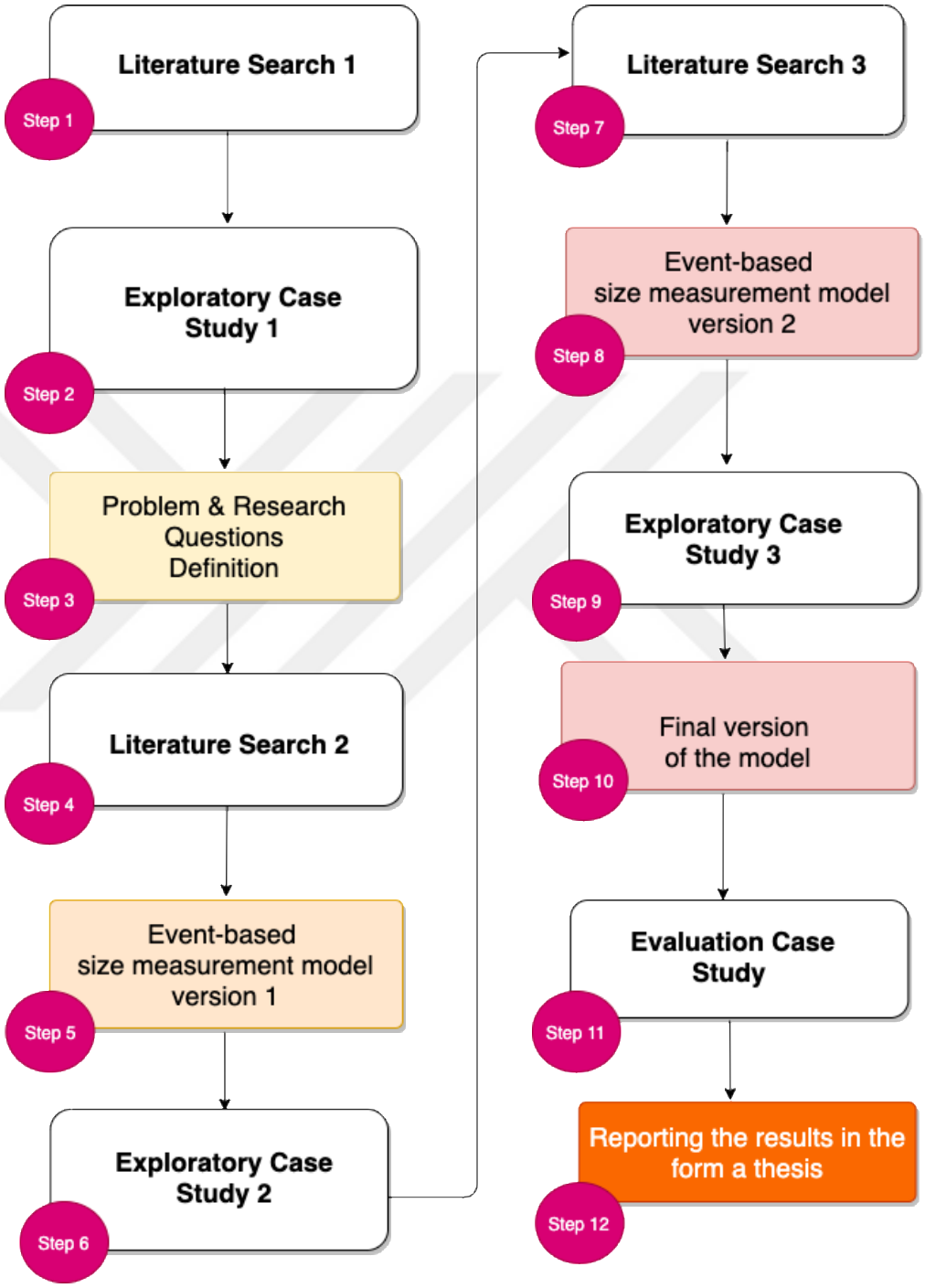


Figure 1 Research Roadmap

As it can be seen in Figure 1, the research roadmap is made up of twelve steps whose details are presented throughout this thesis. To give an idea for the reader about the rest of this thesis, the purposes of these steps and major findings are summarized as follows:

To answer RQ1, we conducted a systematic literature review and multiple exploratory case studies:

At Step 1, to diagnose the research problem, as a suggested practice in Design Science Research (Offermann et al., 2009), literature research is aimed to be conducted. For this aim, the research presented in this thesis was started with a systematic literature review (SLR) to identify existing software sizing approaches and the problematic issues reported in the literature concerning software size measurement in Agile software development. With this SLR, we seek an answer for two primary research questions: what types of size estimation/ measurement methods are being used in Agile software development and what challenges are existing related to size estimation/ measurement.

Within the scope of the SLR that is conducted considering the guidelines suggested by (B. Kitchenham & Charters, 2007); the research problem, related research questions, keywords and online databases in which search will be carried out were identified; inclusion and exclusion criteria guiding whether to add the retrieved article into the pool were defined; searches were made; duplicating articles were eliminated. The articles were scanned according to the title and abstract to find out the relevant articles; each included article's full text were read and analyzed and synthesized in order to answer the research questions. As a result of the SLR, different size measures and estimates used in Agile software development and challenges regarding the applicability of these size measures and estimates are identified, the related process is presented in Section 3.1 and can be seen in detail in (Hacaloğlu & Demirörs, 2018).

At Step 2, in the lights of the results obtained from the SLR conducted at Step 1, the Multiple Exploratory Case Study 1 (Yin, 2017) was conducted in three organizations' projects selected considering the case selection criteria, in order to experience these shortcomings first-hand by utilizing COSMIC Functional Size Measurement (FSM) method. In this way, we aimed to discover potential improvement opportunities and obtain an inspiration for research.

From this case study, we deduced following conclusions: available estimation related meeting takes significant time; measuring Agile projects with COSMIC FSM involves lots of assumptions; it is challenging to identify functional processes, data groups, and object of interests, and data movements. Although data analysis is a suggested practice in (COSMIC, 2017a) such a data modeling practice is not observed in the case organizations. Enhancing requirements with screens, and specifying requirements in the form of step-by-step user-system interaction facilitate the measurement. Based on the findings, the research problem: "how a size measurement model should be for contemporary software architectures?" and related research questions were formulated.

At Step 3, as a result of the Multiple Exploratory Case Study 1, the research problem focused in this thesis and its respective research questions were identified. The different characteristics of contemporary software, the demonstration of fading out status of classical data modeling practice among Agile practitioners, the cost of estimation activity,

the importance of adequate requirement specification for the accurate measurement are all identified within the scope of this case study.

To answer RQ2, we conducted literature search and exploratory case studies to understand how events can be used as a size measure and exploratory case studies

At Step 4, inspiring from the findings obtained at Step 3, the importance of transition from data-centric systems to event-based systems is realized. Accordingly, as a suggested practice in DSR by Offermann et al. (2009), as a part of solution design, an extensive literature search 2 were conducted in order to formulate a solution model. Within the scope of the literature search 2, the literature on size related studies for new generation software is investigated.

At Step 5, in the lights of the Exploratory Case Study 1, at Step 3 and literature search 2 conducted at Step 4, the first version of the solution model which is an "event-based size measurement model" were defined. In that model, the events occurring in a requirement statement are broadly divided into two categories such as: "user events" and "system events".

At Step 6, the first version of the model was applied within the scope of the Exploratory Case Study 2 with three cases chosen considering the case selection criteria. The projects' requirements have been modeled with extended Event-Driven Process Chain. The events have been identified. Besides, the functional sizes of the projects have been measured using COSMIC FSM method. Since, the effort of each requirement are available for the projects, the Pearson Correlation Analysis have been conducted to explore the relationship between the number of events and effort. In addition, the correlation values were compared with that of COSMIC FSM based sizes - COSMIC Function Points (CFP) and effort. As a result, it is observed that events correlates better with effort than CFP and event based approach for size measurement gives promising results.

At Step 7, by motivating from the findings at the previous step, with the aim of providing an objective, repeatable and clear definition of event abstraction; literature search 3 was conducted. As a result of this search, different event abstractions defined in the literature are identified.

At Step 8, the event-based size measurement model was updated to define events occurring within the system were classified based on their sources such as human user, hardware, system itself, and external systems.

At Step 9, another exploratory case study 3 was conducted to explore the applicability of the second version of the measurement model. eEPC was used to model the requirements. The events defined according to the second version of the model are counted. In addition, the functional size of the requirements is measured with COSMIC FSM method. Similar to the previous case study, correlation analysis was conducted between the event and effort and between the CFP and effort. It is found that there exists strong positive correlation between event and effort and this is higher than that of COSMIC FSM and effort.

Moreover, it is seen that events occurring as a result of an activity by the system correlating well with effort. This situation directed us to define system events as in the form of computation events.

At Step 10, we refined our model by clearly defining the events to be counted as *Computation Events*". These computation events are further defined in a clear manner as being *"Display event", "Calculation/ Processing event", "Record event", "Decision event", "Retrieval event"* and "*Communication event"* and "*System Boundary Event*". This classification has been done to facilitate the work of measurer while identifying the events hence to provide a repeatable and objective measurement. As a result, the final version of the model is defined.

<u>To answer RQ3, we conducted evaluation case studies in 2 organizations and derived effort prediction models:</u>

At Step 11, an evaluation case study was performed with projects from two large software organizations. The projects are measured according to the event-based measurement model. Pearson Correlation Analyses are conducted between events and effort and CFP and effort. To evaluate the success of the event-based size measurement, simple linear regression analyses are conducted and effort prediction models was formed. These models' accuracies are evaluated using accuracy indicators such as MRE, MMRE and PRED. It is seen that both event-based and CFP-based effort prediction models yielding acceptable results in projects of the first organization where event-based size outperforms CFP in terms of correlation with effort in projects in the second organization.

1.4. Structure of the thesis

The remaining parts of this thesis is organized as follows: Chapter 2 describes the methodology adopted. In Chapter 3, we present the problem identification activities such as the systematic literature review and Exploratory Case Study 1. In Chapter 4, solution design including related literature search, exploratory multiple case study and several versions of the event-based size measurement model that we proposed are explained. In Chapter 5, we describe the event-based size measurement model where in Chapter 6 presents the studies regarding the evaluation of it. In Chapter 7, limitations and validity threats are presented where Chapter 8 offers the conclusion and further research suggestions.

CHAPTER 2

METHODOLOGY

In this section, the research methodology that has been followed in this study is described. Software Engineering is an interdisciplinary field which comprises both technical and social aspects (Wohlin et al., 2012). In this context, Wohlin, Höst, & Henningsson (2003) draw attention to the necessity of incorporating empirical methods in software engineering research due to the human-intensive characteristic of software development. Empirical methods bring several advantages to software practitioners and academicians by being repeatable, by considering real world data and by providing scientific proofs (Malhotra, 2016). With these methods where statistical tests are carried out, the gap between the theory and the practice is decreased and consequently software development processes and procedures can be assessed and improved (Malhotra, 2016).

On the other hand, since these methods are brought into software engineering domain from different disciplines, there is a lack of consistent terminology and agreement on how to separate each method from the other one (Easterbrook, Singer, Storey, & Damian (2008). Consequently, in software engineering literature, there exists various classification of empirical methods. For example, Easterbrook et al. (2008) classified empirical research methods as controlled experiments, case studies, survey research, ethnographies and action research. Wohlin et al. (2003) divided empirical studies as being quantitative and qualitative and presented the research methods in software engineering as controlled experiments, case studies, surveys, post-mortem analysis. Similar to Wohlin et al. (2003), Malhotra (2016) categorized empirical studies as being quantitative and qualitative; and positioned experiment, survey research, systematic reviews, post-mortem analysis are under quantitative and case studies are under the qualitative categories.

Rabhi & Demirors (2018) in their White Paper study classified the research in software engineering into two major categories as empirical research and Design Science Research (DSR). Rabhi & Demirors (2018) further explained the type of research questions that these two major group of methodologies address. According to the authors, empirical research (Controlled experiments, Case Studies, Surveys, Action Research) aims to answer knowledge questions that can be of exploratory, base rate, and relationship type; and design science methodology answers design question such as “what is the most effective way to achieve X through an artifact” where this artifact can be a code, a process or a model (Rabhi & Demirors, 2018).

In addition, Rabhi & Demirors (2018) pointed out that although both methodologies can be combined in a research study; design science research methodology is superior but uses an empirical research methodology to assess the solution. The superiority of design science research is explained with its applicability on cases where the problem formulation requires the solution creation which is an iterative, flexible and exploratory process.

Consequently, in this thesis, in order to develop a novel software size measurement model, Design Science Research (DSR) is adopted. The human-made characteristics of software engineering artifacts make software engineering field suitable for design science research which aims to comprehend and ameliorate human-created design (Runeson, Engström, & Storey, 2020).

DSR is a paradigm elaborated by Hevner et al. (2004) for information systems and extended by Wieringa (2014) into software engineering (Runeson et al., 2020). Design Science paradigm is in the pursuit of "what is effective", by creating innovative artifacts to expand the edges of human and organizational capabilities (Hevner et al., 2004). Since the objective of this thesis is to create a design artifact in the form of a new software size measurement model conforming to the characteristics of new generation software architectures; DSR paradigm, by being proactive as regards the technology (Hevner et al., 2004) fits well as a research methodology. To demonstrate the appropriateness of DSR research to our thesis, we followed the guidelines suggested by Hevner et al. (2004). Then, we adopted and followed the research process suggested by (Offermann et al., 2009).

2.1.How DSR fits our research?

Hevner et al. (2004) describe Design Science Research (DSR) as being a problem solving paradigm whose result is an IT artifact addressing an important organizational problem and suggest seven guidelines for a DSR. In this thesis, we took these guidelines as reference in the development of a new software size measurement model and its evaluation as a design artifact.

2.1.1. Guideline 1: Design as an artifact

Hevner et al. (2004) specifies that the outcome of DSR in Information Systems to be an IT artifact that aims to solve a significant organizational problem. The corresponding design artifact in this thesis will be an "event-based software size measurement model" for new generation software architectures which is an unsolved significant problem having both technical and organizational aspects. The proliferation of new generation architectures poses both technical and organizational challenges from the software measurement perspective. In this context, we can exemplify the problem by referring to the study by Engel et al. (2018) who seek to identify challenges regarding microservices which can be an example of contemporary software architecture. Engel et al. (2018) points out that 'What size does a microservice have" are shown amongst the challenges related

to microservices. In addition, Abeysinghe (2016) points to a misconception about the size of a microservice. Kalske et al. (2017) who investigate challenging issues related to the transition from monolith to microservices, draw attention to the fact that this transition having both architectural and organizational challenges.

2.1.2. Guideline 2: Problem Relevance

This guideline suggests that developing a technology-based solution to a significant business problem is the aim of the DSR (Hevner et al., 2004). In this respect, it is important to explain the relevance of the problem that this thesis aims to solve. Based on a survey conducted among the software practitioners in Turkey by Garousi, Coşkunçay, Betin-Can, & Demirörs (2015) it is found that around half (54%) of 202 participants stated that they do not use any software size measurement method.

On the other hand, the research on software size measurement and effort estimation worldwide draw attention to the fact that subjective estimates are highly adopted in software engineering community (Commeyne et al., 2016; Hacaloğlu & Demirörs, 2018; Usman et al., 2014). However, using subjective estimates does not provide objective results, prevent the analysis of the productivity; thus complicating the comparisons across projects and organizations (Commeyne et al., 2016).

The importance of software size measurement and the significant capabilities it may provide for performance assessment, estimations and improvements are recognized in the community (Ozkan et al., 2008; Ugalde et al., 2020). However, the perceived difficulty regarding the usage of formal size measurement methods, specifically, the need of decomposing requirements into FSM related components drive practitioners to adopt informal subjective estimates (Hussain et al., 2013).

Also, software development is undergoing a paradigm shift where traditional architectures are giving their places to distributed modular architectures where database centric, transaction-oriented approach is being replaced with events. Nonetheless, to the best of our knowledge, there is no conventional approach to measure the size of these new types of systems involving a different philosophy rather than being data-centric. Hence, a novel software size measurement approach is needed to fulfill both needs.

Consequently, the artifact that is aimed to be created in this thesis addresses a software size measurement problem having both technological and organizational aspects. From the organizational aspect, the solution approach that this thesis aims to suggest can ease the estimation of effort, schedule and cost of software development projects using a repeatable and justifiable method.

2.1.3. Guideline 3: Design Evaluation

Hevner et al. (2004) specifies that the design artifact should be proven in terms of its utility, quality and efficacy by using well-performed evaluation methods; where this evaluation comprise the incorporation of the artifact into both technical and business environment.

In this respect, case study (Yin, 2017) method -that is also suggested by Hevner et al. (2004) as an observational evaluation method for DSR- is chosen for evaluating our model. Case study is an empirical research method examining a contemporary phenomenon rigorously within its real-world context, where the frontiers between the phenomenon and its context can be unclear (Yin, 2017). Case study is based on observation (Wohlin et al., 2003), providing researchers to have opportunities such as concentrating on a case in depth, holistically and by having a real-world perspective (Yin, 2017). Another advantage of case studies over experimental or survey type research is being able to explain real life situations including their complexities (Zainal, 2007).

Case study research is frequently applied in a wide set of disciplines including psychology, sociology, political science etc.; as well as practical domains such as healthcare, accounting and software engineering (Yin, 2017). Multi-disciplinary characteristic of software engineering field which combines the disciplines that case study is already used makes it naturally suitable for software engineering (Runeson & Höst, 2009). In addition, by helping to avoid scale-up problems case studies suit well for evaluating software engineering methods and tools in industrial settings (Wohlin et al., 2003). In this context, "a case" in software engineering can be a project, an organization, a process, a technology etc. (Runeson & Höst, 2009).

Case study can be carried out on any topic (Yin, 2017) and it increments the comprehension of the phenomenon under investigation (Malhotra, 2016). For this reason, it is possible to apply case study as a research method for exploratory purpose, for understanding and explaining a phenomenon or to build up a theory in both prospective or retrospective fashion (Perry, Sim, & Easterbrook, 2004). In addition, Wohlin et al. (2003) draw attention to carrying out case study as a comparative research strategy where it is possible to compare the results of more than one method.
Case studies are categorized under three categories as "exploratory", "descriptive" and "explanatory" case studies (Yin, 2017). Exploratory case studies are conducted to investigate a phenomenon in the data that act as a field of interest, descriptive case studies are carried out to describe the natural phenomenon that arise in the aforementioned data and explanatory case studies are conducted by shallow and deep investigation to explain the phenomenon in the data (Zainal, 2007).

Within the scope of this thesis, exploratory case study method is chosen to be applied. The reason behind this choice is that exploratory case studies provide initial exploration of phenomena to construct a theory and develop hypotheses (Easterbrook et al., 2008). Accordingly, since the aim in this thesis is to define a novel software size measurement

method for contemporary architectures, using exploratory case study, it is possible to investigate the strengths and weaknesses of available size measurement methods in contemporary software development projects to develop a hypothesis and a theory.

Besides, according to the method of application, case studies can be single or multiple (Yin, 2017). By providing replication opportunity, multiple case studies form a more powerful research design and are applied to strengthen the reliability of the research (Runeson, Host, Rainer, & Regnell, 2012; Yin, 2017; Zainal, 2007) and offers greater validity (Easterbrook et al., 2008). In addition, as stated in (Runeson et al., 2020) multiple case studies are suggested by Aken, (2004) as the representative research methodology in order to acquire design knowledge.

Yin (2017) describes the process of conducting case study research as linear but iterative process having six major steps which are plan, design, prepare, collect, analyze and share. Similarly, Malhotra (2016) describes typical phases in a case study as follows: case study design, data collection, execution of case study, data analysis and reporting. We plan to follow these stages in all the case studies conducted in this thesis.

As a conclusion, in this thesis, in order to both increase the repeatability of the research and make a stronger design "exploratory multiple case studies are planned with several projects from different organizations. In order to conduct multiple case studies, we adopted and adapted the multiple-case study process suggested by (Yin, 2017) and presented in Figure 2.

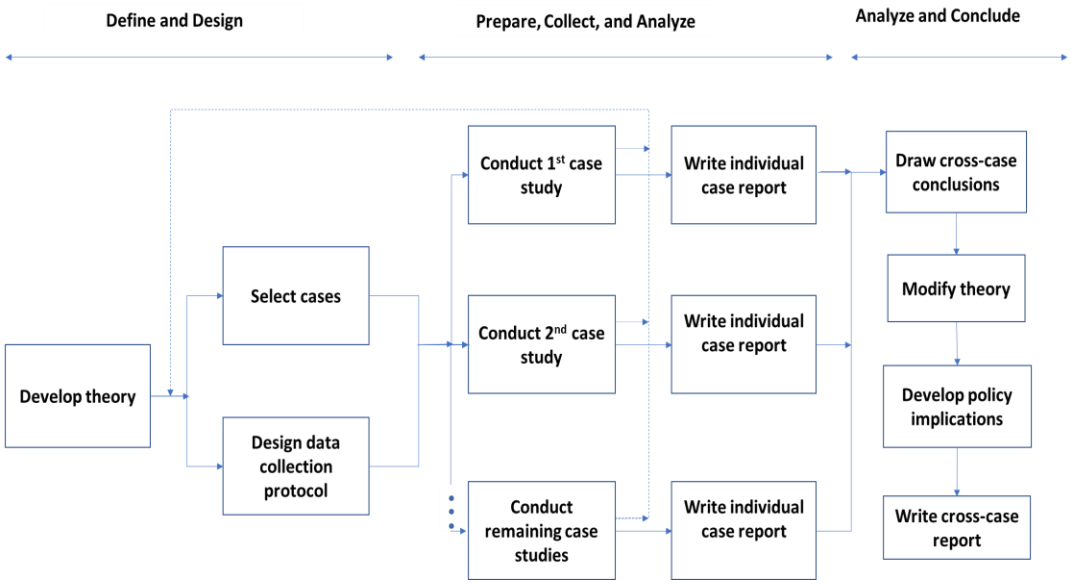


Figure 2 Multiple case study process by (Yin, 2017)

Within the scope of the Multiple Case studies;

- The problems mentioned in the literature can be assessed with first-hand experience. We explored the problems related to FSM in projects.
- The utility of the proposed model can be evaluated based on the amount of time / effort spent for the measurement, and ease of use.
- The efficacy of the model can be evaluated based on the accuracy of the effort estimation produced using the software size measured using the proposed model. Formal proof methods are used to validate the effectiveness of the model.
- The quality of the model can be evaluated based on the successful result frequencies produced as a result of multiple case studies. For this aim, the success of the proposed size measurement-based effort can be compared and contrasted with other artifacts such as story points or existing FSM methods such as COSMIC FSM.

As suggested by Runeson & Höst (2009) quantitative analysis techniques such as descriptive statistics, correlation analysis, development of predictive models, and hypothesis testing can be conducted and used for the evaluation. Details regarding these activities are given in Section 2.2.3.

2.1.4. Guideline 4: Research Contribution

As stated by Hevner et al. (2004), the artifact itself is the contribution of DSR. Accordingly, in this thesis, a novel size measurement model is the research contribution. With this model, we aim to contribute to the software engineering research domain and the way the practitioners' do their business in the form foundation- a novel size measurement model. More specifically, we aim to suggest a software size measurement model through which we define the size of the software in terms of events. The proposed model can contribute to the software measurement and estimation literature by providing a novel size measurement method based on events; and by creating effort estimation models grounded on the developed size measurement model.

2.1.5. Guideline 5: Research Rigor

In this guideline, Hevner et al. (2004) emphasize that accurate methods are needed to be employed to develop and evaluate the artifact. We adopted systematic literature review technique (B. Kitchenham & Charters, 2007), case study research (Yin, 2017) and related data analysis techniques such as Correlation and Regression to strengthen the research rigor. SLR studies have become popular in Software Engineering domain since 2004 as an important research methodology (Babar & Zhang, 2009). With the help of systematic literature reviews, all relevant studies in the field in specified time interval are compiled systematically. In this way, the problems mentioned in the relevant literature can be discussed in depth. Reviewing the literature in a systematic way differs from unsystematically reviewing the literature by following set of steps which are well and strictly defined according to a protocol which is developed in advance (Biolchini, Mian,

Natali, & Travassos, 2005). Similarly, case studies are shown as an empirical research method suitable for software engineering (Runeson & Höst, 2009). To conduct all the case studies in this thesis, we aim to follow the steps such as "Case Study Design", "Data Collection", "Execution of Case Study", "Data Analysis" and Reporting (Malhotra, 2016).

In empirical research studies, data analysis are carried out by considering the type of the dependent variable which can be continuous or binary (Malhotra, 2016). For binary type of dependent variables statistical methods and machine learning methods; and for continuous type of dependent variables machine learning and statistical methods are suggested (Malhotra, 2016). Since the dependent variable in this study is the actual effort spent to realize the project- which is a continuous variable- we prefer to apply two statistical techniques such as Correlation Analysis and Regression Analysis for the empirical evaluation of the event-based size measurement model.

Correlation analysis investigates whether there exists a relationship between two variables by looking at its correlation coefficient which can take values between the range of +1 to -1. +1 signifies that there exists strong positive correlation between two variables, i.e., if the value of one variable increases so does the other one while -1 indicates the reverse behavior. (Field, 2013; Schober, Boer, & Schwarte, 2018). Within the scope of this thesis, Pearson Correlation Analysis will be employed to identify whether there is a relationship between the effort spent in the projects and the components of the proposed size measurement model.

If high correlation can obtained between two variables, regression analyses can be applied (Humphrey, 1995). Regression analysis is a technique which explore the relationship between variables (Sykes, 1993). Regression-based models are employed in the literature with the aim of data description, parameter estimation, prediction and control (Montgomery, Peck, & Vining, 2012). In the literature, several types of regression analysis methods exist. Among these, simple linear regression analysis, multiple regression analysis, and non-linear were referred for effort estimation purposes in software engineering (Salmanoglu et al., 2017; Ungan et al., 2014) where estimations can be carried in two ways: it can be based on expert judgment or mathematical model where regression analysis is situated under this category (Abran, 2015). In this thesis, we will refer to regression analysis to estimate the effort spent based on the size measured using the model proposed.

When there is single explanatory variable, the type of regression is called simple regression and when the number of explanatory variable is multiple than the type of regression is called multiple regression (Sykes, 1993). Considering the aim of using regression analysis in this thesis study, which is to predict the effort spent by using the measurement model proposed, the type of regression will be decided based on the number of measurement components that will be defined in the measurement model proposed. In other words, if there will be a single size measurement component defined, then simple regression will be applied. However, if the size measurement model will be made up of more than one component, then, multiple regression analysis can be employed.

To manage and control a software project, it's crucial to estimate development effort accurately (Shepperd & Schofield, 1997). For this reason, after creating the estimation models, the accuracy of the predictions will be evaluated by using the most commonly used accuracy metrics (Port & Korte, 2008): “Mean Magnitude of Relative Error” (MMRE) and “Percentage Relative Error Deviation within x” (PRED(x)). Magnitude of Relative Error (MRE) constitutes the base of the both of these measures (Port & Korte, 2008). These measures are defined as follows (Idri, Abnane, & Abran, 2018) :

$$MRE \left| \frac{Effort_{actual} - Effort_{estimated}}{Effort_{actual}} \right| \times 100 \quad (1)$$

Mean Magnitude of Relative Error (MMRE) is the mean of MRE values, where n corresponds to the number of projects.

$$MMRE = \frac{1}{n} \sum_{i-1}^{n} MRE_i \qquad (2)$$

Despite its frequent usage, MMRE was criticized by several researchers (Foss, Stensrud, Kitchenham, & Myrtveit, 2003; B. A. Kitchenham, Pickard, MacDonell, & Shepperd, 2001; Myrtveit, Stensrud, & Shepperd, 2005). Jorgensen (1995) emphasized that MMRE is sensitive to high MRE- values and suggested to use Median Magnitude of Relative Error (MdMRE) as an alternative to MMRE, because MdMRE is a metric which is less influenced by extreme values (Jeffery, Ruhe, & Wieczorek, 2000).

Percentage Relative Error Deviation within x” (PRED(x)) is defined with formula (3) in which, “n” denotes the total number of observations and “k” denotes the number of observations having MRE value less than or equal to “x” (Idri et al., 2018).

$$Pred(x) = \frac{k}{n} \qquad (3)$$

When comparing several software development effort estimation techniques, it is suggested to prefer the technique having the lowest MMRE and/or the highest PRED(x) value (Idri et al., 2018). For these metrics, several threshold values were suggested in the literature. For example, According to (Conte, Dunsmore, & Shen, 1986), an MMRE and MdMRE value less than or equal to 0,25 can be considered as an acceptable threshold value for effort estimation models. On the other hand, (Hastings & Sajeev, 2001) points to consider the MMRE value greater than 0,50 unacceptable; between 0,20 and 0,50 acceptable and less than 0,20 as predictive. Regarding PRED values, PRED (0.25) and PRED (0.30) have been mentioned as the most used PRED values in the literature (Van Koten & Gray, 2006). MacDonell (1997) by referring to the study by Tate and Verner (Tate, Verner, & Veryard, 1991) who pointed out that PRED(30)>=70% would be a more realistic accuracy threshold. In addition, there exists a point of view by MacDonell & Gray (1998) who indicated that model having a prediction accuracy with PRED (30)= 60% can be considered as good model.

2.1.6. Guideline 6: Design as a Search Process

Hevner et al. (2004) describes the design science as an iterative search process to find out a way to a problem. As the research process, we adopt and adapt the process suggested by Offermann et al. (2009). The process suggested by Offermann et al. (2009) is composed of three main stages: problem identification, solution design and evaluation. The process will be explained sub-section 2.3 in details.

2.1.7. Guideline 7: Communication of Research

In this guideline Hevner et al. (2004) refer to the importance of transferring the DSR results to both technological and managerial audiences. Since the model proposed within the scope of this study will be a PhD dissertation, the thesis will be publicly available for both academic and industrial audiences. In addition, different phases of the study will be presented in the academic conferences where both practitioners and academicians meet. In addition, Hevner et al. (2004) discusses the importance of the communication of the research in terms of repeatability from the academic audiences and the adoption related decisions from the management audience perspective. With this research we satisfy the repeatability requirement by presenting step-by-step application of the DSR process in this thesis and we give clues for the adoption related decision by applying case studies in real software organizations and evaluating the results with well-established techniques.

2.2. Applying the Design Science Research Process

The guidelines suggested by Hevner et al. (2004) provide very useful knowledge about what a DSR possesses; but they do not define a process about how to conduct this research. Runeson et al. (2020) supports this view by stating that design science does not specify a particular process to carry out a research study. Offermann et al. (2009) also points to the lack of a clear approach on how to integrate different research methods for DSR and describe their process as being a roadmap for this purpose. As an alternative, Offermann et al. (2009) proposed a detailed research process for DSR, by combining qualitative and quantitative methods in a formal way; which we adopted and tailored according to the characteristic of our research. Consequently, by conforming to the guidelines suggested by Hevner et al. (2004) and by following the process suggested by Offermann et al. (2009), our research process adopts a methodology which is a synthesis of both studies.

The process suggested by Offermann et al. (2009) comprises three phases: “Problem identification”, “Solution Design” and “Evaluation” and has an iterative nature. Similarly, in a recent study, Runeson et al. (2020) points that design science covers problem conceptualization, solution design and validation. A large number of research methods are included in design science, where empirical research methods are adopted especially for problem conceptualization and validation among which natural configurations are the preferred ones (Runeson et al., 2020).

To ease the readability, the mapping between our research roadmap given in Figure 1 and the steps suggested by Offermann et al. (2009) is combined and presented in Figure 3.

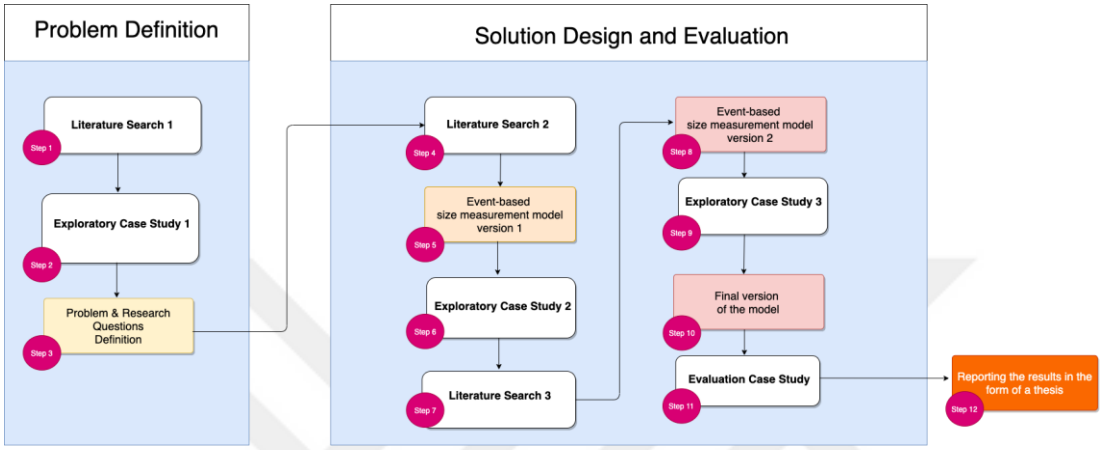


Figure 3 Research Activities adapted from (Offermann et al., 2009)

2.2.1. Problem Identification

According to Offermann et al. (2009), this phase requires sub-stages such as problem identification, literature review, expert interviews and pre-evaluation of the problem relevance. Wohlin et al. (2003) point out the impossibility of starting the improvement directly and draw attention to the importance of understanding the current situation to determine the improvement opportunities.

Therefore, firstly, we planned a systematic literature review study (B. Kitchenham & Charters, 2007) on Agile software development domain where we identified the problems stated in Chapter 3. As a result of reviewing the literature, we identified the problem on the applicability of FSM in Agile projects, the study is presented in section 3.1and can also be seen in detail in (Hacaloğlu & Demirörs, 2018). Then, as the second step of the problem identification stage, to explore the root-causes of this problem at first-hand, we conducted first exploratory multiple case studies on projects of three organizations which were selected based on the case selection criteria. The exploratory case study is presented in section 3.2 whose details can be accessed at (Hacaloglu & Demirors, 2019).

As a result of this phase, we identified that contemporary software is no more displaying a data-centric characteristic; it may involve different architecture, and may be composed of different services and structures. New generation architectures, for example microservices, are separated from old style of architectures such as monoliths and SOA by focusing more on scalability, independence and semantic cohesiveness (Mazzara et al., 2020). Dragoni et al. (2017) emphasize the difference of these new architectures such as microservices being independent and deployed in isolation; from the monoliths which use

the resources of the same machine. In other words, where we cannot mention a single data store in these distributed systems. As a result of this situation, it becomes inadequate and therefore, challenging to approach to the size measurement from the data-centric perspective by using data-centric functional size measurement methods. Instead, the behavioral aspect of the software becomes more prominent.

In this respect, when we observe contemporary architectures, we notice that events become important concept in this technological shift. Today's systems adopt event-driven architecture in which applications can communicate asynchronously or integrate with other systems by using events (Parulkar, 2020b). Event-driven programming are being adopted by world's leading companies such as eBay, PayPal and LinkedIn (Zhu et al., 2015).

Migration to event-driven architecture corresponds to shift from data-centric structure to event-centric fashion (TIBCO, n.d.). Michelle (2013) distinguishes event from entity data as follows: while entity is giving information regarding the present state of the application, event illustrates the actions which are occurred in the application. The way event-driven application is operating is also different from the traditional request-driven structure. Events such as user behavior, sensors, messages from programs dictates the flow of the program in an event-driven application (Chima, 2018). The flow is arranged in a way to execute action as a result of an event (TIBCO, n.d.). According to Boner (2017) modeling events is modeling the behavior of the system instead of its structure and one need to focus on what happens (verb-events) rather than things (noun-domain objects).

2.2.2. Solution Design

To design the solution, Offermann et al. (2009) suggest two steps; such as artifact design and a literature search that support it. The design artifact of this thesis is the measurement model based on events, which is conceptualized as a result of the problem identification phase. To create a first version of the model, the first decision to make was the event identification. For this aim, the authors made extensive research -which is a suggested practice by Offermann et al. (2009)- on "what is an event?" and "what are the possible events?" stated in the literature. In addition, the literature searches also comprise, existing methods for the event modeling and how to model them. All related information concerning the literature search are presented in section 4.1 and 4.3.
After having synthesized the event related information as a result of the literature search, the measurement model has been started to be formed. As it is presented by Offermann et al. (2009) in Chapter 2, Hevner et al. (2004) draw attention to the iterative and incremental nature of the design process and specifies that the outcome of the design evaluation is a feedback for the construction. Therefore, the proposed solution went through several update process. Two multiple exploratory case studies are conducted to assess its applicability where the measurement model is improved based on the findings of these case studies. The activities conducted in this stage is presented in Chapter 1.

2.2.3. Evaluation

As a typical evaluation method for design science research process Offermann et al. (2009) suggest to apply either expert judgment, lab experiment or case study / action research. Among these, for our study, the most appropriate approach is the case study which is an empirical method whose objective is to explore contemporary cases in their context (Runeson & Höst, 2009). The details regarding case study is previously explained in Section 2.1.3. In this evaluation step, the usefulness and feasibility of the proposed software size measurement model are assessed within the scope of multiple case studies. The multiple case studies are used to examine the success of the model that is developed in the "Solution Design" phase and to identify possible improvement opportunities.

For quantitative and qualitative data, different analysis techniques are proposed; for quantitative data, descriptive statistics, correlation analysis, development of predictive models, and hypothesis testing are suggested in the case study research (Runeson & Höst, 2009). Therefore, to evaluate the success of the model statistical analysis methods are employed. In order to identify whether a relationship exists between the effort spent in a project and the components of the measurement methods which is made up of units for contemporary software Pearson Correlation Analyses are conducted. To determine how successful is the effort prediction model constructed using the proposed method Regression analysis are used. Correlation analysis shows direction and the magnitude of the relationship between two variables where Regression analysis shows whether there exists a cause-effect relationship. Within the scope of this thesis study, Regression analysis is used to determine how successful is the proposed measurement method to explain the actual effort spent in a project.

In this perspective, effort spent in a project will be the dependent variable where the proposed measurement unit will be the independent variable. When there exists single independent variable Simple Linear Regression and when there exists more than one independent variable Multiple Regression Analysis are conducted. The accuracies of the obtained effort estimation models are tested using two metrics such as MMRE, MdMRE, and PRED (n). As explained in section 2.1.5, Mean Magnitude of Relative Error is used to determine the deviation between actual and estimated effort values.

2.3. Summary of the Chapter

This chapter started with an overview of research approaches in software engineering. Then the research methodology adopted in this thesis that is Design Science Research is described. The seven guidelines suggested by Hevner et al. (2004) are discussed within the scope of the aim of this thesis. Then, we presented how DSR is applied in this thesis. For this aim, we adopted the steps suggested by Offermann et al. (2009) as the research process. As recommended by Offermann et al. (2009) the research process involves three major steps, namely, problem identification, solution design and evaluation. In the

chapter, all the activities planned during these steps are described. In the next chapter, the activities carried out in problem identification phase are explained in details.

CHAPTER 3

PROBLEM IDENTIFICATION

3.1. Literature Search I: Systematic Literature Review

As mentioned previously in order to explore the research problem, we carried out a Systematic Literature Review (SLR) study. SLR presented in this thesis is conducted according to the approach suggested by Kitchenham & Charters (2007). The aim of this SLR study is to present the state of the art regarding size measurement practices existing in Agile community and more particularly to identify and exhibit related challenges that are articulated in the literature. As suggested by Kitchenham (2004) SLR has three major phases which are plan, conduct and reporting the review.

We present the SLR study in the form of these phases as in subsequent sections. This SLR study has been carried out in 2017 and published as a conference paper in the International Workshop on Software Measurement and the International Conference on Software Process and Product Measurement (IWSM-MENSURA) in 2018. A comprehensive version regarding this research can be accessed from the conference proceeding (Hacaloğlu & Demirörs, 2018).

3.1.1. Review Planning

During the planning phase, the formulation of the research problem has been studied. It is a known fact that due to focusing on quick software production, Agile practitioners consider measurement as non-value-added task. They usually adopt subjective estimates (Usman et al., 2014). However, although these subjective estimation methods fulfill some needs in Agile software development; they fail to answer the needs such as utilization of prior estimations, assessing their quality and improving their accuracy (Commeyne et al., 2016). Therefore, how to manage the size measurement and effort estimation in Agile remain still as a challenge. Consequently, this SLR study is carried out in order to reveal an understanding about how software size is used and perceived in Agile software development, to understand which challenges exist in size estimation and measurement, to identify which size estimates/ measures are available, and to present the state-of-the art on size measurement / estimation in Agile software development.

Upon the identification of the research problem, the study process is continued with following major steps: determination of the research questions, defining the search process (identifying online databases, keywords and search protocols) in order to be able to carry out the searches.

3.1.1.1.Research Questions

With this SLR study we aimed to address two primary research questions (RQs):

RQ1. What size measurement/estimation methods are utilized in Agile software development?
RQ2. What are the challenges in Agile size estimation and measurement?

With RQ1, having the awareness regarding the benefits software size measurement bringing in traditional software development (Abran et al., 2016; Gencel & Demirors, 2008); we aimed to identify the methods which address either the measurement or the estimation of the software size experienced in the literature of Agile software development.

With RQ2, by observing the Agile practitioners adopting expert based estimation techniques, we aimed to identify what type of situations are bounded to software size specifically in Agile software development ecosystem creating challenge during the software development lifecycle.

3.1.1.2.Search Protocol

Scopus[1], IEEE Xplore[2], Web of Science[3], ScienceDirect[4] and ACM Digital Library[5] are chosen as online databases in which the search will be conducted. Then, keywords of the search are identified. The keywords are specified by looking at the previously conducted SLRs in the literature, for example (Usman et al., 2014) and by taking the domain knowledge into consideration. Accordingly, we obtained four set of keywords as follows:

Set 1= {agile OR XP OR “extreme programming” OR Scrum}
Set 2= {size AND {estimate* OR measur* OR metric}}
Set 3= {COSMIC OR IFPUG OR “function point” OR “functional size”}
Set 4= {“story point” OR “use case point” OR “line of code” OR “LOC” OR “object points”}

[1] https://www.scopus.com/search/form.uri?display=basic
[2] https://ieeexplore.ieee.org/Xplore/home.jsp
[3] http://www.webknowledge.com
[4] https://www.sciencedirect.com/
[5] https://dl.acm.org/

3.1.2. Conduct of the searches

We conducted manual search from each of the online databases specified in the previous paragraph. During the search process, keyword Set 1 is aggregated with the other sets using AND clause. In this way, we reached at a total of 2,581 articles. The distribution of articles per databases are given in Table 1.

Table 1 Distribution of Articles per Database (Hacaloğlu & Demirörs, 2018)

Database	**Number of articles retrieved**
Scopus	944
IEEE Xplore	580
ScienceDirect	434
ACM	458
Web of Science	165

Since an article can be appearing in more than one database, duplicates were common in the search results. After removing the duplicating articles, we scanned the title and abstract of each article and eliminated irrelevant studies by considering the inclusion and exclusion criteria we set prior.

3.1.2.1.Inclusion and Exclusion Criteria

We identified several inclusion and exclusion criteria for the articles to consider for the review. To include an article in our pool, we set following criteria: the article should be presenting a study related to Agile software development, in which specifically software size is used (for example, for estimation related purposes), improved or compared. In addition, the studies that are written in English, published as a conference/ workshop proceeding or journal article whose full-text can be accessed are included in our study.
On the other hand, there were some reasons why we excluded the articles we retrieved from the databases. The articles in which software size is not used during Agile software development, whose full text are inaccessible and which are not written in English. In addition, we incorporated a backward and forward snowballing process (Wohlin, 2014) similar to (Garousi, Petersen, & Ozkan, 2016) in order to be able to include more relevant articles in our pool.

3.1.3. Reporting the results

As a result of this SLR study, we obtained 40 articles. Regarding RQ1, in which we investigated what size measurement/estimation methods are being utilized in Agile projects, we obtained that story points are used as the most frequent size estimate in Agile, where COSMIC Function Points (CFP) follows it by being the second one. The type of size measures/ estimates observed as result of this SLR study and the articles are presented in Table 2. In Table 2, in the first column the measure/ estimate types and in the second column, the articles studying these measures/ estimates are presented. Figure 4 visualizes the distribution of the articles with respect to size measures and estimates.

Table 2 Size measures/ estimates and articles (Hacaloğlu & Demirörs, 2018)

Main Size measure/ estimate type	Respective article
Simplified Function Points (SiFP) (Lavazza & Meli, 2014)	(Lenarduzzi & Taibi, 2014), (Lenarduzzi, Lunesu, Matta, & Taibi, 2015)
NESMA Function Points (NESMA, 2004)	(Huijgens & Solingen, 2014)
Function points (Albrecht & Gaffney, 1983)	(Kang, Choi, & Baik, 2010), (Soni & Kohli, 2017)
IFPUG Function Points (IFPUG, 2009)	(Santana, Leoneo, Vasconcelos, & Gusmão, 2011), (Lenarduzzi et al., 2015)
COSMIC Function Point (CFP) (COSMIC, 2017b)	(Buglione & Trudel, 2010), (J.-M. Desharnais, Kocaturk, & Abran, 2011), (J.-M. Desharnais, Buglione, & Kocatürk, 2011), (J. Desharnais, Buglione, & Kocaturk, 2011), (Hussain et al., 2013), (Ungan et al., 2014), (Ochodek, 2016), (Salmanoglu et al., 2017), (Dumas-Monette & Trudel, 2014), (Berardi & Santillo, 2010), (Fehlmann & Santillo, 2010)
Story Points (SP)	(Power, 2011), (Santana et al., 2011), (Hamouda, 2014), (Huijgens & Solingen, 2014), (van Valkenhoef, Tervonen, de Brock, & Postmus, 2011), (Bhalerao & Ingle, 2009), (Miranda, Bourque, & Abran, 2009), (Kang et al., 2010), (Kumar, Kumari, & Perumal, 2014), (Aslam, Ijaz, Lali, & Mehmood, 2017), (Popli & Chauhan, 2014b), (Popli & Chauhan, 2014a), (Popli & Chauhan, 2014c), (Popli & Chauahn, 2015), (Choudhari & Suman, 2012b), (Choudhari & Suman, 2012a), (Zahraoui & Idrissi, 2015), (Bhalerao & Ingle, 2011), (Satapathy & Rath, 2017)
Use Case Points	(Khatri, Malhotra, & Johri, 2016), (Nunes, 2009), (Parvez, 2013)
Actual times	(Hohman, 2005)
Web Objects (Reifer, 2000)	(Čelar, Šeremet, Marušić, & Turić, 2013), (Soni & Kohli, 2017)
User point	(Ali, Shaikh, & Ali, 2015)

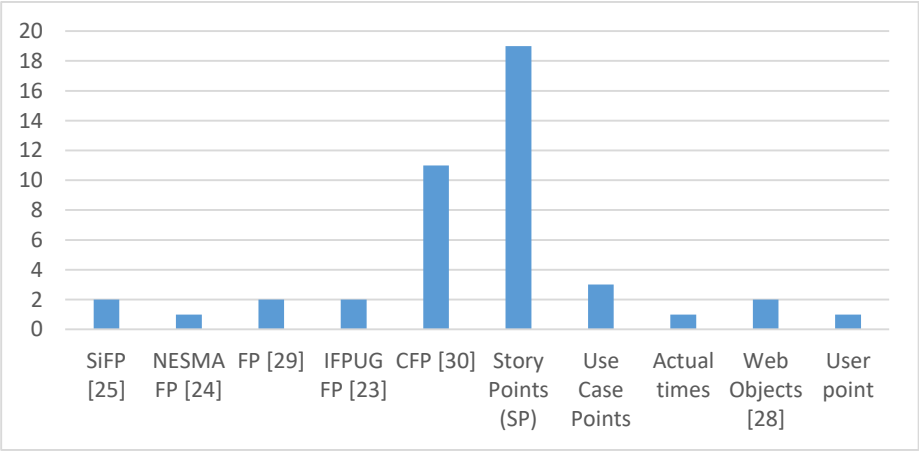


Figure 4 Size measures/ estimates types

Accordingly, a typical conclusion from this finding is that functional size measures such as SiFP (Lavazza & Meli, 2014), NESMA FP (NESMA, 2004) , FP (Albrecht & Gaffney, 1983), IFPUG (IFPUG, 2009), CFP (COSMIC, 2017b) are being applied in Agile software development projects. This behavior indicates that there exists an awareness on the applicability of size measurement in Agile community. However, other estimates which have not been developed initially for "size" measurement purposes are also very common in Agile software development ecosystem.

Regarding RQ2, in which we focused on identifying the problematic issues concerning the size measurement in Agile software development; we observed three categories of challenges: "misinterpretation of size measure", "difficulties in application", and "measurement/estimation process acceptability". In the lights of these findings, to be able to discuss the problem in depth, we explored additional articles in the literature. (Detailed discussion regarding the different interpretation of software size can be found in our article (Hacaloğlu & Demirörs, 2018)).

With respect to the misinterpretation of size concept, it is observed that, authors have different perception about software size and utilized it accordingly. For example, in a study by Popli & Chauahn (2015) the size of the software is quantified using number of days. In addition, although story point is not a size measure, there exists authors which relate it with the size. For example, in following studies (Aslam et al., 2017; Bhalerao & Ingle, 2009, 2011; Choudhari & Suman, 2012a, 2012b; Hamouda, 2014; Kang et al., 2010; Kumar et al., 2014; Miranda et al., 2009; Power, 2011; Santana et al., 2011; van Valkenhoef et al., 2011) story points denote the size of the software.

However, in study by Hohman (2005), it denotes the complexity. In this respect, size and effort related misconception has been an already stated problem previously by Özkan & Demirors (2016) and it is also stated in Agile guideline of COSMIC (COSMIC, 2011). The same problem is observed in the present SLR too. Although there exists studies for example (Satapathy, Panda, & Rath, 2014) indicating that story point is a measure of effort, there are studies considering story points as a measure of size (Bhalerao & Ingle, 2011; Power, 2011; Santana et al., 2011) which is not the case in reality. In addition, duration is also confused with size, for example, in the study by (Popli & Chauahn, 2015), we observe the size of a user story being defined with number of days. From these findings, it can be understood that there is a misconception on the meaning of software size, and the story point. To overcome this problem, the relationship regarding the size and other related concepts such as effort, duration and complexity must be built. With this way a clearer consensus on these concepts can be obtained and specific importance of size among these concepts and its objective measurement can be distinguished better.

Regarding second challenge category that is the difficulty in the application, we observed researchers complaining following deficiencies of story points; Relativity (Huijgens & Solingen, 2014; Kang et al., 2010; Popli & Chauhan, 2014a; Power, 2011; Soni & Kohli, 2017), subjectivity (Berardi & Santillo, 2010; Huijgens & Solingen, 2014), the way they are assigned with respect to the team members perception at the time estimation is

conducted (Berardi & Santillo, 2010; Buglione & Trudel, 2010; Hamouda, 2014), the characteristic of not being transferrable to the outside of the team (Berardi & Santillo, 2010), the way they are assigned by using a comparison with either the simplest (Kang et al., 2010; Popli & Chauhan, 2014a) or the average story (Buglione & Trudel, 2010), the way they are assigned as high values for non-functional requirements (Buglione & Trudel, 2010), the characteristic of being either under or overestimated (Popli & Chauhan, 2014c), and the problems occurring due to the uncertainty in requirements (Popli & Chauhan, 2014c).

Concerning these deficiencies related to story points, an implication is that although story points are highly adopted estimates in Agile software development, they do not perfectly fulfill the needs of practitioners. The way they are assigned into stories, being dependent of both the estimators' subjective judgment and being based on the stories available at the time estimation are made specifically causing them to having relative values which can also change when the referenced stories change.

Alternatively, as it can be seen in the Table 2 and Figure 4; there exists attempts on applying FSM in Agile projects. For example, in (J.-M. Desharnais, Buglione, et al., 2011; J.-M. Desharnais, Kocaturk, et al., 2011; J. Desharnais et al., 2011; Salmanoglu et al., 2017; Ungan et al., 2014) COSMIC FSM method is adopted. These studies have been converged on the empirical analysis of the applicability of existing FSM methods in various Agile projects. In this respect, Javdani et al. (2013) draw attention to the fact that Agile software measurement suffer from a lack of an accepted and standardized approach.

When studies focusing on the utilization of FSM in Agile are observed, it is seen that FSM possessing challenges regarding their applicability, for instance, researchers are complaining about the difficulty in applying FSM (Huijgens & Solingen, 2014), there exists a perception that FSM is for a complete projects or part of projects (Lenarduzzi et al., 2015). In addition, the lack of model based (Berardi & Santillo, 2010) and mathematical based (Hamouda, 2014) estimations are articulated as challenges in Agile software development.

The last category on this issue is the acceptability of measurement/ estimation process in Agile software development. Regarding this challenge, following arguments are stated by the researchers: FSM is difficult and adjusting requirements to apply it is challenging (Huijgens, Bruntink, van Deursen, van der Storm, & Vogelezang, 2016; Hussain et al., 2013) and Agile teams want to be independent and do not prefer to feel a pressure on using a formal method (Hohman, 2005). Regarding story points, the assignment process is criticized for being costly (Ali et al., 2015), occupying long time (Salmanoglu et al., 2017), and influenced by the dominant characters (Power, 2011) and political pressure (Popli & Chauhan, 2014b).

As a conclusion, this SLR study helped in determination of the problem and motivated us to conduct a case study regarding the applicability of the FSM in Agile projects. At this point, Wohlin et al. (2003) remark the impossibility of starting the improvement directly

and draw attention to the importance of understanding the current situation to determine the improvement opportunities. Accordingly, to explore the root-causes of this problem at first-hand, we conducted first exploratory multiple case studies on projects of three organizations. The exploratory case study (Hacaloglu & Demirors, 2019) is presented in Section 3.2.

3.2. Exploratory Case Study 1

In this section, the details regarding Exploratory Case Study 1 is presented. The idea of conducting this exploratory multiple case study has been emerged as a result of our previous systematic literature review study (Hacaloğlu & Demirörs, 2018) which is presented in Section 3.1. From the systematic literature review study, we concluded that Agile practitioners are more inclined towards subjective estimates even though these estimates received a lot of criticism. In addition, as another result, it is observed that there exist views in the literature reflecting that it is challenging to apply FSM in Agile requirements. These findings formed the foundations of this exploratory multiple case study in which we aspire to observe to what extent FSM is applicable in Agile project, and to identify situations complicating the measurement at first hand.

In accordance with this purpose, we gathered a total of six projects which are developed using Agile methods from three organizations. We applied COSMIC FSM method to measure their functional size. By means of this case study, whose application process will be summarized in following paragraphs, we also took a snapshot of how these organizations perform the measurement/ estimation activities. This study has been published as a conference paper in 45th Euromicro Conference on Software Engineering and Advanced Applications (SEEA) in 2019. A comprehensive version regarding this research can be accessed from (Hacaloglu & Demirors, 2019).

3.2.1. Case Study Design

With the objective of assessing the applicability of functional size measurement in software projects in which Agile methodologies are adopted, we defined following research questions:

> RQ1. "What types of problems are encountered during the application of COSMIC FSM process in Agile projects?"
> RQ2. "What measurement components are remained unidentified, thus affecting the accuracy of the measurement?"

Then, we aimed to define an initial case selection protocol and decided to conduct this exploratory multiple case study by considering following case selection criteria; in organization or projects operating in different business domains, being of different types such as business application, distributed systems etc., operating in different platforms such as desktop, web or mobile and developed using Agile methodologies. To strengthen the

research, we identified another target which is including projects that are documented different requirements specification formats used in Agile projects such as user stories, use cases and any other style; and also agility levels of projects such as (Özcan-Top & Demirors, 2019).

These were defined as initial criteria to include a project within the scope of our study. However, not all of them was feasible to provide due to the challenge of carrying out a research in the industrial setting as stated by Malhotra (2016). Further detail regarding these challenges can be found at Chapter 7 where limitations are presented.

As the functional size measurement method, COSMIC FSM (COSMIC, 2017b) is chosen. The motivation behind this choice is that COSMIC FSM is a second generation functional size measurement method which is ISO compliant (ISO/IEC 19761:2017, 2017) and easy to use. Before the execution of the case study, the measurer has studied COSMIC FSM's manual (COSMIC, 2017b), guidelines (Berardi et al., 2011; COSMIC, 2017a) and exemplar case studies (COSMIC, 2015, 2018). It is important to emphasize that in order to assure the objectivity of the measurement and to provide consistency in the assumptions required during the measurement process, it is planned that the measurement to be held by an external measurer.

3.2.2. Description of the Cases

In this section, the cases, in other words, selected projects and their corresponding organizations are described. Organization 1 has a software development team of five people. The organization is developing both web and mobile applications. Scrum and Kanban are adopted in project development. The team documents the requirements in the form of user stories. They try to make and record estimations which were based on duration earlier, and based on story points currently. From the organization's issue tracking repository four completed projects that the estimator is familiar with their domain are selected.

Organization 2 works as an information technology and consulting firm adopting Scrum as the Agile methodology in their projects. They document the requirements in the form of user stories. However, they elaborate them into a form in which the interaction of user with the system are becoming atomic. The details include normal flow, acceptance criteria, pre-post conditions. They enhance the understandability of requirements with mock-up screens. Similar to the Organization 1, Organization 2 makes use of story points for the estimation purposes. Two sprints belonging to a web application has been collected from this organization.

Unlike Organization 1 and 2 which are from Turkey, Organization 3 operates in Australia. A business web application from the finance domain has been acquired from Organization 3. Different from the previous organizations, requirements are documented in the form

use-cases in a fully-dressed format. This documentation includes the actor, use case pre-requisite chain, use case description, main and exceptional flows, and post conditions. In addition, screens explaining each use cases are available in the project document set.

3.2.3. Execution of the Exploratory Multiple Case Study

During the case study execution, we followed the procedures suggested by related manuals (COSMIC, 2017b) and supporting guidelines (Berardi et al., 2011; COSMIC, 2017a). According to the measurement manual (COSMIC, 2017b) functional size measurement with COSMIC FSM method involves three main phases: "Measurement Strategy Phase", "Mapping Phase" and "Measurement Phase". Figure 5 depicts these phases. In Measurement strategy phase, the purpose, scope, software architecture layers where the measurement will be carried out, level of decomposition, the functional users, and level of granularity are determined; In Mapping Phase, the functional processes, data groups, and data movements in the functional user requirements are identified; and In Measurement Phase, 1 CFP is assigned for each data movement identified and these data movements are aggregated to form the functional size of the focused piece of software (COSMIC, 2017b).

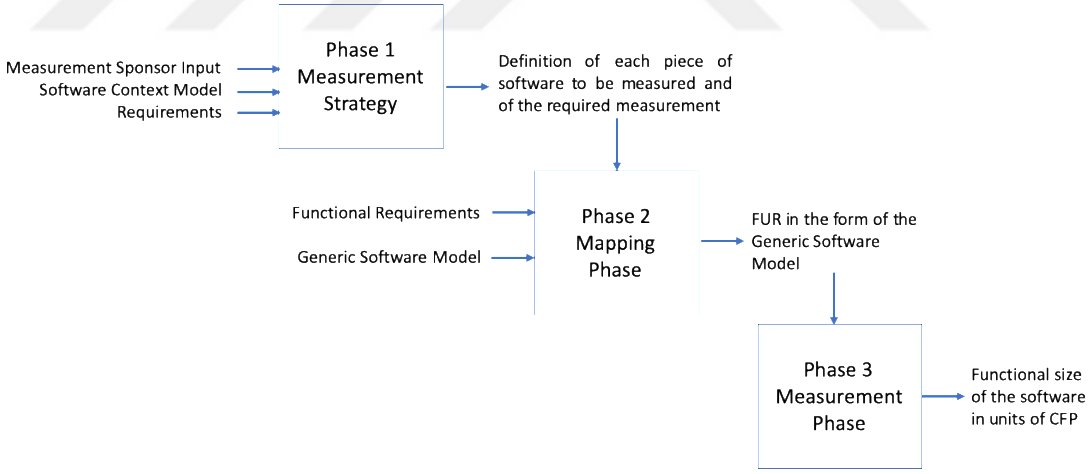


Figure 5 COSMIC Functional Size Measurement Phases (COSMIC, 2017b)

3.2.4. Results and Discussions

The data collected from the projects of three organizations is represented in Table 3.

Table 3 Project demographics adopted from (Hacaloglu & Demirors, 2019)

Organization	Project/ Sprint	Platform	Domain	Actual Effort in person-hour	Team's estimation	Status
Organization 1	Project 1	Web	Online Retailer	386,5	Estimate based on duration	Completed
	Project 2	Web	Online Retailer Update[6]	241,3	Estimate based on duration	Completed
	Project 3	Web	B2B E-commerce	363,5	No estimate entered	Completed
	Project 4	Mobile	Mobile Retailer	71,7	Estimate based on duration	Completed
Organization 2	Project1-Sprint 1	Web	Official Dealing Application	27	Estimate based on story points	Completed
	Project 1-Sprint 2	Web	Official Dealing Application	45	Estimate based on story points	Completed
Organization 3	Project 1-10 use cases	Web	Finance	225	No estimate	Partially completed

Among the COSMIC FSM method's phases shown in Figure 5, it is the mapping phase where the identification of the data movements is performed. The guideline for sizing business application software by COSMIC (COSMIC, 2017a) defines the data movement as being a flow of data group describing a single object of interest (OOI) in a functional process and clearly emphasizes that a data analysis is required OOI to identify data movements. The applicability of COSMIC FSM will be discussed to what extent these data movements can be identified, indirectly to what extent functional processes, data groups, object of interests can be identified.

With the framework of COSMIC FSM, we assessed the results in terms of availability of project description, data analysis process, requirements characteristics, and ease of mapping the user stories into single functional process and presented them in the following paragraphs respectively.

[6] the update project has been measured independent of the previous version. It was also defined as a new project in the issue tracking tool

3.2.4.1. Availability of Project Description

In COSMIC FSM, functional users are critical components to identify data movements. The functional users are generally made visible using a context diagram. To form a context diagram, a description of the system can be sufficient. However, excluding the project of Organization 3, there was neither a description and nor a context diagram that ease the identification of functional users at the first glance.

3.2.4.2. Data Analysis Process

As mentioned previously, data analysis is a suggested practice in COSMIC FSM especially for data-intensive applications such as Business Application Software (BAS) (COSMIC, 2017a) in order to identify OOI and their related data groups. In the guideline (COSMIC, 2017a), Relational Data Analysis, UML Class Diagram and Entity/ Relational Diagram are presented as recommended techniques for data analysis where the guideline connects entity-type in Entity/ Relationship Diagram, class in UML Class Diagram and Third Normal Form in relational data analysis with potential OOI in the software to be measured. However, in none of the projects, there was a data model. We have several implications form this finding. First, if data analysis using Relational Data Analysis, UML Class Diagram or Entity/ Relational Diagram is not a current practice in today's software development, the practitioners may perceive to perform it for only measurement purpose as an overhead. Consequently, this fact is likely to cause them to be distant to idea of adopting FSM in Agile teams. Second, it is a known fact that for a while, especially with the emergence of cloud computing and big data incorporating systems NOSQL database modeling is becoming popular (Banerjee & Sarkar, 2016). Moreover, the "quick schema iterations and frequent code pushes" ability of non-relational models make them more compatible for agility ("NoSQL Databases Explained," 2018). On the other hand, these non-relational models do not force for normalization like relational models. NoSQL community focus more on physical data model rather creating a logical data model (Hsieh, 2014). On the contrary, normalization is perceived as an unproductive practice for NoSQL databases and data modeling as inappropriate in Agile (Desmarets, 2018).

When we look closer to the process of finding OOI, we saw that it may further be discussed in the following point of views: Even though data modeling techniques such as Relational Data Analysis, UML Class Diagram and Entity/ Relational Diagram seem to be the approach suggested to determine OOIs of persistent data groups they fail to identify transient data groups (COSMIC, 2017a). This situation affects the accuracy of the measurement. In addition to the lack of such kind of data model, scarcity in the requirements in Agile, causing to not providing all the data groups which in turn complicating the understanding of which OOIs are present. Hussain et al. (2013) supports this view by stating that when complete data-group list and data models do not exist, requirements are poor resources to identify data groups for COSMIC FSM. Here, it is important to emphasize that even though briefly documentation in Agile can be shown as one of the barrier in the identification of data groups and OOIs, OOI identification is a

problem which already mentioned in the literature even for projects having more documented requirement artifacts and which are developed by not adopting Agile methods (Turetken, Top, Ozkan, & Demirors, 2008).

Regarding OOI, one more important detail is that as suggested in (COSMIC, 2017a) while analyzing data using Entity-Relationship Diagrams, the entities having "many-to-many" relationships are candidate for OOIs. To be able to define this type of relationships, it is challenging both to identify participating entities and their cardinality, which is a requirement detail that is not always obvious for the measurer to identify, as in some of the user stories in Project 1.

As the result of this case study which is conducted in Agile projects, it is seen that in COSMIC FSM, OOI identification is a challenging step and this finding is compliant with the previous studies in the literature and is not specific to Agile projects (Top, Demirors, & Ozkan, 2009; Turetken et al., 2008). Therefore, an enhancement in COSMIC FSM for identification of OOI identification concept or a more to-do-point approach for measurement in Agile projects seems to be needed.

3.2.4.3. Characteristics of the requirement artifact

We also assessed the characteristics of requirements artifacts we faced in the projects during the case study execution. As presented in 3.2.2. Case Description section, three organization have their own style for requirement specification. In the projects of Organization 1, requirements are documented in the form of user stories which have "As a user, I should be able to do task X" format. An interesting finding is that in Organization 1, apart from the functional requirements, social tasks such as meetings and technical implementation activities such as providing some connection to system are all written in the form of user stories. Another interesting thing is that "other tasks" are also estimated in the same way as the conventional user stories conveying functional requirements. This shows the fact that for this organization, the total estimated effort reflects not only the functional part of the project but all other tasks.

The implication from the tendency of the team which misapply both user stories and story points can explain the reason of high adoption of story points which are perceived by the teams that they can be used to fit for any kind of effort estimations.

In Organization 2, requirements are written in the form of user stories but in a more detailed format including normal flow, acceptance criteria, pre-post conditions where these are supported with mock-up screens. In Organization 3, use case approach is adopted, where fully dressed use case description showing all the interaction of the user with the system along with their respective actors, post-conditions, exceptions, prerequisite relationships are documented and screens are also present.

3.2.4.4. Ease of mapping the user story into single functional process

We assessed to what extent a user story can be mapped into single functional process. To judge the ease of mapping a user story into functional process we took into account two issues: First, for an accurate size measurement COSMIC Measurement Manual (COSMIC, 2017b) defines the required the granularity level of a requirement as the one in which functional processes and respective sub-processes can be identifiable. Second, since in this exploratory multiple case study, we deal with Agile projects, we also referred to the guideline of COSMIC for Agile Projects (Berardi et al., 2011). where the required granularity level of a user story is specified as it should define a single functional process.

When the requirement in the projects are assessed in this perspective, it is seen that projects of Organization 1 having varying levels of abstractions. The user stories of the projects of Organization 1 have either large user stories that describes more than one functional process or written in a very high abstraction level where it is difficult to identify functional processes, or in a very low abstraction level which corresponds to a step in the requirement. All these specification complicates the identification of functional processes and their respective sub-processes, data groups and their corresponding OOIs. The implication from this fact is that when the user stories are the only requirement artifact and when it is not written in the format required for the measurement as suggested in (Berardi et al., 2011) it becomes challenging to apply FSM. Documenting the user stories in Create, Read, Update, Delete, List (CRUDL) format can ease the process. On the other hand, in Organization 1, there was some sub-tasks which correspond to physical implementation and unit testing related information. These information, for example, implementation classes and their data attributes, were beneficial for the measurement which facilitated to understand some data groups and OOI. In Organization 2, user stories were supported with mock-up screen where with the navigation labels, text fields; with these functional processes, their corresponding sub-processes and data groups involved are identified easier when compared to the requirement specification style in Organization 1. Similar to the Organization 2, in Organization 3, in addition to the mock-up screens, the fully dressed use case description enhanced the understanding of sub-processes.

Since it is possible to use COSMIC FSM method to measure the software size at any stage of SDLC (COSMIC, 2017b), the post-development artifacts for example screens of the application in operation can be used for the measurement. In Project 4 of Organization 1, we could access to the completed software from the Apple Store. So, we aimed to compare the FSM results from the user stories and from the installed application screens. It is seen that the functional size of the software measured from user stories is 154 CFP and is decreased to 89 CFP when it is measured from the implemented application. The reason of this decrease can be the followings: first, the implemented version's functional size reflects only the functional processes that are initiated from the end-user perspective which is the customer in this mobile application. However, in the measurement results from user stories just 128 of 154 CFP are related to the customers. Moreover, during the measurement of user stories it is seen that two user stories belonging to the customer role

do not have a corresponding effort value entered in the system. This may show that these user stories are not implemented and therefore their corresponding functionality does not exist in the implemented system.

An advantage observed while conducting this measurement is that when the implemented software is measured, the measurer has the chance to see error/confirmation messages which are generally omitted in the one sentence user stories and consequently obtaining a measurement result which is more close to real functional size.

In COSMIC FSM (COSMIC, 2017b), a human, a device or another system can take the role of a functional user. Accordingly, in the user stories of the projects of Organization 1, it is seen that there exist system-type functional users from which the software under measurement receives some services. However, since there is no description of the system either in a context diagram format or in a natural language text format; it is challenging for the measurer to understand whether there is a persistent data storage which saves data or the data is sent to an external system. It may affect the overall functional size because the number of data movements may change and also the type of data movement involved.

3.2.5. Concluding Remarks for Multiple Exploratory Case Study 1

As a result of this case study, it is seen that COSMIC FSM measurement is challenging from user stories, many of the components related to the measurement method cannot be identified precisely, thus it requires assumptions to be made by the measurer. Agile teams do not create a data model; do not document all the data groups. On the other hand, supporting the user stories with mock-up screens facilitates very much the measurement by allowing to identify functional processes and data groups involved. In addition, step-by-step user-system interaction description via fully dressed use cases also beneficial.

After having completed this study, we concluded that relational databases are being replaced by NoSQL database models and practices belonging to relational data analysis are not used. Furthermore, now, it is known fact that due to emergence of service-oriented, cloud related applications, data-centric approach is being replaced by event-oriented approaches more and more. We started this study with Agile projects and saw that data-centric approach losing its compatibility with current projects and relational data modeling practices are started to not being adopted as it was earlier. This fact has formed to turn our perspective for the measurement into another direction.

3.3. Summary of the Chapter

In this chapter, first step in DSR according to Offermann et al. (2009) that is “problem identification” is presented. In the first part of this chapter, the SLR study that is conducted to identify the size measures/ estimates used in Agile software development projects and related challenges are presented. In the second part of the chapter, the exploratory case

study where COSMIC FSM is applied as functional size measurement method on Agile software development projects is presented. With this case study, we aim to observe practically the types of problems that are encountered during the application of COSMIC FSM process in Agile projects and to determine which measurement components that can be remained unidentified, and affecting the accuracy of the measurement. Based on the findings, we got insights of the solution. In the next chapter, the solution design is presented in details.

CHAPTER 4

SOLUTION DESIGN

By being motivated from the emphasis on events (Boner, 2017; Parulkar, 2020a; Sucaciu, 2020) in today's famous software applications, for example eBay, PayPal and LinkedIn (Zhu et al., 2015), we propose a size measurement model that counts the events in requirements specifications. Therefore, the proposed model presented in this article is based on event identification. However, since event concept is defined in various abstraction levels in software engineering domain, a literature review and exploratory case studies have been carried out to identify the most appropriate abstraction level of events in order to form an event-based software size measurement model. In this section, the research methodology that has been followed is described.

4.1. Literature Search II

There exist only handful studies in measuring the size of new generation software. Representative examples of size measurement of new generation software are encountered in microservice architecture related studies. Previously, Abeysinghe (2016) points to a misconception about the size of a microservice. Then, Engel et al. (2018) draw attention to the fact that 'What size does a microservice have" are shown amongst the challenges related to microservices. Later, Asik & Selcuk (2017) proposed a size measurement method for microservices where they take the counting of resources and clients that are in charge of interaction between microservices or external services. However, the success of their proposed method is not demonstrated yet. Vural, Koyuncu, & Misra (2018), within the scope of a case study, applied COSMIC FSM method to measure firstly the size of a monolith; and secondly the size of microservices which are created by dividing the monolith into microservices. They used domain driven design for this transformation. However, the success of size measurement with COSMIC FSM method was not assessed in their paper. Taibi & Systä (2019) proposed a measurement framework for microservices. Their framework is composed of four dimensions such as coupling, number of classes, number of duplicated classes and frequency of external calls. In this model, they connected number of classes and number of duplicated classes to size microservices.

When we discuss the proposed methods, we see that none of them address completely to problem that we want to study: early size measurement from software requirements. For instance, the measurement elements such as resource count and client count suggested by Asik & Selcuk (2017); and coupling, number of class, number of duplicated classes and frequency of external calls by Taibi & Systä (2019) do not address to the problem of early size measurement using functional requirements. Concepts such as coupling or class belong to the design and/or implementation stage of SDLC. However, a method which provide size measurement from requirements and effort estimation at the initial stages of SDLC is needed. In this perspective, the study by Vural, Koyuncu, & Misra (2018) where they used COSMIC FSM to measure the size of requirements can be considered an attempt. However, in their method, they studied the transformation of a monolith into microservices using Domain Driven Design; but, when an early effort estimation is needed in a project, how the system will be designed is not a decided step yet. In addition, as mentioned by Vural, Koyuncu, & Misra (2018) the problem in defining the boundaries can cause problems for developers when working on same data repositories. For this reason, we think that making a measurement over such a microservice decomposition still do not address the early size measurement.

Starting from this point of view, a literature research has been conducted as a part of the solution design which is also a suggested DSR practice by Offermann et al. (2009). This literature search is carried out to understand required and accurate abstraction level of events which can provide effective results for the measurement and also the method for identifying them.

After having decided to explore the usage of events for sizing the software, in this stage, we aimed to build a foundation of knowledge about the event concept in software engineering domain with a literature search. Firstly, with this aim, the usage of event in software engineering is observed. It is seen that events appear primarily in business process modeling domain. The related literature indicated that business process modeling methods are also being used to model the behavior of the functional requirements in software engineering domain. In this sense, business process models can be utilized to elicit and document requirements in Agile, which will further help for maintenance (Dragicevic, Celar, & Novak, 2014). According to Amjad, Azam, Anwar, Butt, & Rashid (2018), for example, with Event-Driven Process Chain (EPC), it is possible to model and verify the simple business requirements. The visualization helps to assess the logical existence of pre-/post- conditions and triggers as well as reengineering of business processes (Lubke, Schneider, & Weidlich, 2008). This type of visualization of requirements with a business process model can be specifically beneficial in Agile, by facilitating to depict the system holistically and can help the modeler to identify the gaps, inconsistencies, ambiguities and incompleteness in the requirements set caused by the inclination of Agile teams towards the minimal documentation.

In this scope, Event-Driven Process Chain (EPC) is a business process modeling language, developed at the University of Saarland, Germany in 1992 (Scheer, Thomas, & Adam, 2005). EPCs is acknowledged by many users due to its easy readability (Van der Aalst,

1999). Such that, even 15 years later, Dragicevic et al. (2014) emphasized that EPC modeling is adopted among business users due to its user friendliness and ease of modeling. Events, functions and control flows constitute the main elements of EPC. Accordingly, a function is "an activity (task, process step) which needs to be executed" and event is the pre-/ post condition of a function (Van der Aalst, 1999). Riehle et al., (2016) reviewed the related literature about the EPC and reported 14 variations of it where extended Event-Driven Process Chain (eEPC) is one of them. By comprising the original EPC (Scheer et al., 2005) which is made up of elements such as functions, events and connectors; eEPC extends the notation through adding the elements such as "organizational units", "information objects", IT systems and process refinements (Riehle et al., 2016).

Here, an important concern is the adaptation of the eEPC to the functional requirements, which is indeed a method to model processes. To overcome this concern, available applications of process modeling techniques into functional requirements have been observed. It is seen that the studies of using EPC for modeling requirements date to 2006. In 2006, Lübke (2006) suggested a method to transform use cases into EPC models where the author suggested an 8-step procedure to perform this transformation. Lübke (2006) pointed out that the control flow between use cases are not shown in use case diagrams. Today, this fact can also be observed in user stories which is the most widely used requirement specification format in Agile also lacks the control flow.

Later, Gross & Doerr (2009) investigated the effectiveness of EPC and Activity Diagram in the requirements engineering context, i.e. the authors investigated the suitability of both methods to support the requirement engineer. Recently, Dragicevic et al. (2014) developed a method called MEDoV for Agile software development in which the method focuses on and encompasses the elicitation, documentation and validation of user requirements. They used eEPC on which they applied a top-down approach where they identified high level process first and mid-level ones after. Here, Dragicevic et al. (2014) supports the view of the usefulness of business process models for requirements elicitation and documentation in Agile and hence maintenance.

In 2018, Amjad et al. (2018) based on a systematic literature review study pointed out that EPC can be utilized to model and verify simple business requirements. One of the result of the study by Amjad et al. (2018) is the lack of complex event processing preventing the modeling and verifying complex business requirements, where modeling and verification of simple business requirements is possible with atomic events. Accordingly, we decided to use eEPC in our modeling activities.

4.2. Multiple Exploratory Case Study 2

In this case study, with the view of using events for size measurement, we investigated whether it is possible to model the requirements with event-driven process chain. In

addition, considering various abstraction level of events, we aimed to identify the standard modeling abstraction level. Here, the goal is to investigate whether there exists a correlation between the size defined with events and effort spent to develop the software fulfilling the project requirements and how this correlation value performs when it is compared to the correlation value of size in terms of CFP and effort. Details regarding the case study are presented in the following sub-sections.

4.2.1. Case Study Design

4.2.1.1. Research Questions

To investigate the relevance of events for size measurement in software projects, following research questions (RQs) were formulated:

RQ1. How events can be used as a size measure?
RQ2. How event-based size measurement compares with FSM?

To answer RQ1, it is aimed to try to identify events in a given piece of software artifact, such as requirements, eEPC and aggregate them by adding them up. It is planned to assess the relevance of events occurring as a result of user and system related activities.

To answer RQ2, it is planned to measure the size of a given piece of software artifact using the COSMIC FSM method; then, investigate the relationship between number of events and effort; and CFP values and effort using Pearson Correlation Analysis and compare the correlation coefficients.

4.2.1.2. Modelling Procedure

After gathering knowledge about the event modeling in requirements and before conducting an exploratory case study to experiment the event identification, we faced with two types of challenging issues regarding the mapping of the functional requirements into event driven process chains:

Challenge 1: How to model the system: Entirely or each requirement individually?
We planned to model each requirement individually; the reason behind this decision is that, Agile projects are developed in a progressive manner; where not all the requirements are completely defined at the beginning of the project and changes happen frequently and are always welcome. For this reason, rather than adopting a holistic modeling view, we preferred to approach to the modeling in a bottom-up manner similar to the bottom-up estimation (Jørgensen, 2004b) which is calculating the effort of the project by adding the project activity estimates. For translating use cases to eEPC models, (Lübke, 2006)'s approach from use cases to EPC is inspired us and this approach is adopted and adapted for modeling the use cases and user stories.

Challenge 2: What is the accurate event abstraction level? In other words, which events to model and consider?

In order to decide on the appropriate abstraction level, it is important to define what an event means. Event is described by Chandy (2006) as "a significant change in a state". When different organizations' requirements sets are observed, it is seen that each organization has their own style of writing the requirements according to their own culture. Even though the requirements are in the form of user stories and use cases, the level of details in the requirements are varying and it is seen that interface design level details are sometimes injected to the descriptions.

On the other hand, it is also common that a requirement can be written without sufficient detail. The varying level of details can complicate to decide on "which event" to take into consideration because events can appear in various abstraction levels.

As mentioned previously, eEPC has been chosen for conducting the modeling activity due to its ease of use (Dragicevic et al., 2014). Among the modeling elements set, in our study we considered "event", "function", "position", "application", "file", "document", "list", "AND/OR/XOR connectors", "control flow", "information flow" and "relation" elements.

Even though there are attempts and approaches presenting the usability of eEPC in requirement management, in some pilot tries, we observed that with eEPC, it can be challenging to model the requirements. The reason for this challenge is that eEPC is not designed for modeling in the requirements level. Therefore, while modeling the requirements, the major questions were "Which event to consider?". For this aim, a literature review study was conducted on the modeling of requirements with eEPC method. It is seen that in the literature, there exists various type of events. These are "atomic level events vs. complex events", "system events vs. business events", "start-intermediate-end events", "external, state events" etc. Even, pressing a button, entering a key via keyboard (Helendi, 2018), moving the mouse can also be events. Scheer et al. (2005) stated that it is possible to create business process models in changing abstraction levels. When we go into requirement modeling level, we noticed that system itself should also have a role in the model. This approach helped us to be in conformance with the rule of eEPC saying that "events and functions should be alternating".

Consequently, as the starting point, we adopted a philosophy of modeling the requirements in high-level, which corresponds to model the interaction of user with the system. Generally, the only role owner was the human user of the requirement (that is actor in the use case terminology) who interacts with the system. However, we also plan to load a role to the system whenever it is required such as controlling a user action and offering the service according to the decision. We plan to collect and record from each project, the requirements; the user events, system events, and effort spent in accomplishing each requirement.

The first version of the model was formed in the lights of literature search, in which we hypothesize that a functional requirement can be modelled as a flow of events which occur

as a result of the actions triggered by user and system. This can be thought as modeling the interaction between the user and the system and events occurring as results of these interactions. The event produced as result of this interaction can be two kinds: user event and system event. In Figure 6, the first version of the model is presented.

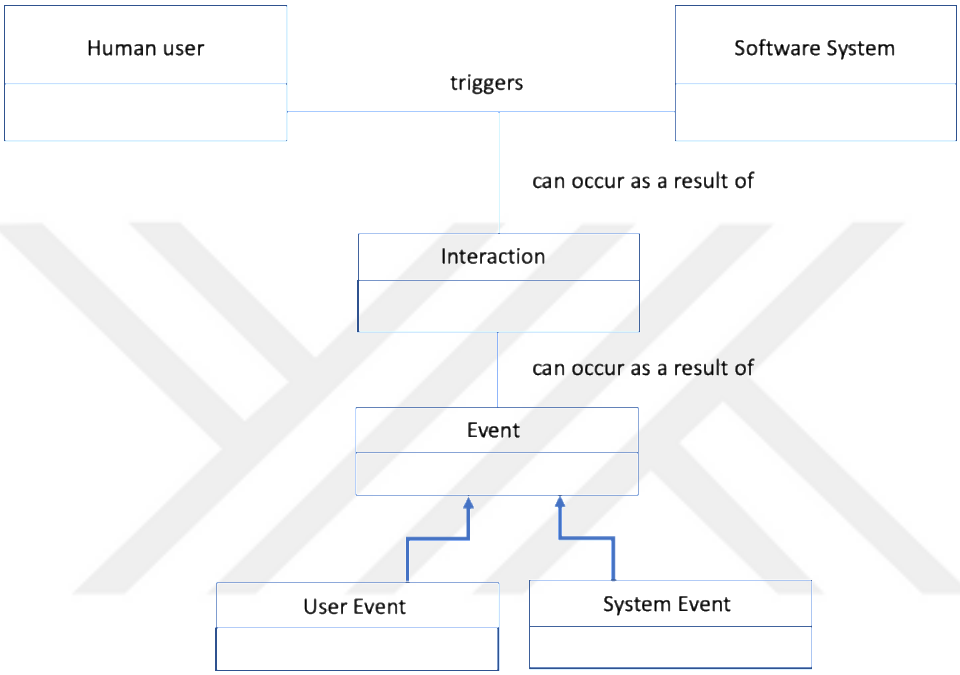


Figure 6 Model Version 1

4.2.1.3. Selection of the Modeling Tool

As the modeling tool for eEPC, we considered the results of the article by Amjad et al., (2018) which comprehensively reviewing the eEPC literature along with the modeling tools. Based on that article, Bflow Toolbox -an open source tool- through which syntactical check of the created models can be performed is chosen for modeling. This syntactical check facility has a special prominence because, eEPC has some restrictive rules such as "events and functions should alternate"; i.e. an event should always be followed by a function and vice versa. Furthermore, an event can never be followed by "XOR" and "OR" type connectors because events have no decision capability. These kind of modeling rules are prone to be violated by the modeler easily. This fact was mentioned also by Gross & Doerr (2009) who stated that this rule was omitted in some interpretations of the model. Therefore, utilization of such kind of tools can prevent these types of violations by performing the syntax check. An example of syntax check of Blow Toolbox is given in Figure 7.

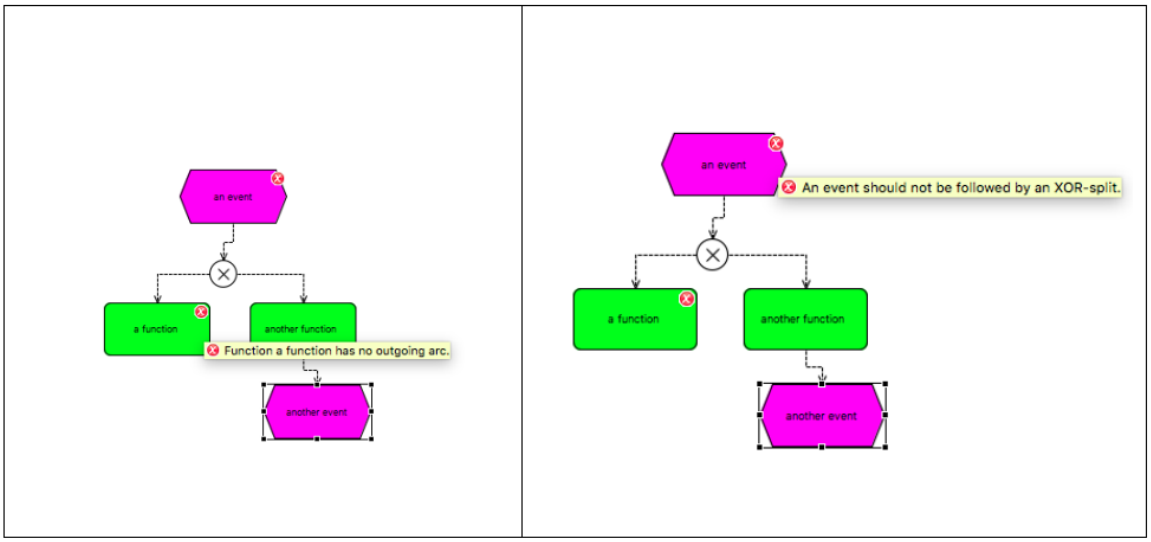


Figure 7 Syntax Check Facility of Bflow Toolbox

4.2.1.4. Case Selection Criteria

Our aim is to explore an event-driven size measurement model from functional requirements and how event-driven size correlates with actual effort spent in the software development projects. Thus, first criterion is the availability of requirements documentation and effort values. Since we investigate the applicability of extended Event-Driven Process Chain (eEPC) in requirements set, considering the different tendencies of software practitioners regarding the documentation of requirements, and the specific reflection of this behavior in Agile software projects, we decided to choose our cases from organizations having requirements specified in different styles and different level of details such as user stories and use cases. Other criteria include choosing projects operating on different platforms such as mobile and web and from different domains are among the case selection criteria.

4.2.1.5. Description of Cases

Case studies are implemented in two organizations. These organizations are adopting Agile philosophy and have different requirements specification styles. Organization A documents their requirements in the form of user stories, where they adopt both Scrum and Kanban. For each story, there exists sub-tasks including details regarding implementation and testing details. An issue tracking tool is being used for recording. We collected two projects from this organization. Project 1 is a mobile application developed for a retailer. The team estimates tasks based on duration. We gathered 11 user stories and their corresponding efforts and team's estimates. Project 2 is a web application for a retailer. It is a project having new requirements related with a previously developed project. It includes 9 user stories, efforts and team's duration-based estimates of the tasks.

Project 3 belongs to the Organization B. Scrum is used as the Agile methodology in the organization. It is about trademark, patent, design application. While documenting the requirements they follow a process such that forming the epic, identifying the user stories and task. The team gives care to model the interactions in the system atomically. There exist descriptions of the user stories as well as pre-post conditions, normal flow and mock-up screens. Two sprints are gathered from the Organization B.

4.2.2. Execution of Case Study

We modeled each requirement, with eEPC using Bflow Toolbox. In other words, during the case study, each requirement is modelled independently at an abstraction level considering events occurred as a result of a user and system related action.

In addition, since requirements in Agile have minimal specifications, we decided to model the "to be" situation as much as possible according to the content maturity. This decision involves to have assumptions in the requirements. To identify events in a more systematical manner, we followed the roadmap given below:

1- Identify the functional users
    a. Identify tasks belonging to these users
    b. Identify events occurring as a result of a trigger by the functional user
2- Identify system tasks
    a. Identify events occurred due to the system action

4.2.3. Results

Not all the requirements have the same abstraction levels. They vary from very general to very specific. Requirements having following status were eliminated:

- Having very generic description, not including clear enough details to be modeled
- Having very specific but technical detail intensive description
- Not including corresponding effort values

After modeling the requirements with eEPC, we counted the events in projects. Measurement results for each project is given below. Following the identification of events, we investigated the correlation between events and effort. Moreover, within the scope of a case study on the applicability of COSMIC FSM method, the functional sizes of these projects were measured in our previous study (Hacaloglu & Demirors, 2019). In that study, we concluded that it is challenging to apply COSMIC FSM in contemporary Agile projects due to the difficulty of identifying FSM method-specific measurement components and due to the need for lots of assumptions. These size values in CFP are used to carry out the Pearson Correlation Analysis with effort and the correlation result is compared with correlation value between event and effort.

From Project 1, which is a mobile application, we measured 11 user stories. User story-wise representation of size with events, effort spent, size in terms of CFP and correlation values are given in Table 4. It is suggested to draw a scatter plot whenever an analysis of the relationship between two variables need to be conducted (Howell, 2012). Scatter plots of event and effort and CFP and effort are presented in Figure 8.

As it is seen in Table 4, in Project 1, we observed a low correlation between event and effort but no correlation between CFP and effort. Even though the correlation coefficient is low, it is still possible to say that event and effort correlates better than CFP and effort. A reason of low correlation with effort can be related to the nature of Agile projects where the novel development is conducted at the same time with previous sprints corrections and updates. It is possible to say that these efforts are also incorporated to the stories but we measure the new request. An important implication can be the reliability of the effort values. Effort collection problem is already a mentioned challenge related to software engineering community in several studies in the literature (Özkaya, Ungan, & Demirors, 2011; VanHilst et al., 2011).

Table 4 Measurement results of Project 1

	User Story	**Effort**	**CFP**	**Events**
Project 1	1	0,5	10	4
	2	1	7	4
	3	3	10	7
	4	3,25	18	4
	5	3,25	16	5
	6	5,25	4	4
	7	5,3	13	5
	8	6	8	9
	9	10,5	50	17
	10	11	12	8
	11	22,7	6	6
	Total	71,75	154	73
	Correlation between event and effort			0,4
	Correlation between CFP and effort			0,1

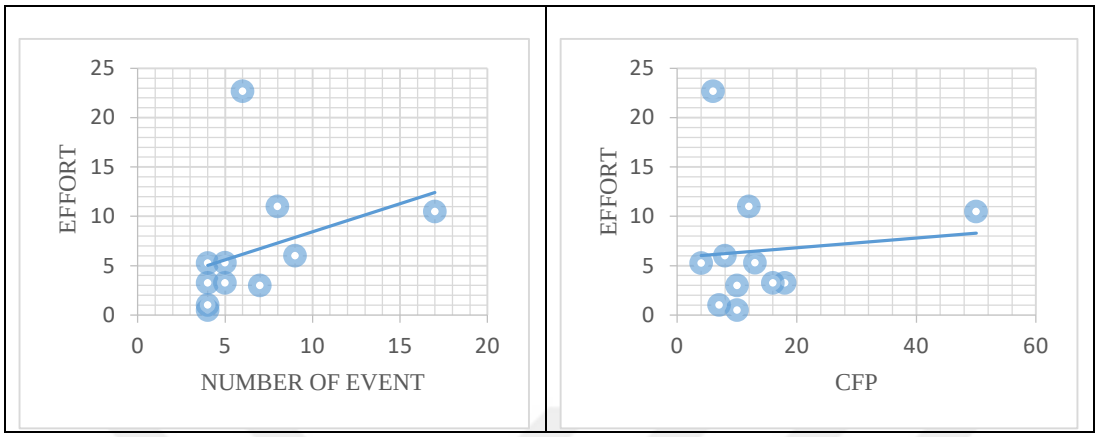


Figure 8 Scatter Plots of Project 1

Project 2 is a web application of a retailer getting services from SAP. Within the scope of Project 2 we measured 9 user stories. The story-wise representation of size with events, effort spent, size in terms of CFP and correlation values are given in Table 5.

Table 5 Measurement results of Project 2

	User story	**Effort**	**CFP**	**Event**
Project 2	1	13,00	10	6
	2	30,00	4	7
	3	34,00	10	7
	4	19,50	11	7
	5	75,50	53	15
	6	11,75	4	3
	7	29,58	7	3
	8	3,00	4	2
	9	25,00	12	5
	Total	241,33	115	55
	Correlation between event and effort			0,9
	Correlation between CFP and effort			0,9

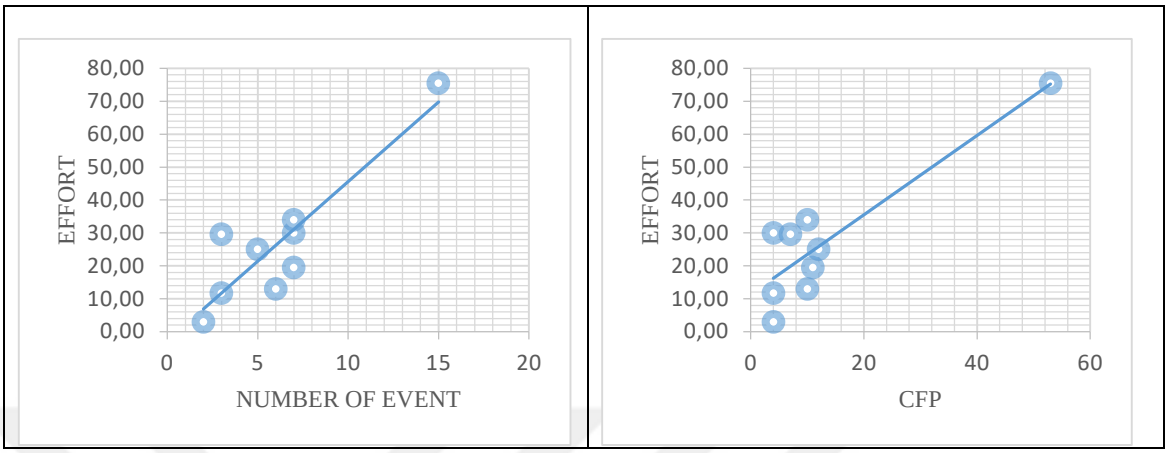


Figure 9 Scatter plots of Project 2

As it is seen in Table 5, and in Figure 9, in Project 2, we observed a strong positive correlation between event and effort and between CFP and effort. The correlation values obtained from these two projects provided promising results and motivated us to improve the approach and form a more concrete model for further analysis. Consequently, we planned multiple exploratory case study 3.

For Project 3, even though we gathered the effort spent for each of these two sprints, since story-based effort distribution is not kept by the team, we could not conduct the correlation analysis between effort and events. Measurement results is given in Table 6. We used projects of Organization A and Organization B to compare the ability of modeling the events appearing in two different documentation style.

When the projects of Organization A, where user stories were the only artifacts conveying the requirement information and sprints of Organization A, where there exist detailed descriptions, pre and post-conditions, mock-up screens which present requirements are compared; it is possible to say that modeling the requirements with eEPC is much easier and requires less assumptions regarding the to-be model in Organization B. Because, in Organization 1, the modeler needs to make assumptions due to the scarcity of requirement detail in an end-to-end manner. However, in sprints of Organization B, requirements are enhanced with descriptions and screens, which facilitate the modelling.

Table 6 Measurement results of Project 3

	Sprint	**Effort**	**CFP**	**Event**
Project 3	1	27	89	48
	2	45	225	299
	Correlation between event and effort			Not applicable
	Correlation between CFP and effort			Not applicable

4.2.4. Implications from the Multiple Exploratory Case Study 2

As a result of this case study, we had some implications especially originated from the assumptions we made during the modeling. Following findings can be considered as the reasons for the low correlation results:

a. In the eEPC models, only two types of events are modeled. Event granularity involving technical details were not taken into consideration. We realized here the importance of defining and identifying the right granularity level of events.
b. Since we model each requirement one by one, when there is a prerequisite relationship between two requirements, the starting/ ending events were counted more than once affecting the sensitivity of correlation results.
c. There is a high probability that reuse is common in some requirements which have similar functions affecting the actual effort spent. These type of stories can be eliminated especially for estimation purposes.
d. Modeling is a subjective activity, that is shaped according to the perception of the modeler. It is not feasible to ask each and every requirement execution to the team representatives. Therefore, assumptions have been made. For example, we made an assumption while modeling a requirement to consider the "to be" model rather than the "as is" model. However, there exists a probability that this assumption may not reflect the actual way the requirement has been developed. In addition, it was difficult to be standard in these types of assumptions.
e. The requirements in software projects are real data produced by the developers. It is difficult to gather high number of dataset. Furthermore, requirements set was not perfect; there was ambiguity in requirements and not all of them were complete as well. The cases where the modeler has no understanding are eliminated and not modeled.
f. From the COSMIC FSM perspective, data modeling is not performed in any of the projects and this situation makes difficult to identify OOI and hence causing measurer to make assumptions in COSMIC FSM.

4.3. Literature Search III

Before conducting third multiple exploratory case study, we aimed to improve the first version of the measurement model by taking the implications presented in section 4.2.4 into consideration and exploring the literature in a more detailed way to identify the most accurate abstraction level of events. The main concerns were to find the event types existing in the literature and those events that can be identified with eEPC modeling of requirements.

One conclusion from this literature research is that, to the best of our knowledge, there is no conventional classification of events exist in the literature yet. It is observed that different naming which overlap in meaning are being used. We present a categorization

of events according to various descriptions encountered in the sources from the literature. However, these different types of event types mentioned had different abstraction levels and overlapping nature. The events mentioned in the literature are given in Table 7.

Table 7 Event types mentioned in the literature

Event types	**Source**
User event	(Aarab, Saidi, & Rahmani, 2016; Chima, 2018; Davis, 2001)
Domain event	(Fowler, 2005; Vernon, 2013)
Software event	(Davis, 2001)
Triggering event	(Amjad et al., 2018; COSMIC, 2017a)
State describing event	(Davis, 2001)
Trivial event	(Bögl, Kobler, & Schrefl, 2008; Software AG, 2016)
External Event	(Aarab et al., 2016; Davis, 2001)
Business Event	(Aarab et al., 2016)
Primitive Event	(Aarab et al., 2016; Kappel, Pröll, Retschitzegger, & Schwinger, 2001; Kong, Jung, & Park, 2009)
Complex Event	(Amjad et al., 2018)
Atomic Event	(Amjad et al., 2018)
Hardware event	(Afshar et al., 2020)
System Event	(Aarab et al., 2016; Chima, 2018)
Composite event	(Kappel et al., 2001; Kong et al., 2009)

From the gathered sources, based on our understanding, we made a synthesis and a classification of events. Figure 10 visualizes the classification of events. In the Figure 10, the events that we have identified are represented in the form of a class diagram.

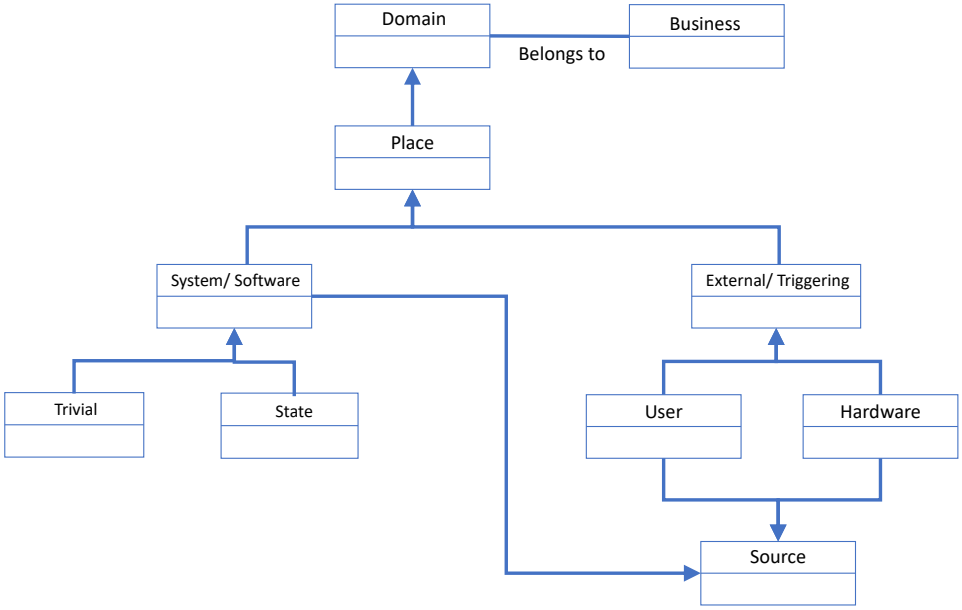


Figure 10 Synthesis of events from the literature

A “domain event” is something occurring in the domain (Vernon, 2013). A domain belongs to a business. Business has “business event” representing an event related to a business rule (Aarab et al., 2016). A domain event can be occurring within the software system (Aarab et al., 2016; Chima, 2018; Davis, 2001) and can be called as “system event” or “software event” or can be occurring outside the system and can be called as “External event” (Aarab et al., 2016; Davis, 2001) or “Triggering event” (Amjad et al., 2018; COSMIC, 2017a) or internally as “Trivial event” (Bögl et al., 2008; Software AG, 2016) and State event (Davis, 2001).

The events which are occurring outside the system can be in two categories: “user event” (Aarab et al., 2016; Chima, 2018; Davis, 2001) and “hardware event” (Afshar et al., 2020). Here, a user can be a human or another system. On the other hand, software/system, user, and hardware events can also be classified as events with respect to source.

In addition, there are other events which we have identified that could not be represented in this figure. Therefore, another means of classifying events is based on the structure of events as being “atomic” and “complex” (Amjad et al., 2018), primitive (Aarab et al., 2016; Kappel et al., 2001; Kong et al., 2009), composite (Kappel et al., 2001; Kong et al., 2009). Primitive events are by the way is the top-category for user, external and business events (Aarab et al., 2016).

Based on the event types gathered from the literature given in Table 7, to concretize the measurement model, we updated the model version 1, with model version 2 as presented in Figure 11. As it can be seen in in Figure 11, from the events gathered as a result of a literature review presented in Table 7, we decided to consider an event abstraction level based on the source it is generated. We envision that during the case study execution, this classification is likely to provide a more concrete understanding of events while modeling.

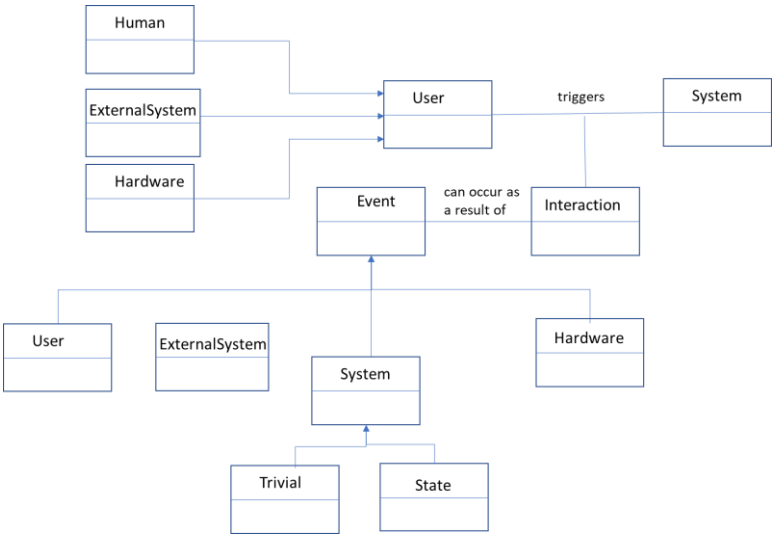


Figure 11 Model Version 2

According to the Model Version 2, presented in Figure 11,

- A human user, an external system or a hardware device triggers the system under measurement by interacting with it,
- An event can occur as a result of this interaction
- This event can be a human user event, external system, hardware or system itself event which is classified based on the source it is generated from
- A system event can be a trivial event or a state event which is occurring as a result of the state of a concern in the process, e.g. "product exits in stock". Here, we value the behavior of the system itself specifically, because, for example, if it pulls something from another system, it creates a request which can be considered as an event; whereas if the system itself carries out the completion of a process, this can also be considered as another event.

To adapt eEPC into requirement level, before carrying the Exploratory Case Study 1, we reviewed application of eEPC in requirements. This time, we also observed other process modeling tools adaption examples and following studies inspired us while modeling the requirements using eEPC. For example, Pavlovski & Zou (2008) used BPMN to model non-functional requirements where the flow of operation between the customer and ATM is modeled using BPMN; Lübke (2006) who modeled the use cases with EPC; and Lubke et al. (2008) modeled uses cases with BPMN.

4.4. Multiple Exploratory Case Study 3

In the second case study, by motivating with promising results and implications from exploratory case study 1, we aimed to conduct a more systematic event exploration and modeling in order to concretize a measurement model. This is accomplished by deepening the event types with the knowledge gathered from the literature regarding events. Moreover, in this case study, we investigated the correlation between the size defined with events and effort spent to develop the software fulfilling the project requirements and how this correlation value performs when it is compared to the correlation value of size in terms of CFP and effort.

4.4.1. Case Study Design

This case study has two main purposes: first one is a discussion on the applicability of eEPC modeling in requirements level that it is not originally designed for; and the second one is the identification of different types of events and assessing how they correlate with effort. For this aim, different from the previous case study, we carried out a more focused study to model requirements with eEPC and to identify possible event types.

4.4.1.1. Research Questions

To carry out this case study, we identified following research questions (RQ):

RQ 1: What are the distinct event types exist in different projects?
RQ 2: Do the events produced via eEPC are correlated with actual effort?
RQ 3: Are events give better correlation results when compared to CFP?

4.4.1.2. Modeling Procedure

We decided on the level of abstraction of events according to the source they are occurring from and provided examples of them for making them more concrete to facilitate the measurer/modeler's work. These examples are presented in Table 8. We aim to model each elementary step which is valuable in terms of event generation during the system execution.

Table 8 Events considered and examples

Event type based on the source	**Example**
Human User	"list products is selected", "request to add comment"
External System	"payment approved", "response received"
Hardware	"temperature is above 20"
Trivial	"new customer is saved", "message is displayed"
State	"product does not exist in stock", "selected option is xyz", "password is correct"

4.4.1.3. Case Selection Criteria

To be able to answer the research questions, we aimed to focus on projects from different organizations which adopt Agile methods but having different requirement documentation styles (in terms of type user story, use case etc. and detailing levels).
The reason for concentrating on different organizations' project is the possibility discovering different type of events. Because, different organizations have different requirement documentation styles which can provide more insight in assessing the applicability of eEPC modeling. In addition, they will have different effort recording habitudes. Additional criterion is the nature of the application for example web, mobile, integration of external system etc.

4.4.1.4. Description of Cases

To accomplish this aim, we gathered three projects which are selected from three different organizations. The common point of these organizations is the adoption of Agile philosophy. However, each company have their own style of requirement documentation. The demographic structure of the cases is presented in the Table 9.

Table 9 Description of Cases

Organization	Project	Requirement Documentation Style	Type
A	1	User Stories	Business application software (mobile)
B	2	Fully dressed use cases and screens	Business application software
C	3	User stories and technical sub-tasks	Business application software (SAP integrated)

Project 1 is belonging to organization A and it is an example of crowdfunding application. In this project, requirements are in the form of user stories. From the requirement set we acquired, there were pure functional requirements and their corresponding effort values. In addition, there were some requirement statements which were in the form technical tasks, mixture of functional requirements and implementation tasks, such as front-end development and some requirements specified with insufficient detail.

Project 2 is from organization B which is operating in Australia and from the finance domain. Requirements are documented in the form of use cases along with their descriptions. The descriptions are in the fully-dressed format: there exists actor, 1-2-line description of the use case, prerequisite, main flow of the use case, post condition, exceptions representing the alternative flows. Especially main flow is documented as the step-by-step interaction of the user with the system. This issue facilitated the modelling. Furthermore, each use case is supported with mockup screens to facilitate the comprehensibility of the use case in terms of modeling.

Project 3 is from organization C a system which gets services from SAP. It is an event-based system where event-listener and event-dispatcher are used. Requirements and corresponding effort were gathered from the issue tracking tool of the company. Requirements are documented in the form of user stories. In many of the user stories, there exists efforts values corresponding to these stories. In some stories there were no effort assigned so we excluded these stories from the case study. Also, some user stories are supported with sub-tasks which include technical implementation and tests details.

4.4.2. Execution of Case Study

The steps were taken during the cases study execution are as follows. The requirements sets of the projects listed in Table 9 are modeled with (eEPC). The events are counted. The Pearson Correlation Analyses are conducted between the total number of events and efforts spent on completion of these projects. These values are compared with sizes of projects obtained using COSMIC FSM values. The roadmap followed is as follows:

1- Identify the components of eEPC model
    - Identify the functional users (human, another system, hardware)
        - Identify tasks belonging to these users,
            - Identify events (external, triggering) occurring from the user
    - Identify system tasks
        - Identify events (trivial and state)
2- Create eEPC models considering the rules imposed by the method
3- Count the total number of events,

In addition, in the previous case study, while modeling, we tried to create "to be" models which might not be reflecting the "as is" process that has actually been developed. In this case study, more care is given to model what is written exactly within the reasonable limits.

4.4.3. Results

In the analysis part of the present study, we first explored the relationship between the events and actual effort spent using Pearson Correlation Analysis. It is suggested to draw a scatter plot whenever an analysis of the relationship between two variables (Howell, 2012). Then, we compared the correlations between events and efforts with correlations between the functional size measured as COSMIC Function Point (CFP) and effort. Since we do not have a statistically significant number of project for each organization; and we choose to investigate task-wise relationship between the number of events and actual effort, rather than project-wise.

Similar type of study was conducted by (Commeyne et al., 2016; Ertaban, Gezgin, Bagriyanik, Albey, & Karahoca, 2017; Hacaloglu, Deveci, Bağrıyanık, & Demirors, 2019) where the authors created effort estimations regression model based on CFP on the tasks of a single Agile projects tasks. In addition, we envision that task-based estimation can be useful in Agile projects where a new requirement or a change in requirement are welcome any time during the development. It can also provide to make estimation quickly when compared to conducting subjective estimation sessions such as planning poker meetings. In the following sub-sections, the results for each project are presented.

4.4.3.1. Project 1

For Project 1, system, user and total number of events, functional size in COSMIC Function Point (CFP) and actual effort is listed in Table 10. The Project includes 15 user stories however two of them were eliminated because they do not include sufficient detail for modeling with eEPC. Figure 12 presents the scatter plot for the event -effort distribution and CFP-effort distribution of Project 1.

Table 10 Measurement Results of Project 1

Story ID	**Effort (person-day)**	**CFP**	**System Events**	**User Events**	**Total Events**
1	1	3	5	2	7
2	2	3	5	4	9
3	1	1	5	2	7
4	1,50	2	7	3	10
5	4	8	6	4	10
6	4	2	4	2	6
7	12	30	9	4	13
8	26	20	12	4	16
9	6	10	8	3	11
10	3,50	4	4	2	6
11	0,50	2	5	2	7
12	1	2	2	0	2
13	3	3	4	3	7
Total	65,5	90	76	35	111
Correlation between event and effort				0,8	
Correlation between system event and effort				0,9	
Correlation between user event and effort				0,5	
Correlation between CFP and effort				0,79	

As it can be seen in Table 10, there exists a high correlation between both events and effort and CFP and efforts. The CFP measurement is carried out by the analyst of the project and this can be a sign that CFP measurement has been conducted with more knowledge. In addition, system events are the best correlating variable with effort. This finding motivated us to investigate further the system events.

In Figure 12, it can be observed that the correlation between the number of events and effort tends to increase as the number of events grows. This tendency is visible from the number of events starting from 10. For this reason, it is worth to investigate a project-wise correlation to observe whether this tendency exist in project-wise values. Similar tendency of is not observed between CFP and effort.

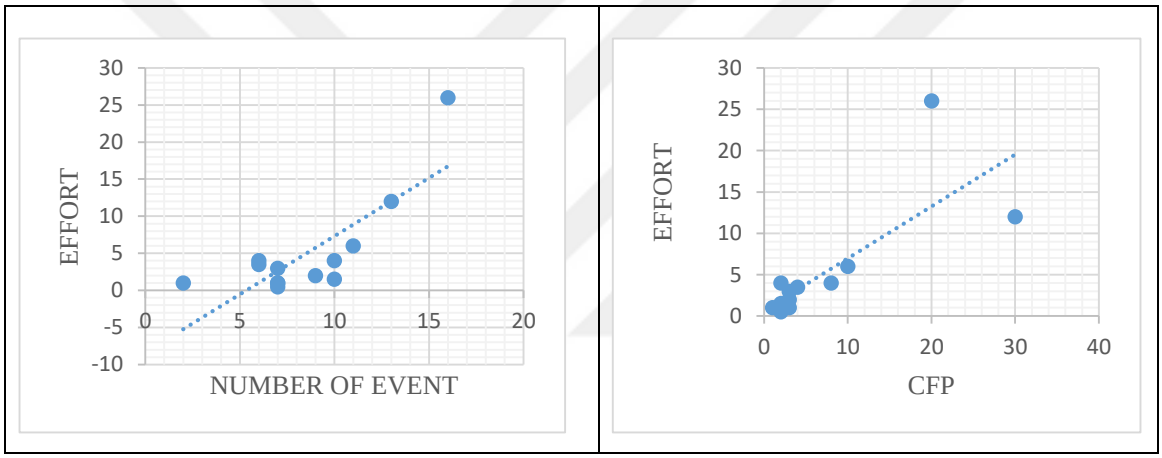


Figure 12 Scatter Plots of Project 1

4.4.3.2. Project 2

The analysis results of Project 2 are given in Table 11. According to the Table 11, events have higher correlation than CFP with effort. Similar to the Project 1, in Project 2 system events is the variable which correlate with effort the best. Therefore, we want to focus specifically to the system events in our future research.

Table 11 Measurement Results of Project 2

Use case	Effort (person-hour)	CFP	System Events	User Events	Total events
1	62,5	24	11	5	16
2	45	14	3	3	6
3	40	3	7	2	9
4	35	9	3	4	7
5	20	5	2	3	5
6	20	9	2	3	5
7	18,75	18	4	4	8
Total	241,25	82	32	24	56
Correlation between event and effort				0,8	
Correlation between system event and effort				0,8	
Correlation between user event and effort				0,4	
Correlation between CFP and effort				0,5	

The scatter plot for effort and total number of events and CFP and effort are presented in Figure 13. In this dataset, not all use case was complete and there were information regarding how much percent are complete, for this reason we rounded less than 100% completed use cases to 100% before the analysis.

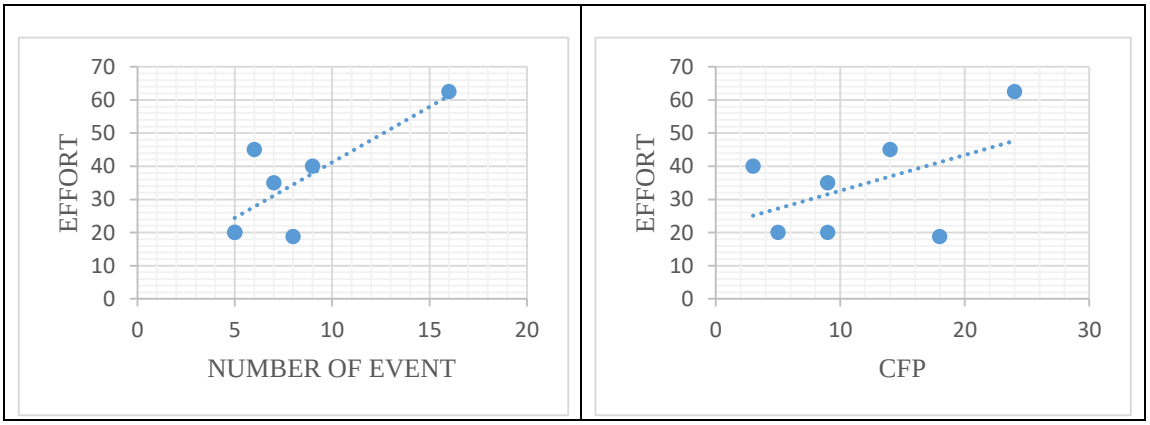


Figure 13 Scatter Plots of Project 2

4.4.3.3. Project 3

In Project 3, we modelled 27 user stories. Table 12 indicates the effort spent for each user story, functional size in CFP, system, user and total number of events. In this project, user stories have sub-tasks including technical details and, in some stories, there were effort breakdown for these sub-tasks where in some others there is not. This issue creates on the measurer's mind a question such as "the effortless tasks are not implemented?". The scatter plot for effort and total number of events is presented in Figure 14.

Similar to the previous projects in this case study, events correlate with effort better than CFP. System and user events have similar correlation values with effort. The correlation between CFP and effort is very low. The reason behind this low correlation can be the lack of data model that is facilitating the determination of OOI which constitutes a critical measurement component in COSMIC FSM method.

This fact is not observed in Project 1, because in Project 1, COSMIC Functional size measurement has been conducted by the project analyst in contrast to Project 2 and Project 3 that we conducted the functional size measurement. This point is open to debate because if measurer is not a project team member, he/she will be an external measurer and normally have less information than a team member, it can be possible that they may not obtain the exact size and this fact complicate to judge the correlation value with effort.

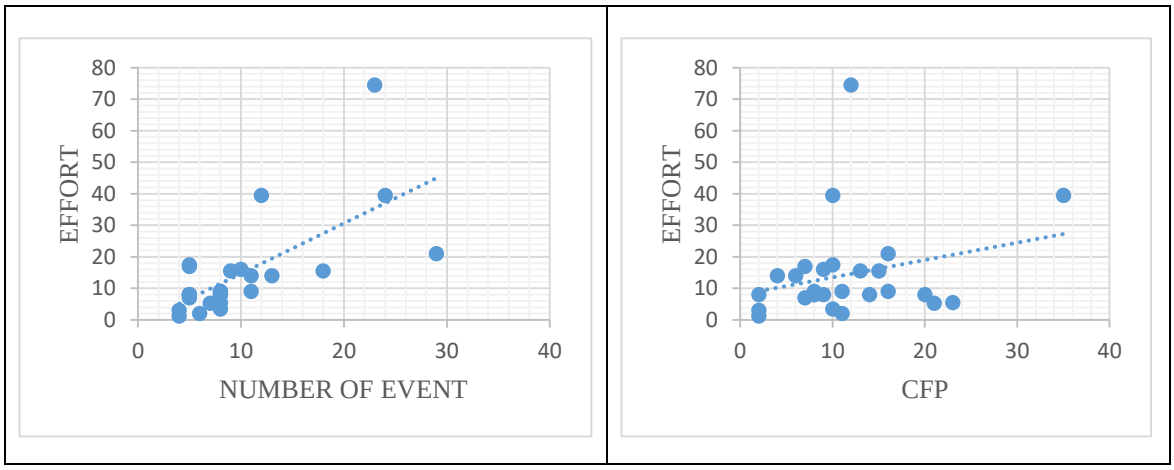


Figure 14 Scatter Plots of Project 3

Table 12 Measurement Results of Project 3

User Story	Effort (person-hour)	CFP	System Events	User Events	Total Events
1	74,5	12	16	7	23
2	39,5	10	8	4	12
3	39,5	35	20	4	24
4	14	4	10	3	13
5	8	2	5	3	8
6	21	16	23	6	29
7	17,5	10	3	2	5
8	17	7	3	2	5
9	16	9	6	4	10
10	15,5	13	6	3	9
11	15,5	15	13	5	18
12	7	7	3	2	5
13	3	2	2	2	4
14	5,5	23	5	3	8
15	14	6	8	3	11
16	9	16	5	3	8
17	9	8	5	3	8
18	9	11	8	3	11
19	8	14	3	2	5
20	8	8	3	2	5
21	8	9	3	2	5
22	8	8	3	2	5
23	8	20	5	3	8
24	5,25	21	4	3	7
25	3,5	10	3	3	8
26	2	11	3	3	6
27	1,25	2	2	2	4
Total	386,5	309	178	84	264
Correlation between event and effort					0,68
Correlation between system event and effort					0,65
Correlation between user event and effort					0,73
Correlation between CFP and effort					0,26

4.4.4. Implications from the Multiple Exploratory Case Study 3

In the Exploratory Case Study 2, we observed the requirements of three projects gathered from three different software organizations. Each organization has different style of documenting the requirements. As a result of this case study, we obtained promising results. Events are correlating with effort. In project 2 and 3 events correlates with effort better than CFP. In addition, we identified the importance of events which are occurring as a result of a system activity.

Deeping the research on system events can also contribute for effort estimation especially when using multiple regression analysis.

Moreover, we assessed the impact of requirement documentation style on event identification. Even though in Project 2, there were fully-dressed use cases with mockups, this was not the case for user stories in Project 1 and Project 3. For user story type requirement specifications, we modeled what we could understand from one sentence explications as in Project 1 and user stories and some supporting technical tasks as in Project 3. For this reason, the comprehensibility of the stories was not perfect and the modeling was subjective due to this fact. Consequently, the ones which we could not understand their flow are eliminated from the case study. This finding explains the criticality of defining the requirements in an understandable level for independent measurers. The same conclusion can also be drawn from CFP and effort correlation because in Project 1, COSMIC FSM was conducted from the project analyst and only in this project CFP value correlated well with the effort. It can signify that other projects do not contain documentation which can be sufficient to conduct FSM. In none of the projects we saw a data model which is further complicating the identification of OOI in COSMIC FSM.

4.5. Summary of the Chapter

In Solution Design Chapter, we presented four parts that describe four main activities that were conducted to design a solution. In the first part, the literature searches II where available size measurement studies regarding new generation architectures were presented. In the second part of this chapter, the first version of the model and Multiple Exploratory Case Study 2 are described. In the third part, results of another literature search where different event types that are found from the literature are presented. In the fourth part of the chapter the updated model and Multiple Exploratory Case Study 3 are described. In Table 13, the measurements are given in a summary. In the next chapter, the event-based size measurement model is presented in detail.

Table 13 Summary of Measurement Results

Study	Projects	Type	Effort	CFP	Total Events	Correlation between CFP and effort	Correlation between events and effort
Exploratory Case Study 2	Project 1	User story	71,75 (person-hour)	154	73	0,1	0,4
	Project 2	User story	241,33 (person-hour)	115	55	0,9	0,9
	Project 3	User Story	72 (person-hour)	314	347	Not applicable	Not applicable
Exploratory Case Study 3	Project 1	User Story	65,5 (person-day)	90	111	0,79	0,8
	Project 2	Use case	241, 25 (person-hour)	82	56	0,5	0,8
	Project 3	User story	386,5 (person-hour)	309	264	0,26	0,68

CHAPTER 5

EVENT POINT: EVENT-BASED SIZE MEASUREMENT MODEL

In the exploratory case study 2 and 3, we took projects from different organizations to explore which types of events that can be identified from the requirements sets and to assess the usability of events to define a size measurement method. We used these projects, to investigate the applicability of event identification with the help of process modeling with extended Event-Driven Process Chain (eEPC) diagram on Agile software requirements which do not contain comprehensive information.

As a result, we realized

- the criticality of defining the right abstraction level of events, especially in order to resolve possible confusions on the modeler's perspective;
- a need for a systematic and repeatable measurement model;
- a correlation between the events and the actual effort spent on these projects,

In the lights of all these findings, in this stage we aimed to create our event classification from the measurement perspective. To create an event-based size measurement approach, a measurement model is needed to be defined. This model should describe the identification of events as precise as possible in order to be repeatable. In this sense, the most important challenge is to define the right level of abstraction, in other words, right event types because an event can exist in various levels of abstraction.

As mentioned previously in section 4.1, events can have varying levels of abstraction. Therefore, deciding on which abstraction level of events should be considered at the point of measurement can be significant both for the repeatability, objectivity, reliability and the accuracy of the results. In addition, to be repeatable and generic, the proposed event-based size measurement method should be independent of technology adopted and other implementation details, as in case in the available functional size measurement methods such as COSMIC FSM.

Yet another aim while defining the model is the applicability of the measurement during any stage of the development process which is also the case in COSMIC FSM. For example, measurers might have aims different than estimation; they might want compare

the size of completed software with the size obtained from the user stories. The measurement model needs to respond to these needs.

Considering all the above mentioned, in the following section, first, event-driven software model is presented; second, event categorization is described; and third the measurement steps are defined.

5.1. Description of the Event- Driven Software Model

Within the scope of this study, the software model whose size is aimed to measure is presented in Figure 15. In this model, user triggers the software system by interacting with it. As a result of the interaction, a single or multiple events occur. Software system can be made up of services. Each service is responsible of performing some kinds computations. A computation can be a display, recording, calculation-processing, retrieval, decision and communication activity. As a result of a computation, an event can occur. An event can have services which subscribes to it and an event can be published by a service.

Decomposing the events as computation events occurring as result of system operation can ease the understanding of measurer about the existing events in the domain and can also contribute to the estimation of effort by making it possible to apply multiple regression analysis.

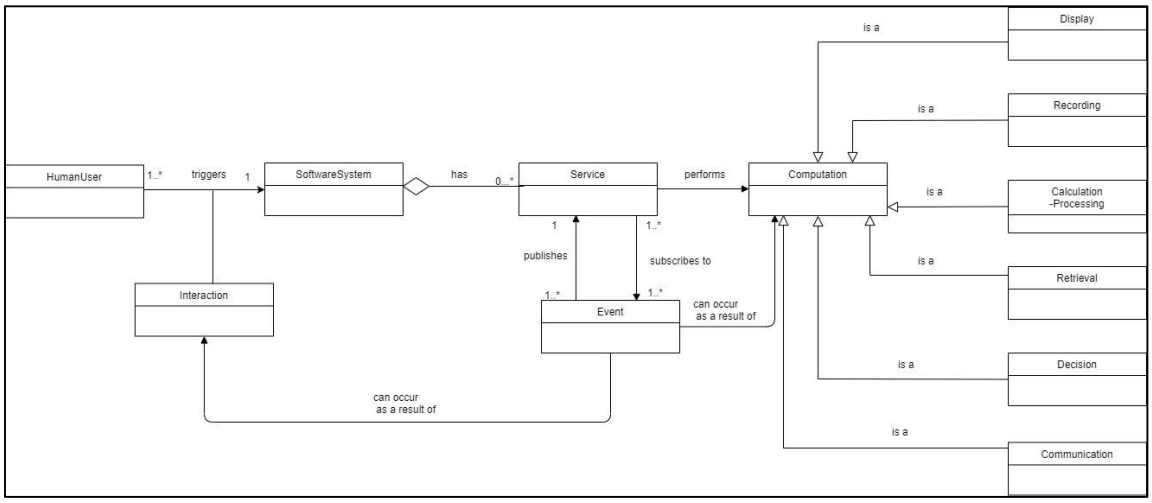


Figure 15 Model Version 3

5.2. Description of the Event-based size measurement method

Similar to the COSMIC FSM method, we envision that the measurement requires mapping and measurement stages. In other words, the measurement process involves extraction of events and assigning them numerical scale by aggregating them.

5.2.1. Method Principles

In this stage, we define a way of mapping the functional requirements into events by describing the events we consider for the measurement. One important thing that needs to be emphasized is that different from COSMIC FSM method, which binds the data movement with an Object of Interest (OOI); the proposed measurement method will not be data-oriented but behavioral, in terms of events. From the event-driven software model presented in Figure 15, we propose following event abstraction for mapping and consequently a measurement philosophy:

"Computation event": This event category covers the type of events occurring as a result of a computation which is performed when the process is supported with the software system.
While defining the categories involved in computation events, we considered the microservices architectural style in which a service is focused on a single purpose. Such an example is given by Skowronski (2019) as follows:

> "When an order is placed on an eCommerce site, a single "order placed" event is produced and then consumed by several microservices:
> - The ordered service, which could write an order record to the database.
> - The customer service, which could create the customer record.
> - The payment service, which could process the payment."

Considering the example of Skowronski (2019), in our model, we consider that in the end of each microservice activity- that we can call them as "computation"-an event is produced and we define a software system as a set of computation events. For the naming of sub-categories of these computation events, we inspired from the classes of activities in a transaction represented by (Fetcke, Abran, & Dumke, 2001). According to (Fetcke et al., 2001) , there are seven types of transactions such as entry, exit, control, confirm, read, write, and calculate.

However, we would like to draw attention to the fact that Fetcke et al. (2001) connects these transaction activities with data elements which are not the case in our approach where the focus is on the events. In our method, we assume that event(s) can occur as a result of a specific condition. Accordingly, computation events are divided into seven categories: *"System Boundary Event", "Display Event", "Calculation/Processing Event", "Record Event", "Decision Event", "Retrieval Event"* and "*Communication Event"*. The event categorization is presented in the form of a class diagram in Figure 16.

1- **System Boundary Event:** is the event type that can occur when the user specifically triggers the software system.
2- **Display Event:** is the type of event that can occur as a result of the software gives an output to the user. The output can be a form, a message etc.
3- **Calculation/Processing event:** is the type of event can occur as a result of performing a calculation or processing. A calculation can be an arithmetic operation, for example,

the calculation of bill of an order or a scheduled job which may create event due to triggering internally.

4- **Record event:** is the type of event that can occur when a successful recording is accomplished.
5- **Decision event:** is the type of event that can occur as a result of an evaluation of a condition. Example for this type of event can be “user credentials are correct”, “there is not sufficient item in stock”, “requested items exceed to the stock limit” etc.
6- **Retrieval event:** is the type of event that can occur when a retrieval is done.
7- **Communication event:** is the type of event that can occur when a service communicates with an external entity, for example a service.

To identify these computation events in a functional requirement, we adopt eEPC which is an easy to use technique (Dragicevic et al., 2014; Van der Aalst, 1999) to model the flow of events.

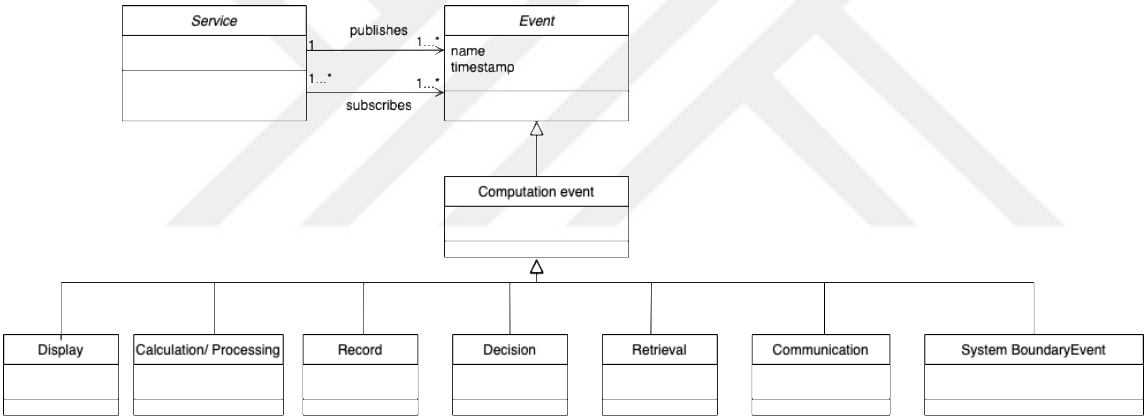


Figure 16 Categorization of Events

5.2.2. Mapping roadmap

We propose following roadmap for the identification of events:

A. Identify the input for modelling

Take functional requirements for example user stories and use cases in a given piece of software as the input for modeling. These will convey functional requirements in a piece of software whose size will be measured.

B. Identify Users triggering the system

A user story or use case is written generally from the human-user perspective in business applications, or hardware and other systems.

C. Model the requirement

We adopt a bottom-up approach for modeling and treat each requirement independently. This will help to size the software especially in Agile software development projects which are progressing in iterative and incremental manner, where the measurement needs to be applicable to the available requirements set. The idea behind this choice is that, at a given point in time in SDLC, not every requirement is complete. The measurers should measure/estimate based on what they have in hands. In addition, trying to create a complete process flow can be complicating.

With eEPC method, model each requirements as a sequence of steps and look for the seven types of events described in section 5.2.1. Consider the interaction of the user and "*System Boundary Events"* occurring as a result of an activity of the user. According to the system behavior, identify the *"Display event", "Calculation event", "Record event", "Decision event", "Retrieval event"* and "*Communication event".*

While modeling takes into consideration what is written in the requirement that is the "as is case", because, trying to create the "to be" situation -an ideal flow of the requirement- may not reflect the real implemented version and hence, may not reflect the size of the software accurately.

5.2.3. Measurement Stage: Definition of the measurement unit

We assign 1 Event Point to the size of a computation event. Each computation event has the same size that is similar to the COSMIC FSM in which each data movement has the same magnitude. We anticipate that a system will be made up of a sequence of events. Therefore, the size of a functional user requirement is the total number of computation events it contains. Consequently, the size of the software under measurement will be equal to the sum of the size of its functional requirements.

5.2.4. Discussion about COSMIC FSM and Event-based measurement model

Even though COSMIC FSM and Event-based size measurement have different philosophies, following event types and COSMIC BFCs can be treated as they are having similarities:

- Triggering Entry Data Movement of COSMIC FSM and System Boundary Event,
- Read Data Movement of COSMIC FSM and Retrieval Event,
- Write Data Movement of COSMIC FSM and Record Event,
- Exit Data Movement of COSMIC FSM and Display Event.

However, the difference between data movement and event is that, according to the COSMIC FSM method, each data movement mentioned above are transferring a data group belonging to the same object of interest (COSMIC, 2017b). On the other hand, in event-based definitions, there is not a data-centric philosophy. The measurer does not have

to know and think about the data structure and model of the software. In addition, in event-based method, we offer additional events such as Calculation/Processing Event, Decision Event and Communication Event which are not considered in COSMIC FSM.

5.3.Summary of the Chapter

In this chapter, based on the insights gained from the previous literature searches and case studies, the final version of the model is presented. In this model, different types of events and how they will be interpreted as size measurement components are given in details. In the next chapter, the results regarding the evaluation of the final version of the model is explained in detail.

CHAPTER 6

EVALUATION CASE STUDIES

6.1. Case Study Design

In the evaluation case studies, the goal is to validate the success and reliability of the proposed event-based size measurement model. The relevance of events in size measurement is aimed to be assessed through correlation and regression analyses. For case selection, an important criterion would be the inclusion of projects where contemporary new generation architectural styles are used as much as possible. Case selection criteria comprise projects having requirements documented along with their effort values. Maturity of the documentation has an important impact on the reliability of measurement. Therefore, another criterion is choosing projects that are documented in a way that are understandable by an external measurer. Lastly, choosing projects where different SDLC approaches such as Waterfall and Agile methodologies are employed can be a positive attribute considering the high adoption of both approaches in software industry.

Accordingly, we formulated the following research questions (RQs):

RQ1. Are events relevant as a size measure?
RQ2. How does event-based size measure perform to estimate effort compared to other FSM methods?

6.1.1. Case Descriptions

Considering case selection criteria and research questions, 3 projects from a large organization that will be called as Organization X in the rest of this thesis and 10 projects from another large software development company that will be called as Organization Y are aimed to be chosen for the case studies. These organizations will be named as Organization X and Organization Y respectively in the rest of this thesis.

In the Organization X, requirements are documented in the form of use cases in a fully-dressed format with detailed descriptions. Effort spent for each use cases are kept by the team members as development and test effort. In addition, COSMIC FSM method is being

adopted in the organization for software size measurement for several years. In other words, the organization has sufficient experience in functional size measurement. Another characteristic of the organization is its experience on Agile software development and the case projects are selected from the teams which adopts Agile methodology.

First project is an application that aims to facilitate the decision related tasks of managers by determining the competencies of the employees based on their decision abilities with respect to the cases that are assigned to them. The application has two types of users such as admin and end-user. According to the analyst, using this application, the job of assessment of employees is decreased from one month to one hour. The development team is made up of four people. It is a responsive application which can be accessed via desktop and mobile devices. It is made up of following modules: authentication and user management which communicates with the master module, video streaming platform and Simple Mail Transfer Protocol (SMTP); admin front-end, a responsive end-user front-end, and reporting; web-services, business layer and data layer. For the front-end development Angular and CSS were used where for the back-end development Hibernate, Spring Framework, Spring Boot and Maven were utilized. As the database management system, ORACLE Database Management Systems was used. It includes 47 use cases with detailed description and alternative flows and related screens for some of these use cases. These use cases are specified in 12 analysis documents. A total of 528 person-day have been spent on the realization of these use cases.

Second project is related with a self-learning system to be used within the organization. In this web application, admin can create content, grant access, and users can do homework etc. We acquired 5 analysis documents describing different small modules of the project; which can be considered as five mini projects. A total of 29 use cases having a total of 780,5 person-day effort with detailed description and alternative flows are presented in the documents. These use cases were documented in fully dressed format and there exists mockup screens facilitating the understanding of the processes. As we observed usually in contemporary projects, there were no data model in the project. The development team is made up of eight people.

Third project is different from the previous ones by being a microservice. This microservice is responsible for integrating different payment modules in a large system. In addition to the human user of the system, other systems are also acting as functional users. It was developed using Spring Boot and iFrame technologies. The microservice is made up of 8 use cases documented as fully-dressed format. In addition, the analyst prepared sample screens of the system, list of parameters and data attributes. A total of 158 person-day were spent in this microservice. The development team is composed of three people.

Organization Y is a large software development company. 10 projects developed based on various architecture were selected for this case study. In Table 14, the details of these project in terms platform, requirement specification format, architecture used and the effort spent for the development are presented.

Table 14 Measurement Results of Projects of Organization Y

Project ID	Project Category	Requirement Specification Format	Architecture	Development Effort
1	Web	Use case scenarios	Client-server, Standalone, Restful	696
2	Web	Use case scenarios and screens	SOA, standalone, client-server (There are batch and UI applications)	354
3	Mobil	Use case scenarios and screens	Web Service	330
4	Unspecified	Use cases	Unspecified	140
5	Web	Use case scenarios and screens	Unspecified but there exists service integration	298
6	Unspecified	Work flow	Service-Oriented	332
7	Web	Use case scenarios and screens	Web application - JSF	224
8	Web	Use case scenarios	MVC	410
9	Web	Use case scenarios and screens	Client-Server Web Based Application	326
10	Mobil	Use case scenarios and screens	Web Service	430

6.2. Conduct of the Case Studies in Organization X

During the conduct of case studies, same procedure is employed in all the projects. Each use case is modelled with eEPC, distinct computation events are identified according to the model proposed. Care is taken to reduce the assumptions as much as possible in order to not create a bias. Consequently, in three projects, a total of 84 use cases are studied and checked whether it can be possible to measure their size using the event-based measurement model. The CFP values presented in Table 15, Table 16 and Table 17 were obtained as a result of COSMIC FSM measurement that were conducted by the analysts of the projects who have comprehensive knowledge about the projects. Measurement results of each project are presented under their own sub-sections as follows.

6.2.1. Measurement Results of Project 1

For Project 1, the case study was conducted on 47 use cases which were present in 12 analysis documents. In Table 15, analysis document-wise project data collected in terms of number of use cases, functional size as CFP, number of events obtained as a result of measurement, development effort, test effort and total effort as person-day are presented.

Table 15 Data collected in Project 1

Analysis Document ID	Number of use cases	CFP	Number of Events	Development Effort (person-day)	Test Effort (person-day)	Total Effort (person-day)
Doc-1	4	31	48	27	10	37
Doc-2	2	16	22	14	5	19
Doc-3	5	23	27	19,75	8,25	28
Doc-4	5	66	82	72	38	110
Doc-5	3	20	29	17	6,5	23,5
Doc-6	3	30	39	21	11	32
Doc-7	3	36	27	28	11	39
Doc-8	3	16	27	11	6	17
Doc-9	9	64	60	63	27,5	90,5
Doc-10	4	70	56	58,25	19,25	77,5
Doc-11	2	11	19	9	3,5	12,5
Doc-12	4	22	38	29	13	42
Total	47	405	474	369	159	528

In project 1, it is seen that in three of the analysis documents, namely, Doc-3, Doc-7 and Doc-9, there were repeating use cases. In addition, it is observed that the effort spent to them were sometimes the same and sometimes varying. When we consulted this issue to the analyst of the company, it is just suggested to exclude the redundant ones. We saw that Doc 9 comprising all the use case of Doc 3 and Doc 7. Therefore, Doc3 and Doc 7 are excluded from the analysis. Table 18 presents the descriptive statistics after this elimination is done.

6.2.2. Measurement Results of Project 2

In Project 2, five analysis documents are observed within the scope of this case study. 1 of the use case have design and technological details, having one more low level details compared to the others. Therefore, this use case has been excluded in the measurement and the results are presented over 29 use cases. Project details in terms of number of use cases, CFP values, number of events, development, test and total effort in person-day are presented in Table 16.

Table 16 Data collected in Project 2

Analysis Document ID	Number of use cases	CFP	Number of Events	Development Effort (person-day)	Test Effort (person-day)	Total Effort (person-day)
Doc-1	5	27	46	118	49	167
Doc-2	2	15	17	50	14	64
Doc-3	6	43	57	124	31	155
Doc-4	6	39	57	67	19,5	86,5
Doc-5	10	92	155	229	79	308
Total	29	216	332	588	192,5	780,5

6.2.3. *Measurement Results of Project 3*

As mentioned previously, third project that has been included in the evaluation case studies was a microservice. This microservice is made up of 8 use cases. Table 17 shows the CFP, number of events, and effort distribution in development and test and total effort spent in person-day.

Table 17 Data collected in Project 3

Use case ID	CFP	Number of events	Development Effort (person-day)	Test Effort (person-day)	Total Effort (person-day)
Use case -1	9	12	16	7	23
Use case -2	38	48	42	22	64
Use case -3	7	7	5	3	8
Use case -4	5	6	4	2	6
Use case -5	5	6	5	3	8
Use case -6	5	9	6	3	9
Use case -7	12	11	14	8	22
Use case -8	6	10	12	6	18
Total	87	109	104	54	158

6.2.4. *Analysis of the results*

The analyses of the results are carried out in three major steps: (1) Investigation of a relationship between effort and software size in terms number of events and between effort and software size in CFP by using Pearson Correlation Analyses; (2) building effort estimation models in each project using regression analyses; and (3) assessing the accuracy of the effort prediction performance of these models. These analyses results are presented in the following paragraphs for each project.

In Table 18, the descriptive statistics of the projects such as number of cases (n), number of uses cases, total effort spent (person-day) in projects, number of events, size in CFP, minimum, maximum, mean and median of the effort values.

Table 18 Descriptive Statistics of Projects

Project	n	Number of use cases	Total effort (person-day)	Total Event	Total CFP	min effort (person-day)	max effort (person-day)	mean effort (person-day)	median effort (person-day)
Project 1	10	39	461	420	346	12,5	110	46,1	34,5
Project 2	5	29	780,5	332	216	64	308	156,1	155
Project 3	1	8	158	109	87	6	64	19,75	13,5

6.2.4.1. Investigation of the Relationship between size and effort

Scatter plots are used in order to investigate the relationship between two variables (Howell, 2012). As a first step, for each project, scatter plots depicting the relationship between event and effort and CFP and effort are created. In this case study, effort is the dependent variable which is shown in the y-axis, where number of events and CFP are independent variables shown in the x-axis in the scatter plots given in Figure 17.

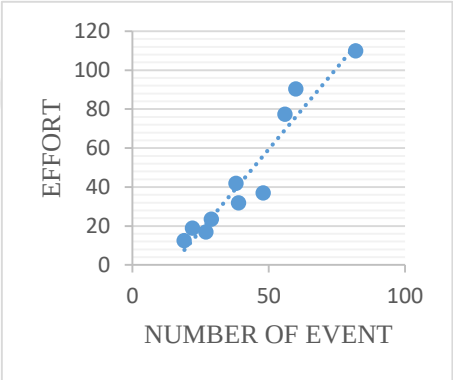
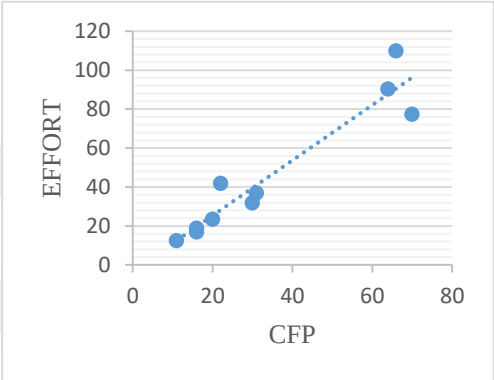
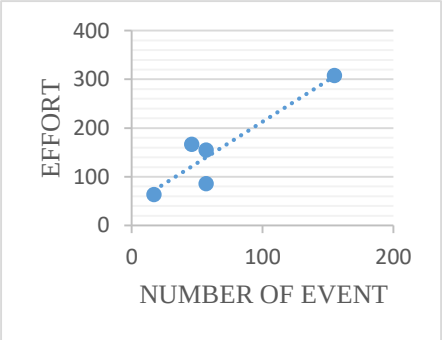
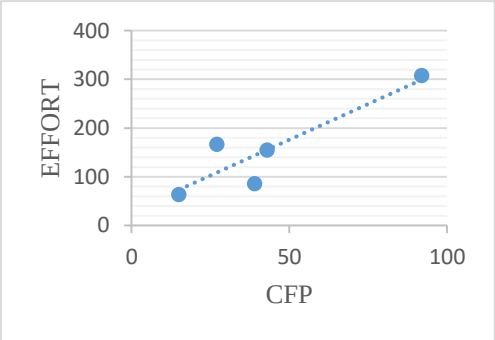
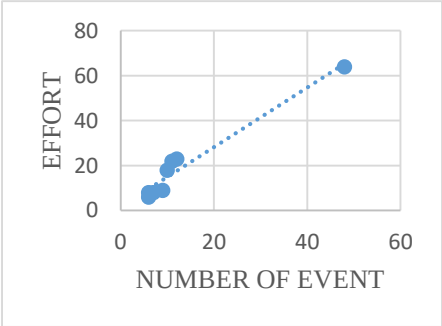
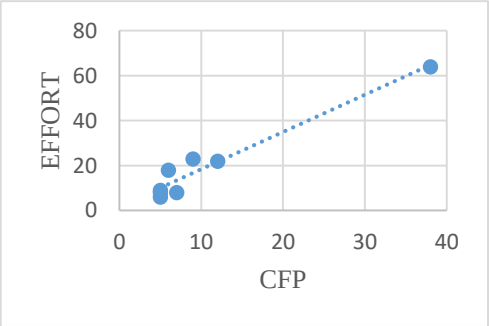
Project 1		
Project 2		
Project 3		

Figure 17 Scatter Plots of Project 1, 2 and 3

Then, correlation analyses were employed. As it can be seen in Table 19, we observed a very strong positive correlation (r>0,9) (Schober, Boer, & Schwarte, 2018) between event and effort and between CFP and effort which are statistically significant with p value is less than 0.05 in all projects.

Table 19 Correlation Analysis Results of Project 1, 2 and 3

Projects	**Correlation R**			
	Event and Effort	**P value**	**CFP and Effort**	**P value**
Project 1	0,96	,000	0,95	,000
Project 2	0,92	,024	0,9	,037
Project 3	0,98	,000	0,97	,000

Even though we obtain similar values for CFP-based and Event-based effort estimation models, we have to emphasize that CFP values were computed by the analysts having deep knowledge about the projects. For CFP measurement, the measurer needs at least a conceptual understanding of the data model, objects of interest and data attributes involved in the system. However, an independent measurer can obtain significant and compatible results using our method by taking the requirements as the input and without having knowledge about data model-related details.

6.2.4.2. Building effort estimation models

The high correlation values obtained from these two projects provided promising results and motivated us to conduct linear regression analyses (Humphrey, 1995). Linear regression analysis is a technique which is appropriate in cases having single independent variable and is extensively used in software effort estimation studies (Commeyne et al., 2016; Jørgensen, 2004a; Moulla & Abran, 2021). We conducted linear regression analysis using Microsoft Excel. The linear regression analyses result for all the projects are given in Table 20.

It is seen that, events can explain 92%**,** 86%**,** 95% of the variations in effort in Project 1, Project 2 and Project 3 respectively. The regression models built with events are statistically significant with Significance F less than 0.05 in all the projects.

On the other hand, when the results of regression analyses with CFP and effort are observed, it is seen that effort prediction models built with CFP are also statistically significant with Significance F value that are less than 0.05 in all the projects. CFP values are able to explain 90%**,** 81%**,** 95% of the variations in effort in Project 1, Project 2 and Project 3 respectively.

When these two independent size measures (event and CFP) are compared using the coefficient of determination, R^2**,** it can be concluded that in Project 1 and Project 2 event have slightly higher R^2 values than CFP and hence slightly higher explaining capability than CFP where in Project 3 they are the same.

Table 20 Linear Regression Results

	Event and Effort			**CFP and Effort**		
Project	**R^2**	**Sig. F**	**Regression Equation**	**R^2**	**Sig. F**	**Regression Equation**
Project 1	0,92	0,000	Effort= Event*1,6678-23,948	0,90	0,000	Effort= CFP*1,4166-2,9155
Project 2	0,86	0,02	Effort= Event*1,6953+43,532	0,81	0,04	Effort= CFP*2,923+29,556
Project 3	0,95	0,002	Effort= Event*1,3264+1,6776	0,95	0,000	Effort= CFP*1,6579+1,7199

6.2.4.3. Assessing the accuracy of the effort estimation

The third step in the analysis is assessing the accuracies of the predictions. For this purpose, metrics such as MRE, MMRE, MdMRE and PRED(30) are used. Before conducting the accuracy assessment of the effort estimation models produced using event and CFP; the projects scatter plots given in Figure 17 are observed. It is seen that in Project 3, one use case having a size-effort value that has a larger horizontal and vertical gap when compared to other cluster of the data points. It has an effort value (64 person-day) which is comparatively larger than the remaining use cases and its size (38 CFP and 48 events) is higher than other use cases.

This use case was further investigated and it is seen that different from the other use cases which are more self-contained, it comprises communications with banking system and it includes considerably more conditions and checks than other use cases and can be excluded as an outlier. Consequently, the regression models are renewed after the removal of the outlier value for Project 3. When this outlier use case is excluded, we obtained the scatter plot with regression equations given in Figure 18.

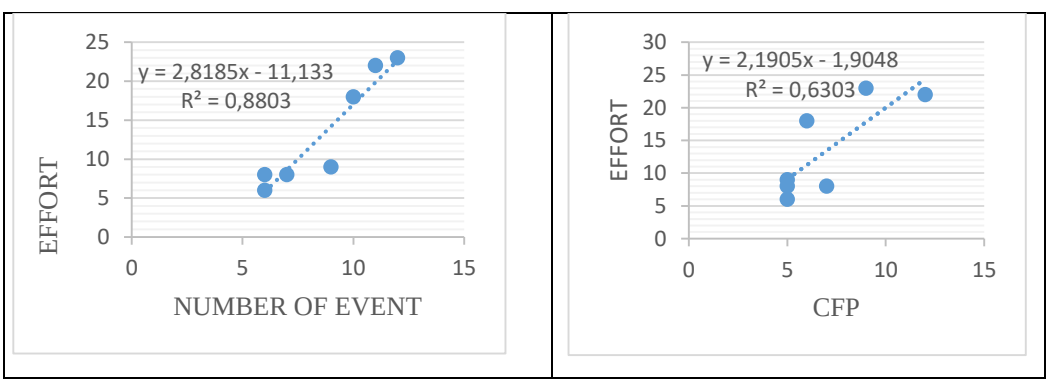


Figure 18 Scatter plots of Project 3 after excluding the outlier

After excluding the outlier, it is seen that the correlation coefficient become R=0,94 with p value=0,002 for event and effort; and R=0,79 with p value=0,033 for CFP and effort. It signifies that event correlates better with effort than CFP correlates with the effort. As a result of the regression analysis, it is seen that coefficient of determination of Events (R^2=0,88) is higher than that of CFP (R^2= 0,63). Meaning that 88% of the variation in effort can be explained with event where CFP can explain 63% of this change. Regression equations are presented in Table 21. This result is specifically important within the scope of this thesis, where we are in the pursuit of a new size measurement approach for new generation projects. This project is a microservice and event-based size measurement model outperform the functional size measurement method.

Table 21 Project 3 Regression Results after excluding the outlier

	Event and Effort			**CFP and Effort**		
Project	**R^2**	**Sig. F**	**Regression Equation**	**R^2**	**Sig F**	**Regression Equation**
Project 3	0,88	0,002	Effort=Event*2,8185-11,133	0,63	0,03	Effort=CFP*2,1905-1,9048

MRE, MMRE, MdMRE and PRED analysis were conducted for all the projects. Table 22 presents the results of these analysis.

Table 22 Accuracy Assessment

Project	**MMRE event**	**MMRE CFP**	**MdMRE event**	**MdMRE CFP**	**Pred(30) event**	**Pred(30) CFP**
Project 1	0,21	0,14	0,20	0,13	0,7	0,9
Project 2	0,22	0,24	0,13	0,15	0,8	0,6
Project 3	0,16	0,29	0,08	0,23	0,71	0,57

According to the results given in Table 22, event-based effort estimation models yield acceptable PRED(30) values (MacDonell & Gray, 1998) where PRED(30) values for CFP based model for Project 2 and Project 3 is slightly less than the threshold. All the MMRE, MdMRE values for models based on both event and CFP are within the acceptable range (Hastings & Sajeev, 2001).

It can be further concluded that error values for CFP is slightly less than event in Project 1. However, in Project 2 and Project 3 which is a microservice, there is a better error value with size measurement using events and it also has better prediction accuracy.

To validate the accuracy of the results, Leave-One-Out-Cross Validation (LOOCV) technique is suggested to be used in software effort estimation studies (Kocaguneli & Menzies, 2013). It is a special version of the k-fold cross validation with k=n (Sigweni, Shepperd, & Turchi, 2016), where n is the total number of cases in the given dataset. In this technique, the procedure is as follows: dataset is divided into two subsets such as

training set and test set. n times, a distinct single case is removed from the dataset as being the test set and tested with the model created using n-1 cases and then returned back to the training set (Sigweni et al., 2016). This technique has been developed to overcome the model overfitting problem (Corazza, Di Martino, Ferrucci, Gravino, & Mendes, 2011). LOOCV was conducted in each project and the MMRE, MdMRE and PRED values are presented in Table 23.

Table 23 Model Validation using LOOCV

Project	MMRE event	MMRE CFP	MdMRE event	MdMRE CFP	Pred(30) event	Pred(30) CFP
Project 1	0,26	0,18	0,24	0,16	0,6	0,8
Project 2	0,31	0,37	0,23	0,26	0,6	0,6
Project 3	0,21	0,39	0,1	0,41	0,71	0,29

As a result of LOOCV, it is observed that all the error related values are within the acceptable range MMRE value for event and CFP is within the acceptable range (Hastings & Sajeev, 2001). As expected, in Project 3, whose architecture is a microservice, the PRED (30) = 0,71 for event-based effort estimation model is within the acceptable range and it significantly outperforms CFP-based effort estimation which has PRED (30) =0,29.

6.3. Conduct of the Case Studies in Organization Y

In Organization Y, the same procedure is followed: use cases in the projects are modelled using eEPC to identify computation events according to the model proposed and CFP values are obtained using COSMIC FSM method. The measurement results are presented in sub-section 6.3.1.

6.3.1. Measurement Results

Within the scope of the case study, in Organization Y, 10 projects are measured using event-based size measurement method proposed in this thesis and using COSMIC FSM method. The analysis documents of each project are taken as input for measurement. In Table 24, the projects, their corresponding size values in terms of events and CFP and the development effort spent in these projects are presented.

Table 24 Measurement Results

Project ID	Number of Events	CFP	Development Effort (person-day)
1	228	166	696
2	124	170	354
3	96	434	330
4	38	258	140
5	74	168	298
6	106	50	332
7	24	62	224
8	152	108	410
9	70	110	326
10	174	120	430
Total	1086	1646	3540

6.3.2. Analysis of the Results

We carried out same three activities during the analysis of the results of the projects of Organization Y: (1) Investigation of a relationship between effort and software size in terms number of events and between effort and software size in CFP by using Pearson Correlation Analyses; (2) building effort estimation models with these projects using regression analyses; and (3) assessing the accuracy of the effort prediction performance of these models.

6.3.2.1. Investigation of the Relationship between size and effort

To investigate the relationship between size and effort we plot the scatter diagram (Howell, 2012). We first investigated the relationship between events and effort; then, with CFP and effort. The scatter plots are given in Figure 19. In the given scatter plots x axis denotes the size value where y axis represents the effort.

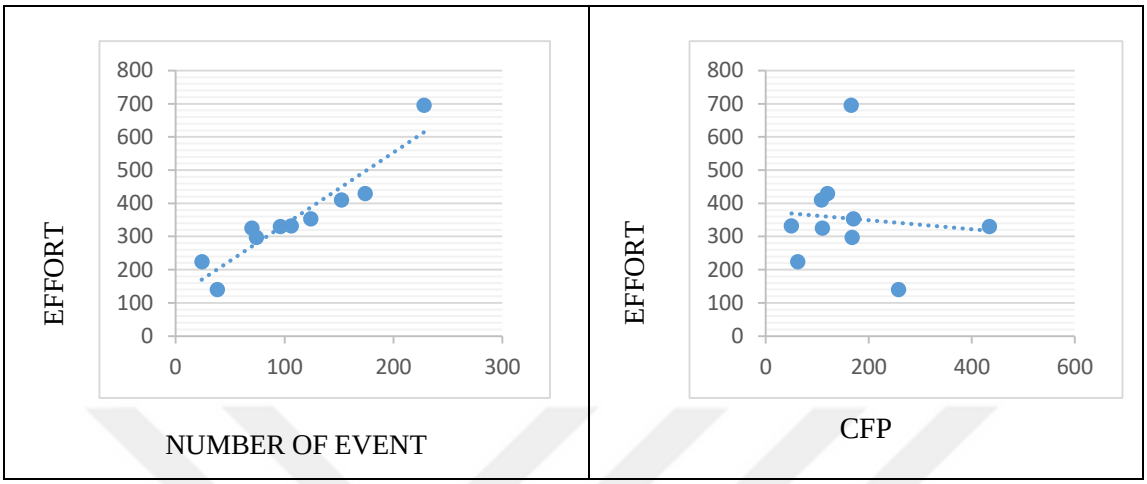


Figure 19 Scatter plots of Projects of Organization Y

Following the visualization of the relationship between size (in terms of events and CFP) and effort, correlation analyses were conducted. As it is presented in Table 25, there exists a very strong positive correlation (r>0,9) (Schober et al., 2018) between event and effort which are statistically significant with p value is less than 0.05. However, the correlation coefficient r is -0,10 between CFP and effort and the correlation analysis is not significant.

Table 25 Correlation Analysis Results

Projects	**Correlation R**			
	Event and Effort	**P value**	**CFP and Effort**	**P value**
	0,94	0,000	-0,10	0,776

6.3.2.2.Building effort estimation models

Motivating from the high correlation value between event and effort, we carried out linear regression analyses (Humphrey, 1995). Results of linear regression analyses are given in Table 26. Since the correlation between CFP and effort value is not allowing to conduct regression analysis we did not created a CFP based regression model.

Table 26 Linear Regression Results

	Event and Effort		
Project	**R^2**	**Significance F**	**Regression Equation**
	0,88	0,000	Effort= Event*2,1802+117,23

Based on the regression model, it can be said that 88% of the variations in effort can be explained with events and this model is significant with Significance F less than 0.05 in all the projects.

6.3.2.3. Assessing the accuracy of the effort estimation

The accuracy of the effort estimation is assessed using MRE, MMRE, MdMRE and PRED (30) metrics. In Table 27, the results of accuracy evaluation are presented.

Table 27 Accuracy Assessment

Project ID	Event	Actual ffort (person-day)	Estimated Effort (person-day)	MRE	MMRE	MdMRE	Pred30
1	228	696	614,3156	0,11736264	0,14	0,11	0,9
2	124	354	387,5748	0,09484407			
3	96	330	326,5292	0,01051758			
4	38	140	200,0776	0,42912571			
5	74	298	278,5648	0,06521879			
6	106	332	348,3312	0,04919036			
7	24	224	169,5548	0,24305893			
8	152	410	448,6204	0,0941961			
9	70	326	269,844	0,17225767			
10	174	430	496,5848	0,15484837			

From Table 28, we can observe that the prediction accuracy of the event based effort estimation model is acceptable with PRED(30) value 0,9 (MacDonell & Gray, 1998). Margins of errors that are assessed using MMRE (0,14) and MdMRE (0,11) metrics are also within acceptable range (Hastings & Sajeev, 2001).

To assess the validity of the accuracy of event-based effort estimation model, we carried out Leave-One-Out-Cross Validation (LOOCV) technique. Table 28 presents the model validation results using LOOCV.

Table 28 Model Validation using LOOCV

Project ID	Event	Actual Effort (person-day)	Estimated Effort (person-day)	MRE	MMRE	MdMRE	PRED30
10	174	430	515,3632	0,19851907	0,19	0,15	0,8
9	70	326	260,56	0,200736196			
8	152	410	455,5864	0,111186341			
7	24	224	146,139	0,34759375			
6	106	332	350,1534	0,054678916			
5	74	298	275,568	0,075275168			
4	38	140	219,0306	0,564504286			
3	96	330	326,1336	0,011716364			
2	124	354	391,5836	0,106168362			
1	228	696	532,6568	0,234688506			

Results of LOOCV also supports the accuracy results obtained previously. All the errors are within the acceptable range (Hastings & Sajeev, 2001) with MMRE=0,19, MdMRE=0,15 and the prediction accuracy is also acceptable with a PRED30 value which is equal to 0,8. As mentioned in case description section, these projects are developed based on the comparatively novel architectures and uses services. Therefore, these results motivate the success of the model proposed in this thesis to define the project size in terms of events.

6.3.3. Implications from the Evaluation Case Study

As a result of the analysis of the measurement, following implications can be as drawn:

COSMIC FSM measurements are conducted by the project analyst in Organization X who has deep knowledge about the project. It means that the measurement model suggested in this study provides similar results with COSMIC FSM even though the measurements with the new model are conducted by an independent measurer. If we were performing the COSMIC FSM, since we are not working in this project, we will have the role of independent measurer and the measurement would be complicated because we could not know all the data groups, object of interests (OOIs) since in the projects there is again no data model.

We observed this situation in Organization Y. In Organization Y, COSMIC FSM was performed by an independent measurer who is not working in the projects. Due to the

requirements' content detail levels, measuring the size of projects using COSMIC FSM requires a lot assumption due to the lack of data models. This complicates the measurement. On the other hand, in projects of Organization Y, by being an independent measurer we could perform the event modelling from the fully-dressed use cases in a convenient manner because:

- There is no need to create a data model as it is required for COSMIC FSM
- There is no need to identify OOIs as it is required for COSMIC FSM
- Availability of use case descriptions ease the identification of the events

This shows the practicability of our model, in other words it can be applicable by an independent measurer and produce promising results. In addition, if process modeling is already in use in the organizations, measuring the project size can be very trivial and identifying the events are easier than identifying COSMIC FSM's base functional components even there is no data description.

The architecture of the third project is a microservice, our event-based size correlates well with the effort and more than CFP does. In addition, after the measurement, when we communicated with the business analyst in Organization X, it has been understood that effort values may not always be accurate in terms of person-day spent for a specific analysis document. There can be cases where the developers are busy with developing some undocumented tasks in addition to the use cases written in the analysis documents within a sprint. Therefore, this fact can be the reason behind low PRED (30) values in prediction models. However, since the correlation between the size obtained in the proposed model and the effort spent are high in each case and both correlation results and MMRE results are similar to the ones obtained using COSMIC FSM method, it may not be wrong to say that the low accuracy may be due to the effort values where effort related problems is an already mentioned issue in the literature (Özkaya et al., 2011).

6.4. Summary of the Chapter

In this chapter, the final version of the model is evaluated in projects of two organizations within the scope of case studies. The results are presented in three steps: relationships between size measures in terms of CFP and events; and the effort; building effort estimation models using size measures as independent variables and assessing the accuracies of these models. In the next chapter, the limitations and threats to validity of this thesis are presented.

CHAPTER 7

LIMITATIONS AND VALIDITY THREATS

Empirical research can face several threats (Feldt & Magazinius, 2010). In the literature, there exists a set of validity threats in different worldviews, which are categorized and analyzed by (Petersen & Gencel, 2013). In this thesis multiple case studies are conducted to exploration of the problem, solution and for the evaluation of the results. For this aim, we adopt the validity threats presented by (Yin, 2017) for case study research. (Yin, 2017) reports four types of validity threats that can be relevant in case study research namely, construct, internal, external validity and reliability and some tactics to minimize these threats. In this section, possible threats to validity of multiple case studies and how we addressed them in this thesis are discussed.

To ensure construct validity, (Yin, 2017) suggests the utilization of multiple source of evidence. We adopted this tactic by gathering software development projects from five software development company. The projects have been chosen from different domains, architectures and considering different format and detail level of requirements specification as much as possible.

External validity refers to the generalizability of the results and to achieve external validity (Yin, 2017) suggests to separate the theory and replication by using single-case studies for the former and multiple-case studies for the latter. In our research, to ensure external validity, different multiple case studies are conducted. Problem identification, the exploration of the solution; and validation of the solution are conducted on different projects. The conduct of case studies in real software organizations and in different types of projects increase the generalizability of the results; hence can decrease the effect of this threat.

However, there exists some limitations concerning the applicability of case selection procedure. Since this study is conducted with real software development projects', main limitation in this context was the collection of data. Data collection related limitation

can be discussed in three dimensions: privacy concerns of organizations (Malhotra, 2016), certain software engineers' lack of documentation tendency, and lack of effort entries.

First, due to privacy reasons, gathering data from the organizations was challenging. Second, the measurements and modeling conducted in the present study are based on everything documented. As mentioned in the literature, practitioners have an inclination to keep the documentation in a minimal fashion (Forward & Lethbridge, 2002; Inayat, Salim, Marczak, Daneva, & Shamshirband, 2015) where the available documentation is rarely updated (Forward & Lethbridge, 2002). The same situation was also faced in this present study. As a consequence, in the projects gathered, it is observed that both the quantity of requirements documented, and their contents are often not completely and perfectly recorded. For example, as observed in Exploratory Case Study 1 (Hacaloglu & Demirors, 2019), some Agile practitioners have the tendency to document social and technical tasks in the form of user stories and estimate them in the similar way as they do while estimating functional user requirements.

Moreover, especially in projects where Agile practices are adopted, it is seen that the user story content diverges from the one imposed by the literature, which points to the discrepancy between the industry and the academia. In addition, some of the documented requirements did not convey useful and understandable information for measurement. This fact is compatible with the interview study by Lethbridge, Singer, & Forward (2003) who claim that significant portion of documentation is untrustworthy. Observed requirements had varying abstraction levels from very high to very low including technical tasks. Identifying functionalities from the requirement set was not straight-forward in every cases.

Third limitation concerning the collected data is the lack of effort values to implement the requirements. Practitioners in the case organizations have the tendency of entering imprecise values for actual effort spent or not recording them at all. In addition, as specified in the evaluation case study, even though software developers were recording the effort spent, these effort values do not correspond purely on the use case mentioned in the analysis document, but some other tasks injected during development process were said to be the part of the efforts. At this point, it is important to emphasize that in software engineering industry, effort collection is an already mentioned challenge which has been the subject of several studies in the literature (Özkaya et al., 2011; VanHilst et al., 2011). For the projects in which we were able to reach comparatively mature requirements sets, since the related efforts were missing, these projects were excluded in the study. For this reason, during the analysis of results, we had circumstances in which carried either task-based (Commeyne et al., 2016; Ertaban et al., 2017; Hacaloglu et al., 2019) or analysis document based analyses.

A data collection related limitation is observed from the statistical analysis perspective. One threat can be violation of normal distribution and sample size assumptions of regression analysis (Wohlin et al., 2012). However, in software engineering domain this

type of circumstances is observed for example (Hastings & Sajeev, 2001) due to the nature of research setting which is practical and hard to obtain large samples. In spite of small sample size in regression analysis, the tendency in terms of correlation and acceptable MMRE, MdMRE values can give idea that with the available projects an organization can use event-based model can be used as an alternative to COSMIC FSM.

Reliability is related with to what extent the study is repeatable in terms of data collection and obtaining similar results (Yin, 2017). To ensure reliability, the tactic suggested by (Yin, 2017) is to use a protocol and develop case study database. In each case study, we followed a case study protocol where we have defined case selection criteria. At this point a threat to validity can be the reliability of measures (Wohlin et al., 2012) because both COSMIC FSM method and event-based measurement model presented in this thesis require some degree of human judgment. However, COSMIC FSM method is known as a highly repeatable functional size measurement method having clearly defined manuals. In addition, the event-based size measurement model suggested in this thesis is described in detail in order to be usable in a repeatable manner for every measurer. To decrease the bias in the interpretation of the data collected, during the measurement, use-cases/user stories artifacts which are not clear enough were eliminated or consulted when it is possible.

CHAPTER 8

CONCLUSIONS AND FUTURE WORKS

8.1. Conclusions

The current landscape of software development demonstrates an architectural paradigm shift towards distribution, modularization and loose coupling (Mazzara et al., 2020). An obvious reflection is visible on the adoption of technologies e.g. Cloud Computing, Microservices, Distributed Systems which encourage loosely coupled methods for data and usage of events. In addition, there exists minimal requirements specification in projects where Agile methods are adopted. Both situations pose challenges in the usage of functional size measurement methods and necessitate the emergence of new size measurement approaches.

Events by forming an indispensable element of these systems are used in this study to define the size of contemporary software. To the best of our knowledge, there is no study in the literature discussing and reporting the challenging issues of new generation software regarding the size measurement perspective within the scope of a case study. In this thesis, we presented results of multiple case studies on the usage of events as base counting units.

While starting the Exploratory Case Study 1, we aimed to identify the potential challenges of functional size measurement. As a result of this case study, it is seen that in Agile projects the applicability of functional size measurement such as COSMIC FSM suffer from assumptions due to the lack of maturity of documentation in terms of identifying COSMIC FSM related measurement components such as object of interests, data groups, functional processes. This case study also revealed the fact that conventional data analysis practices such as relational data modeling etc. – suggested for COSMIC FSM (COSMIC, 2017a) too- are not adopted in the observed projects. On the other hand, considering the distributed nature of contemporary architectures, we defined our research problem as "how a size measurement model should be for contemporary software architectures?"

As mentioned previously, considering the important place of event in today's architectures, in Exploratory Case Study 2, we aimed to assess the relevance of events extracted from the requirements as a size measure by forming an initial model. We obtained promising results concerning the correlation between events and effort. Then, we deepened the literature search regarding the different event types and within the scope of the Exploratory Case Study 3, we practically tried to identify them from the requirements. We decided to update the model to consider events according to their sources. We saw that events are still correlating with the effort. In addition, we saw that system events are better correlating with the effort. As a result, we realized the criticality of defining the right abstraction level of events, especially in order to resolve possible confusions on the modeler's perspective; a need for a systematic and repeatable measurement model; a correlation between the events and the actual effort spent on these projects. With this model we envision that organizations can measure the event-based size of the software without being dependent on the data model.

In Evaluation Case Study, we evaluated our model in two different organizations with a total of thirteen projects. For three projects of Organization X, it is seen that both events and CFP correlates well with effort. When we observe the success of both size measures in effort estimation with LOOCV, we can say that except Project 1, MMRE and MdMRE values of effort estimation model created with events are less than the ones created with CFP: for Project 2 MMRE (event)=0,31 where MMRE (CFP)=0,37; and MdMRE (event)=0,23 and MdMRE (CFP)=0,26. For Project 3 which is specifically important by being a microservice, it is seen that MMRE (event)=0,21 and MMRE (CFP)=0,39 and MdMRE (event)=0,1 and MdMRE (CFP)=0,41. Prediction accuracies are also evaluated with PRED 30 metric. It is seen that in Project 1 CFP produce better results than events and in Project 2 they have same value as 0,6. However in Project 3 events outperform CFP with PRED30 (event)=0,71 where PRED 30 (CFP)=0,29. In Organization Y, with 10 projects it is seen that events and effort correlates with R=0,94 where no correlation was observed with CFP and effort. In addition, event based effort estimation yield successful error and prediction accuracy values with MMRE(event)=0,19; MdMRE (event)=0,15 and PRED30(event)=0,8.

Using this model, organizations can create early effort estimations with regression analysis, using functional requirements at the initial stages of software development lifecycle. The model does not involve a design related decision such as the creation of a data model.

8.2. Future Works

As the future work, we aim to apply the measurement model on more projects and evaluate the success of the system by creating effort estimation models and by comparing the deviation of predicted efforts from the actual ones and also by comparing with other measurement methods' performance. Since events are grouped under distinct categories,

the contribution of each different event type into effort estimation can also be assessed using multiple regression analysis. For this aim, we also plan to conduct multiple regression analysis with different types of events to estimate effort in a more accurate way.

Another issue to observe can be the efficiency of the measurement. In studies planned to be conducted in the future, time of measurement using proposed model can be kept and be compared with available functional size measurement methods to identify the most efficient one. In addition, the usage of a new software size measurement method can also be a baseline for the automation of software size measurement related studies for further research purposes.

REFERENCES

Aarab, Z., Saidi, R., & Rahmani, M. D. (2016). Event-driven modeling for context-aware information systems. In 2016 IEEE/ACS 13th International Conference of Computer Systems and Applications (AICCSA) (pp. 1–8). IEEE.

Abeysinghe, A. (2016). Scope Versus Size: a Pragmatic Approach to Microservice Architecture.

Abran, A. (2015). Software project estimation: The fundamentals for providing high quality information to decision makers. John Wiley & Sons.

Abran, A., Desharnais, J.-M., Zarour, M., & Demirörs, O. (2015). Productivity-based software estimation models and process improvement: an empirical study. Int. J. Adv. Softw, 8(1&2), 103–114.

Abran, A., Symons, C., Ebert, C., Vogelezang, F., & Soubra, H. (2016). Measurement of software size: Contributions of cosmic to estimation improvements. In The International Training Symposium, At Marriott Bristol, United Kingdom (pp. 259–267).

Afshar, S., Ralph, N., Xu, Y., Tapson, J., Schaik, A. van, & Cohen, G. (2020). Event-based feature extraction using adaptive selection thresholds. Sensors, 20(6), 1600.

Aken, J. E. van. (2004). Management research based on the paradigm of the design sciences: the quest for field-tested and grounded technological rules. Journal of Management Studies, 41(2), 219–246.

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

Ali, M., Shaikh, Z., & Ali, E. (2015). Estimation of Project Size Using User Stories. In International Conference on Recent Advances in Computer Systems. Atlantis Press.

Amjad, A., Azam, F., Anwar, M. W., Butt, W. H., & Rashid, M. (2018). Event-driven process chain for modeling and verification of business requirements–a systematic literature review. IEEE Access, 6, 9027–9048.

Andriole, S. J. (2017). The death of big software. Communications of the ACM, 60(12), 29–32.

Asik, T., & Selcuk, Y. E. (2017). Policy enforcement upon software based on microservice architecture. In 2017 IEEE 15th International Conference on Software Engineering Research, Management and Applications (SERA) (pp. 283–287). https://doi.org/10.1109/SERA.2017.7965739

Aslam, W., Ijaz, F., Lali, M. I. U., & Mehmood, W. (2017). Risk Aware and Quality Enriched Effort Estimation for Mobile Applications in Distributed Agile Software Development. J. Inf. Sci. Eng., 33(6), 1481–1500.

Babar, M. A., & Zhang, H. (2009). Systematic literature reviews in software engineering: Preliminary results from interviews with researchers. In 2009 3rd International Symposium on Empirical Software Engineering and Measurement (pp. 346–355). IEEE.

Bahsoon, R., Ali, N., Heisel, M., Maxim, B., & Mistrik, I. (2017). Introduction. Software Architecture for Cloud and Big Data: An Open Quest for the Architecturally Significant Requirements. In Software Architecture for Big Data and the Cloud (pp. 1–10). Elsevier.

Banerjee, S., & Sarkar, A. (2016). Modeling NoSQL databases: from conceptual to logical level design. In 3rd International Conference Applications and Innovations in Mobile Computing (AIMoC 2016), Kolkata, India, February (pp. 10–12).

Barroca, L., Sharp, H., Salah, D., Taylor, K., & Gregory, P. (2018). Bridging the gap between research and agile practice: an evolutionary model. International Journal of System Assurance Engineering and Management, 9(2), 323–334.

Berardi, E., Buglione, L., Cuadrado-Collego, J., Desharnais, J.-M., Gencel, C., Lesterhuis, A., … Trudel, S. (2011). Guideline for the use of COSMIC FSM to manage agile projects. Common Software Measurement International Consortium.

Berardi, E., & Santillo, L. (2010). COSMIC-based project management in agile software development and Mapping onto related CMMI-DEV process areas. In International Workshop on Software Measurement-IWSM 2010. Citeseer.

Bhalerao, S., & Ingle, M. (2009). Incorporating Vital Factors in Agile Estimation through Algorithmic Method. IJCSA, 6(1), 85–97.

Bhalerao, S., & Ingle, M. (2011). Agile estimation using caea: A comparative study of agile projects. In International Conference on Computer Engineering and Application (IPCSIT) (pp. 76–82).

Biolchini, J., Mian, P. G., Natali, A. C. C., & Travassos, G. H. (2005). Systematic review in software engineering. System Engineering and Computer Science Department COPPE/UFRJ, Technical Report ES, 679(05), 45.

Bögl, A., Kobler, M., & Schrefl, M. (2008). Knowledge Acquisition from EPC Models for Extraction of Process Patterns in Engineering Domains. In Multikonferenz Wirtschaftsinformatik.

Boner, J. (2017). How Events Are Reshaping Modern Systems.

Bonér, J., Farley, D., Kuhn, R., & Thompson, M. (2014). The reactive manifesto. Dosegljivo: Http://Www. Reactivemanifesto. Org/.[Dostopano: 21. 08. 2017].

Buglione, L., & Trudel, S. (2010). Guideline for sizing agile projects with COSMIC. Proceedings of the IWSM/MetriKon/Mensura, Stuttgart, Germany.

Čelar, S., Šeremet, Ž., Marušić, Ž., & Turić, M. (2013). Using of Web Objects method in agile web software projects. In 2013 21st Telecommunications Forum Telfor (TELFOR) (pp. 873–876). IEEE.

Chandy, K. M. (2006). Event-driven applications: Costs, benefits and design approaches.

Gartner Application Integration and Web Services Summit, 2006.

Chima, R. (2018). Event-Driven Applications In Software Development. Retrieved from https://www.bbconsult.co.uk/blog/event-driven-applications

Choudhari, J., & Suman, U. (2012a). Phase wise effort estimation for software maintenance: an extended SMEEM model. In Proceedings of the CUBE International Information Technology Conference (pp. 397–402).

Choudhari, J., & Suman, U. (2012b). Story points based effort estimation model for software maintenance. Procedia Technology, 4, 761–765.

Commeyne, C., Abran, A., & Djouab, R. (2016). Effort Estimation with Story Points and COSMIC Function Points-An Industry Case Study. Position Paper Software Measurement News.

Conte, S. D., Dunsmore, H. E., & Shen, Y. E. (1986). Software engineering metrics and models. Benjamin-Cummings Publishing Co., Inc.

Corazza, A., Di Martino, S., Ferrucci, F., Gravino, C., & Mendes, E. (2011). Investigating the use of support vector regression for web effort estimation. Empirical Software Engineering, 16(2), 211–243.
COSMIC. Course Registration ('C-Reg') System Case Study Version 2.0 (2015).

COSMIC. (2017a). Guideline for Sizing Business Application Software v1.3.

COSMIC. ACME Car Hire Case Study v1.0.1 (2018).

COSMIC, C. S. M. I. C. (2011). The Guideline for the use of COSMIC FSM to manage Agile Projects.

COSMIC, C. S. M. I. C. (2017b). The COSMIC Functional Size Measurement Method Measurement Manual (The COSMIC Implementation Guide for ISO/IEC 19761: 2017) Version 4.0.2.

Czarnacka-Chrobot, B. (2009). Standardization of software size measurement. In Internet–Technical Development and Applications (pp. 149–156). Springer.

Davis, R. (2001). Business process modelling with ARIS: a practical guide. Springer Science & Business Media.

Demirors, O., & Gencel, C. (2009). Conceptual association of functional size measurement methods. IEEE Software, 26(3), 71–78.

Desharnais, J.-M., Buglione, L., & Kocatürk, B. (2011). Using the COSMIC method to estimate Agile user stories. In Proceedings of the 12th International Conference on Product Focused Software Development and Process Improvement (pp. 68–73).

Desharnais, J.-M., Kocaturk, B., & Abran, A. (2011). Using the cosmic method to evaluate the quality of the documentation of agile user stories. In 2011 Joint Conference of the 21st International Workshop on Software Measurement and the 6th International Conference on Software Process and Product Measurement (pp. 269–272). IEEE.

Desharnais, J., Buglione, L., & Kocaturk, B. (2011). Improving Agile Software Projects Planning Using the COSMIC Method. In workshop on Managing Client Value Creation Process in Agile Projects (Torre Cane, Italy.

Desmarets, P. (2018). Data Modeling is Dead...Long Live Schema Design! Retrieved February 2, 2020, from https://www.dataversity.net/data-modeling-dead-long-live-schema-design/

Dingsøyr, T., Falessi, D., & Power, K. (2019). Agile development at scale: the next frontier. IEEE Software, 36(2), 30–38.

Dragicevic, S., Celar, S., & Novak, L. (2014). Use of method for elicitation, documentation, and validation of software user requirements (MEDoV) in agile software development projects. In 2014 Sixth International Conference on Computational Intelligence, Communication Systems and Networks (pp. 65–70). IEEE.

Dragoni, N., Giallorenzo, S., Lafuente, A. L., Mazzara, M., Montesi, F., Mustafin, R., & Safina, L. (2017). Microservices: yesterday, today, and tomorrow. In Present and ulterior software engineering (pp. 195–216). Springer.

Dumas-Monette, J.-F., & Trudel, S. (2014). Requirements engineering quality revealed through functional size measurement: an empirical study in an agile context. In 2014 Joint Conference of the International Workshop on Software Measurement and the International Conference on Software Process and Product Measurement (pp. 222–232). IEEE.

Easterbrook, S., Singer, J., Storey, M.-A., & Damian, D. (2008). Selecting empirical methods for software engineering research. In Guide to advanced empirical software engineering (pp. 285–311). Springer.

Engel, T., Langermeier, M., Bauer, B., & Hofmann, A. (2018). Evaluation of microservice architectures: A metric and tool-based approach. In International Conference on Advanced Information Systems Engineering (pp. 74–89). Springer.

Ertaban, C., Gezgin, S., Bagriyanik, S., Albey, E., & Karahoca, A. (2017). Cevik Yontemlerde Cosmic Islev Puani ve Hikaye Puaninin Birlikte Kullanimi (Using Cosmic Funcion Points and Story Points Together in Agile Frameworks). In UYMS (pp. 378–390).

Fehlmann, T., & Santillo, L. (2010). From story points to cosmic function points in agile software development–a six sigma perspective. In Metrikon-Software Metrik Kongress (p. 24).

Feldt, R., & Magazinius, A. (2010). Validity threats in empirical software engineering research-an initial survey. In Seke (pp. 374–379).

Fenton, N. E., & Pfleeger, S. L. (1997). Software metrics: a rigorous and practical approach PWS Publishing Co. Boston, MA, USA.

Fenton, N. E., & Pfleeger, S. L. (1998). Software Metrics: A Rigorous and Practical Approach: Brooks. Cole.

Fetcke, T., Abran, A., & Dumke, R. (2001). A Generalized Representation for Selected Functional Size Measurement Methods. In in Current Trends in Software Measurment-Proceedings of the 11th International Workshop on Software Measurement Montreal:IWSM (pp. 1–25).

Field, A. (2013). Discovering statistics using IBM SPSS statistics. sage.

Forward, A., & Lethbridge, T. C. (2002). The relevance of software documentation, tools and technologies: a survey. In Proceedings of the 2002 ACM symposium on Document engineering (pp. 26–33).

Foss, T., Stensrud, E., Kitchenham, B., & Myrtveit, I. (2003). A simulation study of the model evaluation criterion MMRE. IEEE Transactions on Software Engineering, 29(11), 985–995.

Fowler, M. (2005). Domain Event. Retrieved from https://martinfowler.com/eaaDev/DomainEvent.html

Fowler, M., & Highsmith, J. (2001). The agile manifesto. Software Development, 9(8), 28–35.

Garousi, V., Coşkunçay, A., Betin-Can, A., & Demirörs, O. (2015). A survey of software engineering practices in Turkey. Journal of Systems and Software, 108, 148–177.

Garousi, V., Petersen, K., & Ozkan, B. (2016). Challenges and best practices in industry-academia collaborations in software engineering: A systematic literature review. Information and Software Technology, 79, 106–127.

Gencel, C., & Demirors, O. (2008). Functional size measurement revisited. ACM Transactions on Software Engineering and Methodology (TOSEM), 17(3), 1–36.

Gross, A., & Doerr, J. (2009). EPC vs. UML activity diagram-two experiments examining their usefulness for requirements engineering. In 2009 17th IEEE International Requirements Engineering Conference (pp. 47–56). IEEE.

Hacaloglu, T., & Demirors, O. (2019). Measureability of Functional Size in Agile

Software Projects: Multiple Case Studies with COSMIC FSM. In 2019 45th Euromicro Conference on Software Engineering and Advanced Applications (SEAA) (pp. 204–211). https://doi.org/10.1109/SEAA.2019.00041

Hacaloğlu, T., & Demirörs, O. (2018). Challenges of using software size in agile software development: A systematic literature review. CEUR Workshop Proceedings.

Hacaloglu, T., Deveci, A., Bağrıyanık, S., & Demirors, O. (2019). Çevik yöntemlerde büyüklük ölçümü: bağımsız ölçümler mümkün mü?,. In 13. Ulusal Yazılım Mühendisliği Sempozyumu. İzmir, Türkiye.

Hamouda, A. E. D. (2014). Using agile story points as an estimation technique in cmmi organizations. In 2014 agile conference (pp. 16–23). IEEE.

Han, J., Haihong, E., Le, G., & Du, J. (2011). Survey on NoSQL database. In 2011 6th international conference on pervasive computing and applications (pp. 363–366). IEEE.

Hastings, T. E., & Sajeev, A. S. M. (2001). A vector-based approach to software size measurement and effort estimation. IEEE Transactions on Software Engineering, 27(4), 337–350.

Hecht, R., & Jablonski, S. (2011). NoSQL evaluation: A use case oriented survey. In 2011 International Conference on Cloud and Service Computing (pp. 336–341). IEEE.

Helendi, A. (2018). What Is Event-Driven Programming And Why Is It So Popular? Retrieved from https://dzone.com/articles/what-is-event-driven-programming-and-why-is-it-so

Hevner, A. R., March, S. T., Park, J., & Ram, S. (2004). Design science in information systems research. MIS Quarterly, 75–105.

Hoda, R., Salleh, N., Grundy, J., & Tee, H. M. (2017). Systematic literature reviews in agile software development: A tertiary study. Information and Software Technology, 85, 60–70.

Hohman, M. M. (2005). Estimating in actual time [extreme programming]. In Agile Development Conference (ADC'05) (pp. 132–138). IEEE.

Howell, D. C. (2012). Statistical methods for psychology. Cengage Learning.

Hsieh, D. (2014). NoSQL Data Modeling. Retrieved March 30, 2019, from https://www.ebayinc.com/stories/blogs/tech/nosql-data-modeling/

Huijgens, H., Bruntink, M., van Deursen, A., van der Storm, T., & Vogelezang, F. (2016). An exploratory study on functional size measurement based on code. In Proceedings of the international conference on software and systems process (pp. 56–65). ACM.

Huijgens, H., & Solingen, R. van. (2014). A replicated study on correlating agile team velocity measured in function and story points. In Proceedings of the 5th International Workshop on Emerging Trends in Software Metrics (pp. 30–36).

Humphrey, W. S. (1995). A discipline for software engineering. Pearson Education India.

Hussain, I., Kosseim, L., & Ormandjieva, O. (2013). Approximation of COSMIC functional size to support early effort estimation in Agile. Data & Knowledge Engineering, 85, 2–14.

Idri, A., Abnane, I., & Abran, A. (2018). Evaluating Pred (p) and standardized accuracy criteria in software development effort estimation. Journal of Software: Evolution and Process, 30(4), e1925.

IFPUG. (2009). FPUG FSM Method: ISO/IEC 20926 - Software and systems engineering –Software measurement – IFPUG functional size measurement method, New York: Interna-tional Function Point User Group (IFPUG).

Inayat, I., Salim, S. S., Marczak, S., Daneva, M., & Shamshirband, S. (2015). A systematic literature review on agile requirements engineering practices and challenges. Computers in Human Behavior, 51, 915–929.

ISO/IEC 19761:2017. (2017). Software Engineering –COSMIC: A Functional Size Measurement Method. International Organization for Standardization (2017).

ISO. (2007). ISO/IEC 14143 Information Technology Software measurement Functional sizemeasurement Part 1-6. ISO, Geneva (1998-2007).

Jamshidi, P., Pahl, C., Mendonça, N. C., Lewis, J., & Tilkov, S. (2018). Microservices: The journey so far and challenges ahead. IEEE Software, 35(3), 24–35.

Javdani, T., Zulzalil, H., Ghani, A. A. A., Sultan, A. B. M., & Parizi, R. M. (2013). On the current measurement practices in agile software development. ArXiv Preprint ArXiv:1301.5964.

Jeffery, R., Ruhe, M., & Wieczorek, I. (2000). A comparative study of two software development cost modeling techniques using multi-organizational and company-specific data. Information and Software Technology, 42(14), 1009–1016.

Jorgensen, M. (1995). Experience with the accuracy of software maintenance task effort prediction models. IEEE Transactions on Software Engineering, 21(8), 674–681.

Jørgensen, M. (2004a). Regression models of software development effort estimation accuracy and bias. Empirical Software Engineering, 9(4), 297–314.

Jørgensen, M. (2004b). Top-down and bottom-up expert estimation of software development effort. Information and Software Technology, 46(1), 3–16.

Kalske, M., Mäkitalo, N., & Mikkonen, T. (2017). Challenges when moving from monolith to microservice architecture. In International Conference on Web Engineering (pp. 32–47). Springer.

Kang, S., Choi, O., & Baik, J. (2010). Model-based dynamic cost estimation and tracking method for agile software development. In 2010 IEEE/ACIS 9th International Conference on Computer and Information Science (pp. 743–748). IEEE.

Kappel, G., Pröll, B., Retschitzegger, W., & Schwinger, W. (2001). Modelling ubiquitous web applications-the wuml approach. In International Conference on Conceptual Modeling (pp. 183–197). Springer.

Khatri, S. K., Malhotra, S., & Johri, P. (2016). Use case point estimation technique in software development. In 2016 5th international conference on reliability, infocom technologies and optimization (trends and future directions)(ICRITO) (pp. 123–128). IEEE.

Kitchenham, B. (2004). Procedures for performing systematic reviews. Keele, UK, Keele University, 33(2004), 1–26.

Kitchenham, B. A., Pickard, L. M., MacDonell, S. G., & Shepperd, M. J. (2001). What accuracy statistics really measure. IEE Proceedings-Software, 148(3), 81–85.

Kitchenham, B., & Charters, S. (2007). Guidelines for performing systematic literature reviews in software engineering.

Kocaguneli, E., & Menzies, T. (2013). Software effort models should be assessed via leave-one-out validation. Journal of Systems and Software, 86(7), 1879–1890.

Kong, J., Jung, J.-Y., & Park, J. (2009). Event-driven service coordination for business process integration in ubiquitous enterprises. Computers & Industrial Engineering, 57(1), 14–26.

Kumar, C. S., Kumari, A. A., & Perumal, R. S. (2014). An optimized agile estimation plan using harmony search algorithm. International Journal of Engineering and Technology (IJET), 6(5).

Laigner, R., Kalinowski, M., Diniz, P., Barros, L., Cassino, C., Lemos, M., … Zhou, Y. (2020). From a Monolithic Big Data System to a Microservices Event-Driven Architecture. In 2020 46th Euromicro Conference on Software Engineering and Advanced Applications (SEAA) (pp. 213–220). IEEE.

Lavazza, L., & Meli, R. (2014). An evaluation of simple function point as a replacement of IFPUG function point. In 2014 joint conference of the international workshop on software measurement and the international conference on software process and product measurement (pp. 196–206). IEEE.

Lenarduzzi, V., Lunesu, I., Matta, M., & Taibi, D. (2015). Functional size measures and effort estimation in agile development: a replicated study. In International Conference on Agile Software Development (pp. 105–116). Springer.

Lenarduzzi, V., & Taibi, D. (2014). Can Functional Size Measures Improve Effort Estimation in SCRUM ? ICSEA 2014 : The Ninth International Conference on Software Engineering Advances.

Lethbridge, T. C., Singer, J., & Forward, A. (2003). How software engineers use documentation: the state of the practice. IEEE Software, 20(6), 35–39. https://doi.org/10.1109/MS.2003.1241364

Lübke, D. (2006). Transformation of Use Cases to EPC Models. In EPK (pp. 137–156). Citeseer.

Lubke, D., Schneider, K., & Weidlich, M. (2008). Visualizing use case sets as BPMN processes. In 2008 Requirements Engineering Visualization (pp. 21–25). IEEE.

MacDonell, S. G. (1997). Establishing relationships between specification size and software process effort in CASE environments. Information and Software Technology, 39(1), 35–45.

MacDonell, S. G., & Gray, A. R. (1998). A comparison of modeling techniques for software development effort prediction.

Malhotra, R. (2016). Empirical research in software engineering: concepts, analysis, and applications. CRC Press.

Mazzara, M., Bucchiarone, A., Dragoni, N., & Rivera, V. (2020). Size Matters: Microservices Research and Applications. In Microservices (pp. 29–42). Springer.

Michelle, W. (2013). Event Data vs Entity Data — How to store user properties in Keen IO. Retrieved December 20, 2020, from https://keen.io/blog/event-data-vs-entity-data-how-to-store-user-properties-in-keen-io/

Miranda, E., Bourque, P., & Abran, A. (2009). Sizing user stories using paired comparisons. Information and Software Technology, 51(9), 1327–1337.

Mistrík, I., Bahsoon, R., Ali, N., Heisel, M., & Maxim, B. (2017). Software Architecture for Big Data and the Cloud. Morgan Kaufmann.

Mongiello, M., Nocera, F., Di Sciascio, E., & Di Noia, T. (2018). Guest Editorial: software architecture for the web of things (SAWoT). INST ENGINEERING TECHNOLOGY-IET MICHAEL FARADAY HOUSE SIX HILLS WAY ….

Montgomery, D. C., Peck, E. A., & Vining, G. G. (2012). Introduction to linear regression analysis (Vol. 821). John Wiley & Sons.

Moulla, D. K., & Abran, A. (2021). Duration Estimation Models for Open Source Software Projects.

Myrtveit, I., Stensrud, E., & Shepperd, M. (2005). Reliability and validity in comparative studies of software prediction models. IEEE Transactions on Software Engineering, 31(5), 380–391.

NESMA. (2004). NESMA functional size measurement method conform ISO/IEC 24570, version 2.2, Netherlands Software Measurement User Association (NESMA).

NoSQL Databases Explained. (2018). Retrieved February 7, 2020, from www.mongodb.com/nosql-explained

Nunes, N. J. (2009). iUCP–estimating interaction design projects with enhanced use case points. In International Workshop on Task Models and Diagrams for User Interface Design (pp. 131–145). Springer.

Ochodek, M. (2016). Approximation of COSMIC functional size of scenario-based requirements in Agile based on syntactic linguistic features—a replication study. In 2016 joint conference of the international workshop on software measurement and the international conference on software process and product measurement (IWSM-MENSURA) (pp. 201–211). IEEE.

Offermann, P., Levina, O., Schönherr, M., & Bub, U. (2009). Outline of a design science research process. In Proceedings of the 4th International Conference on Design Science Research in Information Systems and Technology (pp. 1–11).

Özcan-Top, Ö., & Demirors, O. (2019). Application of a software agility assessment model–AgilityMod in the field. Computer Standards & Interfaces, 62, 1–16.

Özkan, B., & Demirors, O. (2016). On the Seven Misconceptions about Functional Size Measurement. In 2016 Joint Conference of the International Workshop on Software Measurement and the International Conference on Software Process and Product Measurement (IWSM-MENSURA) (pp. 45–52). https://doi.org/10.1109/IWSM-Mensura.2016.018

Ozkan, B., Turetken, O., & Demirors, O. (2008). Software functional size: For cost estimation and more. In European Conference on Software Process Improvement (pp. 59–69). Springer.

Özkaya, A., Ungan, E., & Demirors, O. (2011). Common practices and problems in effort data collection in the software industry. In 2011 Joint Conference of the 21st International Workshop on Software Measurement and the 6th International Conference on Software Process and Product Measurement (pp. 308–313). IEEE.

Paetsch, F., Eberlein, A., & Maurer, F. (2003). Requirements engineering and agile software development. In WET ICE 2003. Proceedings. Twelfth IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises, 2003. (pp. 308–313). IEEE.

Parulkar, S. (2020a). The importance of event-driven architecture in the digital world.

Parulkar, S. (2020b). The importance of event-driven architecture in the digital world. Retrieved from https://www.redhat.com/en/blog/importance-event-driven-architecture-digital-world

Parvez, A. W. M. M. (2013). Efficiency factor and risk factor based user case point test effort estimation model compatible with agile software development. In 2013 International Conference on Information Technology and Electrical Engineering (ICITEE) (pp. 113–118). IEEE.

Pavlovski, C. J., & Zou, J. (2008). Non-Functional Requirements in Business Process Modeling. In APCCM (Vol. 8, pp. 103–112).

Perry, D. E., Sim, S. E., & Easterbrook, S. M. (2004). Case studies for software engineers. In Proceedings. 26th International Conference on Software Engineering (pp. 736–738). IEEE.

Petersen, K., & Gencel, C. (2013). Worldviews, research methods, and their relationship to validity in empirical software engineering research. In 2013 joint conference of the 23rd international workshop on software measurement and the 8th international conference on software process and product measurement (pp. 81–89). IEEE.

Popli, R., & Chauahn, N. (2015). Managing uncertainty of story-points in agile software. In 2015 2nd International Conference on Computing for Sustainable Global Development (INDIACom) (pp. 1357–1361). IEEE.

Popli, R., & Chauhan, N. (2014a). Agile estimation using people and project related factors. In 2014 International Conference on Computing for Sustainable Global Development (INDIACom) (pp. 564–569). IEEE.

Popli, R., & Chauhan, N. (2014b). Cost and effort estimation in agile software development. In 2014 international conference on reliability optimization and information technology (ICROIT) (pp. 57–61). IEEE.

Popli, R., & Chauhan, N. (2014c). Estimation in agile environment using resistance factors. In 2014 International Conference on Information Systems and Computer Networks (ISCON) (pp. 60–65). IEEE.

Port, D., & Korte, M. (2008). Comparative studies of the model evaluation criterions mmre and pred in software cost estimation research. In Proceedings of the Second ACM-IEEE international symposium on Empirical software engineering and measurement (pp. 51–60).

Power, K. (2011). Using silent grouping to size user stories. In International Conference on Agile Software Development (pp. 60–72). Springer.

Rabhi, F., & Demirors, O. (2018). Software Engineering Research-White Paper.

Reifer, D. J. (2000). Web development: estimating quick-to-market software. IEEE Software, 17(6), 57–64.

Ren, A., & Yun, C. (2013). Research of Software Size Estimation Method. In 2013 International Conference on Cloud and Service Computing (pp. 154–155). https://doi.org/10.1109/CSC.2013.32

Richards, M. (2015). Software architecture patterns (Vol. 4). O'Reilly Media, Incorporated 1005 Gravenstein Highway North, Sebastopol, CA ….

Riehle, D. M., Jannaber, S., Karhof, A., Thomas, O., Delfmann, P., & Becker, J. (2016). On the de-facto Standard of Event-driven Process Chains: How EPC is defined in Literature. Modellierung 2016.

Riggins, J. (2017). Effective Microservices Architecture with Event-Driven Design.

Rodríguez, P., Mäntylä, M., Oivo, M., Lwakatare, L. E., Seppänen, P., & Kuvaja, P. (2019). Advances in using agile and lean processes for software development. In Advances in Computers (Vol. 113, pp. 135–224). Elsevier.

Runeson, P., Engström, E., & Storey, M.-A. (2020). The design science paradigm as a frame for empirical software engineering. In Contemporary empirical methods in software engineering (pp. 127–147). Springer.

Runeson, P., & Höst, M. (2009). Guidelines for conducting and reporting case study research in software engineering. Empirical Software Engineering, 14(2), 131.

Runeson, P., Host, M., Rainer, A., & Regnell, B. (2012). Case study research in software engineering: Guidelines and examples. John Wiley & Sons.

Salmanoglu, M., Hacaloglu, T., & Demirors, O. (2017). Effort Estimation for Agile Software Development : Comparative Case Studies Using COSMIC Functional Size Measurement and Story Points. ACM International Conference Proceeding Series. https://doi.org/10.1145/3143434.3143450

Salmanoğlu, M., Öztürk, K., Bağrıyanık, S., Ungan, E., & Demirörs, O. (2015). Benefits and challenges of measuring software size: early results in a large organization. In 25th International Workshop on Software Measurement and 10th International Conference on Software Process and Product Measurement, IWSM-Mensura.

Santana, C., Leoneo, F., Vasconcelos, A., & Gusmão, C. (2011). Using function points in agile projects. In International Conference on Agile Software Development (pp. 176–191). Springer.

Satapathy, S. M., Panda, A., & Rath, S. K. (2014). Story point approach based agile software effort estimation using various svr kernel methods.

Satapathy, S. M., & Rath, S. K. (2017). Empirical assessment of machine learning models for agile software development effort estimation using story points. Innovations in Systems and Software Engineering, 13(2–3), 191–200.

Scheer, A.-W., Thomas, O., & Adam, O. (2005). Process Modeling Using Event-Driven Process Chains. Process-Aware Information Systems, 119.

Schober, P., Boer, C., & Schwarte, L. A. (2018). Correlation coefficients: appropriate use and interpretation. Anesthesia & Analgesia, 126(5), 1763–1768.

Shepperd, M., & Schofield, C. (1997). Estimating software project effort using analogies. IEEE Transactions on Software Engineering, 23(11), 736–743.

Sigweni, B., Shepperd, M., & Turchi, T. (2016). Realistic assessment of software effort estimation models. In Proceedings of the 20th International Conference on Evaluation and Assessment in Software Engineering (pp. 1–6).

Skowronski, J. (2019). Best Practices for Event-Driven Microservice Architecture.
Software AG. (2016). BPM WITH ARIS THE ARIS METHOD. Lecture Slides Part II. Retrieved June 9, 2019, from http://docplayer.net/45964277-Bpm-with-aris-the-aris-method-lecture-slides-part-ii-software-agall-rights-reserved.html

Soni, D., & Kohli, P. (2017). Cost estimation model for Web applications using agile software development methodology. Pertanika J. Sci. Technol., 25, 931–938.

Stephens, R. (2015). Beginning software engineering. John Wiley & Sons.

Sucaciu, B. (2020). How event-driven architecture solves modern web app problems. Retrieved December 20, 2020, from https://stackoverflow.blog/2020/03/16/how-event-driven-architecture-solves-modern-web-app-problems/

Sykes, A. O. (1993). An introduction to regression analysis.

Taibi, D., & Systä, K. (2019). A Decomposition and Metric-Based Evaluation Framework for Microservices. In International Conference on Cloud Computing and Services Science (pp. 133–149). Springer.

Tate, G., Verner, J., & Veryard, R. (1991). Software costing in practice. The Economics of Information Systems and Software, 101–126.

The Reactive Manifesto v1.0. (2013). Retrieved from https://www.reactivemanifesto.org/pdf/the-reactive-manifesto-1.0.pdf

TIBCO. (n.d.). What is Event-driven Architecture? Retrieved December 20, 2020, from https://www.tibco.com/reference-center/what-is-event-driven-architecture

Top, O. O., Demirors, O., & Ozkan, B. (2009). Reliability of COSMIC functional size measurement results: A multiple case study on industry cases. In 2009 35th Euromicro Conference on Software Engineering and Advanced Applications (pp. 327–334). IEEE.

Turetken, O., Top, O. O., Ozkan, B., & Demirors, O. (2008). The impact of individual assumptions on functional size measurement. In Software Process and Product Measurement (pp. 155–169). Springer.

Ugalde, F., Quesada-López, C., Martínez, A., & Jenkins, M. (2020). A comparative study on measuring software functional size to support effort estimation in agile.

Ungan, E., Cizmeli, N., & Demirors, O. (2014). Comparison of functional size based estimation and story points, based on effort estimation effectiveness in SCRUM projects. In Proceedings - 40th Euromicro Conference Series on Software Engineering and Advanced Applications, SEAA 2014. https://doi.org/10.1109/SEAA.2014.83

Usman, M., Mendes, E., & Börstler, J. (2015). Effort Estimation in Agile Software Development: A Survey on the State of the Practice. In Proceedings of the 19th International Conference on Evaluation and Assessment in Software Engineering. New York, NY, USA: Association for Computing Machinery. https://doi.org/10.1145/2745802.2745813

Usman, M., Mendes, E., Weidt, F., & Britto, R. (2014). Effort estimation in agile software development: a systematic literature review. In Proceedings of the 10th International Conference on Predictive Models in Software Engineering (pp. 82–91). ACM.

Van der Aalst, W. M. P. (1999). Formalization and verification of event-driven process chains. Information and Software Technology, 41(10), 639–650.

Van Koten, C., & Gray, A. R. (2006). An application of Bayesian network for predicting object-oriented software maintainability. Information and Software Technology, 48(1), 59–67.

van Valkenhoef, G., Tervonen, T., de Brock, B., & Postmus, D. (2011). Quantitative release planning in extreme programming. Information and Software Technology, 53(11), 1227–1235.

VanHilst, M., Huang, S., Mulcahy, J., Ballantyne, W., Suarez-Rivero, E., & Harwood, D. (2011). Measuring effort in a corporate repository. In 2011 IEEE International Conference on Information Reuse & Integration (pp. 246–252). IEEE.

Vernon, V. (2013). Implementing domain-driven design. Addison-Wesley.

Vural, H., Koyuncu, M., & Misra, S. (2018). A Case Study on Measuring the Size of Microservices. In International Conference on Computational Science and Its Applications (pp. 454–463). Springer.

Wieringa, R. J. (2014). Design science methodology for information systems and software engineering. Springer.

Wohlin, C. (2014). Guidelines for snowballing in systematic literature studies and a replication in software engineering. In Proceedings of the 18th international conference on evaluation and assessment in software engineering (pp. 1–10).

Wohlin, C., Höst, M., & Henningsson, K. (2003). Empirical research methods in software engineering. In Empirical methods and studies in software engineering (pp. 7–23). Springer.

Wohlin, C., Runeson, P., Höst, M., Ohlsson, M. C., Regnell, B., & Wesslén, A. (2012). Experimentation in software engineering. Springer Science & Business Media.

Yin, R. K. (2017). Case study research and applications: Design and methods. Sage publications.

Zahraoui, H., & Idrissi, M. A. J. (2015). Adjusting story points calculation in scrum effort & time estimation. In 2015 10th International Conference on Intelligent Systems: Theories and Applications (SITA) (pp. 1–8). IEEE.

Zainal, Z. (2007). Case study as a research method. Jurnal Kemanusiaan, 5(1).

Zaki, A. K. (2014). NoSQL databases: new millennium database for big data, big users, cloud computing and its security challenges. International Journal of Research in Engineering and Technology (IJRET), 3(15), 403–409.

Zhelev, S., & Rozeva, A. (2019). Using microservices and event driven architecture for big data stream processing. In AIP Conference Proceedings (Vol. 2172, p. 90010). AIP Publishing LLC.

Zhu, Y., Richins, D., Halpern, M., & Reddi, V. J. (2015). Microarchitectural implications of event-driven server-side web applications. In Proceedings of the 48th International Symposium on Microarchitecture (pp. 762–774).

CURRICULUM VITAE

Tuna Hacaloğlu

EDUCATION

2014-2021	Middle East Technical University, Information Systems, Ph.D.
2010-2013	Middle East Technical University, Information Systems, M.S.
2005-2009	Atilim University, Software Engineering, B.S.
1997-2004	Ankara Tevfik Fikret High School

FOREIGN LANGUAGES

Turkish	Native
French	Advanced
English	Advanced

WORK EXPERIENCE

2016-CONT.	Instructor, Department of Information Systems Engineering, Atilim University
2009-2016	Research Assistant, Department of Information Systems Engineering, Atilim University

HONORS & AWARDS

1	2009- Department of Software Engineering first ranked student award
2	2009- First Ranked Senior Project Award (Title: CMMI Compliant Code Review Plug-in for Eclipse)
3	2008-2009 Full scholarship
4	2007-2008 Full scholarship

CITATIONS

Sum of times cited without self-citations (ISI Web of Science):	65
H-index (ISI Web of Science):	5

PUBLICATIONS (SCI-E / SSCI)

1	Cigdem Turhan, Ibrahim Akman, Tuna Hacaloglu, (2019), Online collaborative tool usage for review meetings in software engineering courses, Interactive Learning Environments (published online)
2	Vahid Garousi, Michael Felderer, Tuna Hacaloglu, (2018), What We Know about Software Test Maturity and Test Process Improvement, IEEE Software, 35(1), 84-92
3	Vahid Garousi, Michael Felderer, Tuna Hacaloglu, (2017), Software test maturity assessment and test process improvement: A multivocal literature review, Information and Software Technology, 85, 16-42
4	Deepti Mishra, Sofiya Ostrovska & Tuna Hacaloglu, (2017), Exploring and expanding students' success in software testing, Information Technology & People, 30 (4), 927-945
5	Aylin Akca Okan, Tuna Hacaloglu and Ali Yazıcı, (2016), Study On Cloud Computing Perception of Turkish It Sector, Technical Gazette, 23(1), 1-8.
6	Deepti Mishra, Sofiya Ostrovska, and Tuna Hacaloglu, (2015) Assessing Team Work in Engineering Projects', International Journal of Engineering Education , 31(2), 627-634.
7	Deepti Mishra, Tuna Hacaloglu and Alok Mishra (2014), "Teaching Software Verification and Validation Course: A Case Study", International Journal of Engineering Education, 30(6A), 1476-1485.

CONFERENCE PRESENTATIONS

1	İbrahim Akman, Cigdem Turhan, Tuna Hacaloglu, Utilization of Online Collaborative Tools in Software Engineering: An Empirical Study on Review Meetings, (2021), 6th International Conference on Computer Science and Engineering, 15-17 Sept. 2021, Ankara, Turkey.
2	Tuna Hacaloglu, Huseyin Unlu, Onur Demirors, Alain Abran, (2020), COSMIC Light vs. COSMIC Manual: Case Studies in Functional Size Measurement, IWSM MENSURA, Mexico City
3	Tuna Hacaloglu, Aylin Deveci, Selami Bağrıyanık, Onur Demirors, Çevik yöntemlerde büyüklük ölçümü: bağımsız ölçümler mümkün mü?, 13. Ulusal Yazılım Mühendisliği Sempozyumu, İzmir, Türkiye
4	Tuna Hacaloglu and Onur Demirors, (2019), Measureability of functional size in Agile software projects: Multiple case studies with COSMIC FSM, Euromicro DSD/SEAA 2019, Kallithea, Chalkidiki, Greece
5	Yavuz İnal and Tuna Hacaloglu, Users' Behavioral Strategies Toward Mobile App Problems: Fight or Flight, (2019), Conference on e-Business, e-Services and e-Society, 37-49
6	Tuna Hacaloglu and Onur Demirors, Challenges of Using Software Size in Agile Software Development: A Systematic Literature Review, IWSM 2018, China, Beijing
7	Salmanoglu, Murat, Tuna Hacaloglu, and Onur Demirors. "Effort estimation for agile software development: Comparative case studies using COSMIC functional size measurement and story points." Proceedings of the 27th International Workshop on Software Measurement and 12th International Conference on Software Process and Product Measurement. ACM, 2017.
8	Hacaloglu, Tuna, Turhan, Çiğdem, and Akman, İbrahim. (2017) Utilization of Social Media in Higher Education with a Reflection on Turkey, 8th International Conference on Business, Law, Education and Disaster Management (BLEDS-17) Oct. 5-6, 2017 Paris (France), 53-55
9	Hacaloglu, Tuna, Eren, P. Erhan, Mishra, Deepti, Mishra, Alok, (2015), "A Software Development Process Model for Cloud by Combining Traditional Approaches", On the Move to Meaningful Internet Systems: OTM 2015 Workshops, Springer International Publishing, vol. 9416, pp. 421-430.
10	Tuna Hacaloğlu, Oya Beyan (2012), Measuring Innovative Characteristics of Students in Higher Education, 2nd International Engineering Education Conference, Antalya, Turkey, pp.34-41