2014 Joint Conference of the International Workshop on Software Measurement and the International Conference on Software Process and Product Measurement

Requirements Engineering Quality Revealed Through Functional Size Measurement: An Empirical Study in an Agile Context

Jean-François Dumas-Monette
Dept. of Computer Science
Université du Québec à Montréal
Montreal, Canada
e-mail: jfmonette@gmail.com

Sylvie Trudel
Dept. of Computer Science
Université du Québec à Montréal
Montreal, Canada
e-mail: trudel.s@uqam.ca

Abstract — **Software development organizations applying continuous process improvement, when faced with the limits of qualitative approaches, are looking into quantitative approaches to support decision making, namely for improvement of the software project estimation process. Quantitative approaches include sizing functional requirements with standards such as ISO 19761, known as the COSMIC method. But defects in the requirements may have an impact on the accuracy of the resulting functional size, as well as an impact on the project relative effort - sometimes known as the 'productivity rate' - and the measurement relative effort. Our research program is investigating the relationship between the attributes of requirements engineering (RE) outputs, the software process relative effort, and the measurement process relative effort. RE outputs studied are requirements and specifications documents and data models. As functional sizing is applied, thorough examination of RE outputs is done, which is likely to lead to identifying quality attributes and related findings.**

As a case study, this paper reports preliminary results related to the quality of requirements artefacts from a software development organization that is applying the Agile approach to its software development process. The functional size of the software developed through five projects was measured and compared with development effort and measurement effort, taking into account the quality rating of requirements. The results led to recommendations of improvement on the RE process that the organization could deploy in its current and next software projects. This paper also presents a list of functional sizing challenges that the measurer has faced, leading to proposed recommendations for planning any software measurement project.

Keywords- Functional size, COSMIC, FSM, Measure, Quality, Defects, Requirements engineering.

I. Introduction

Many software organizations apply continuous software process improvement in order to keep or enhance their perceived competiveness. Several approaches have been used in the past to improve the software process, such as software process assessment against a best practices model – done internally or by external experts [1][2], or such as retrospectives [3] carried out at regular interval by an Agile software team, especially targeting the improvement of the team's software process.

Assessments and retrospectives provide mostly qualitative results on the improvement of the software process. Organizations may measure the progress of improving their software process as a level on a maturity or capacity scale, such as levels 1 through 5 of the Capability Maturity Model Integration (CMMI) [4]. Nonetheless, some organizations are interested in measuring and managing quantitatively the efficiency of their software process, which can be expressed as the relative effort per size unit and often labelled as the 'productivity rate'.

Measuring the software productivity rate has several advantages for an organization seeking process improvement. First, it provides quantitative feedback over time on the efficiency of the improvement initiatives. Second, once the productivity rate is known, it can be used as an input to establish estimation models [25]. Third, the known productivity rate of a given project can be benchmarked against similar projects within and outside the organization.

The key element remains sizing the project, more specifically the software developed during the project, in units deemed useful for these purposes. Measuring the functional size has been proven useful [25] and several methods exist for that matter [26]. Functional size methods use the requirements and specifications as their primary input. When not available, requirements can often be derived from the existing software itself.

However, the quality of sizing results is directly linked to the quality of information serving as the basis for measurement activities [5]. When an organization performs functional size measurement, accurate and adequate information is required, typically in the form of requirements engineering artefacts, such as requirements and specification documents and logical data models.

In many cases, there are defects in the requirements, such as ambiguities, incompleteness and inconsistencies, despite effort being spent to verify and validate these requirements.

978-1-4799-4174-2/14 $31.00 © 2014 IEEE
DOI 10.1109/IWSM.Mensura.2014.43

These defects are likely to lead to inaccurate functional sizing. Requirements may need to be assessed to ensure a more accurate functional sizing result. Should this assessment be done prior to measurement or should the measurement activity itself serve as a defect identification mechanism in the requirements?

As a case study, this paper reports preliminary findings related to the quality of requirements engineering (RE) outputs from a small software development organization applying the Agile approach to its software process. Five projects were measured and observed. These preliminary findings led to recommendations of improvement on the RE process that the organization could deploy in its undergoing and next software projects. This paper also presents a list of functional sizing challenges that the measurer has faced, leading to proposed recommendations for planning any software measurement project.

The remainder of this paper is organized as follows. Section II surveys related work on quality assessment of inputs to functional size measurement and using the COSMIC method to identify defects in requirements. In section III, we describe the case study design and research approach. In section IV, we present a case study and its context. Section V describes the conduct of the case study while section VI presents the results. In section VII, we discuss other findings. Section VIII documents the threats to validity of our approach. Section IX draws four recommendations. Finally, section X concludes and sketches future work.

II. RELATED WORK

Functional size measurement (FSM) is a means for measuring the size of a software application, regardless of the technology used to implement it.

The COSMIC FSM method [23] is supported by the Common Software Measurement International Consortium (COSMIC) and is a recognized international standard (ISO 19761 [24]).

In the measurement of software functional size using COSMIC, the software functional processes and their triggering events must be identified. The unit of measurement in this method is the data movement, which is a base functional component that moves one or more data attributes belonging to a single data group. Data movements can be of four types: Entry (E), Exit (X), Read (R) or Write (W). The functional process is an elementary component of a set of user requirements triggered by one or more triggering events, either directly or indirectly, via an actor. The triggering event is an event occurring outside the boundary of the measured software and initiates one or more functional processes. The sub processes of each functional process constitute sequences of events, and a functional process comprises at least two data movement types: an Entry plus at least either an Exit or a Write. An Entry moves a data group, which is a set of data attributes, from a user across the boundary into the functional process, while an Exit moves a data group from a functional process across the boundary to the user requiring it. A Write moves a data group lying inside the functional process to persistent storage, and a Read moves a data group from persistent storage to the functional process. Figure 1 illustrates the generic flow of data groups through software from a functional perspective.

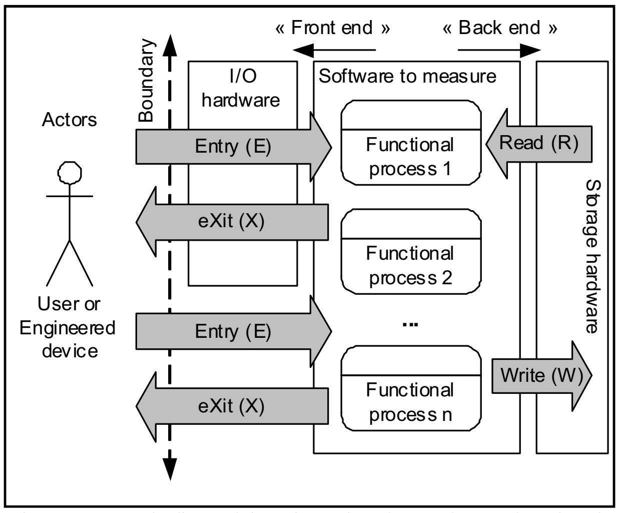

Figure 1. Generic flow of data through software from a functional perspective.

Several experiments reported on typical measurement errors made by inexperienced measurers, such as [8] and [9]. Some of these errors were the result of improper training on the COSMIC method. The other measurement errors were the result of defects in the requirements, such as ambiguities, incompleteness and inconsistencies. In other words, the defects in the requirements are likely to result in inaccurate sizing. Finding defects in requirements could be done with a review when the aim is to fix them. As measurement is often done on completed projects for which the organization has no intention to spend effort on fixing these defects, accurate sizing for estimation purposes is nonetheless required.

As measurement accuracy could be a challenge due to requirements defects, some researchers proposed various ways to identify these quality issues prior to or as part of applying the measurement process.

First, Desharnais and Abran have proposed a quality rating as a result of assessing the requirements used as an input to functional sizing [5]. Table 1 shows their proposed rating scale along with the description for each value.

This quality rating scale can be used at early stages of a measurement process applying the COSMIC method [23] to help understanding the level of precision of measurement results. This rating scale was later used in the COSMIC Guideline for assuring the accuracy of measurement [6]. The Quick Reference Guide for sizing business software [7] used the same quality rating as an indication to which extent measurement assumptions should be made as part of the measurement activities to compensate for defects in the requirements. To obtain a more accurate measure, a measurer should not ignore these defects but is encouraged to elicit assumptions and measure the software size accordingly.

TABLE 1. Documentation quality rating scale, from [5].

Rating	Description
a	The functional process is completely documented together with its data movements by type.
b	The functional process is documented but the description of the data moved is unclear. The input, output, stores and retrievals of each functional process are also described but not clearly enough to identify the number of data movements.
c	The functional process is identified only but their data movements are not.
d	The number of the functional process is given but they are not specified.
e	The functional process is not mentioned in the artefacts but is implicit.

Further research showed that the COSMIC method was used as a means to identify defects in functional requirements, such as [10] and [11]. In [11], experiments were conducted using a set of controlled requirements written as use cases, where a single document contained all functional processes.

III. Case study design

A. Research objectives

This paper reports on a case study that is part of a broader research program which aims to improve methods for specifying requirements in order to prevent defects. This research program has three objectives:

1. To propose techniques for defect identification in completed software requirements using Functional Size Methods (FSM), in order to improve requirements quality.
2. To improve the RE process to prevent the occurrence of defects earlier in the RE life cycle.
3. To adapt research findings in order to deploy them to industry.

The strategy to reach the objective 1 is to first investigate the impact of requirements defects on the software productivity rate, or relative project effort, but also to compare various requirements documentation techniques with the relative project effort on a larger scale. These activities should bring us to better understand the relationship between requirements quality, requirements documentation technique, and the software project relative effort.

B. Research assumptions

The design of this case study explores the following assumptions:

A1. Measurement productivity: *There is a direct relationship between requirements quality and the relative effort for sizing these requirements.* The quality rating is an objective qualitative value indicating the availability of information for sizing purposes. Therefore, requirements containing all required information for sizing should take less effort to measure than incomplete requirements. By confirming this assumption, we hope to convince software managers to sustain FSM effort as an affordable activity providing value to the software process, which relates to the third objective of our research program.

A2. Development productivity: *There is a direct relationship between requirements quality and the project relative effort.* The same information required to measure the functional size is also required by developers to fulfil their tasks. Therefore, requirements containing all required information for sizing should take less effort to develop than when information is missing. This assumption is linked with our first research objective.

A3. Extensibility: *The usage of the COSMIC method can be extended to identify defects in software functional requirements from the industry.* Previous research has shown that it can be done with a controlled set of requirements written as use cases. We believe it can be extended to various types of industry requirements. By confirming this assumption, we expect that FSM methods become generalized tools within a RE process, which relates to our second research objective.

C. Case selection

The strategy to select a case study was to find a software development organization having motivation factors to improve their RE process. The case study also had to be suitable to be undertaken as a software engineering graduate study project. As such, a single organization having multiple software projects was sought for. For those projects, the organization had to provide RE artefacts that were valid for functional sizing purposes. Project effort for those projects also had to be available.

D. Scoping the projects within the selected case study

The criteria to define the scope of this case study, that is which projects would be selected as subjects, were defined as follow:

1. Projects should have been completed within the last two years.
2. The project applied software process should be representative of the types of projects the organization was most likely to undertake in its foreseeable future.

As the case study experiment was limited to a four-months period, the number of projects to be selected would depend on factors influencing the measurement effort, thus their functional size and the quality of their requirements.

E. Data collection procedures

1) Startup measurement project

- Provide training to the measurer for the functional sizing method to be used, in this case the COSMIC method.
- With the organization's high-level management, establish the measurement project scope and the logistics, such as granted access to premises and to key personnel for the assigned measurer.
- For every project within the measurement scope:
  - o Provide the measurer access to RE artefacts.
  - o Obtain effort data.

2) Observe requirements while measuring software functional size

For each software project within the measurement scope:

- Measure the functional size using the COSMIC method.
- Record detail measurement data in a tool (Excel).
- While measuring, observe and record qualitative information about the RE artefacts using the Documentation Quality Rating Scale from Table 1 and basic requirements quality attributes defined in IEEE-Std-830 [12].
- Record the measurement effort.

3) Verify data quality

- Review measurements. The measurer will first review the inner consistency of functional size data. Areas of discrepancy in COSMIC sizing to be aware of include, but are not limited to:
  - o Objects of interests and data groups that are manipulated but never created.
  - o Modified functional processes that were not created.
  - o New functional processes that do not have a triggering entry.
- Verify functional size data. Although being an experienced software engineer, the measurer assigned to this case study was inexperienced, at first, in applying the COSMIC method. To ensure the quality of functional size data, a certified COSMIC expert measurer would verify these data. It was expected that this verification activity be more thorough on the first measured functional processes, until the COSMIC expert could judge that functional size data was of quality. Then, verification would be applied by sampling FSM data on all measured projects. This verification activity would also ensure that adequate observations were recorded about the quality of requirements.
- Verify effort data. Mechanisms for and cultural behaviours related to recording effort had to be verified to understand and qualify the accuracy of the recorded project effort.

F. Analysis and validation procedures

Measurement effort per project, project software development effort and their related software functional size would be recorded in a spreadsheet to analyze the following:

- The relative measurement effort in minutes per CFP.
- The relative project effort in staff-hours per CFP.

Data would be verified by the certified and experienced COSMIC measurer to ensure integrity with detailed measurement data.

From the recorded quality observations, identify similar defects or issues, group them, and report these findings. Findings would be validated by the certified COSMIC measurer by sampling on specific requirements items at the source of defect identification to ensure groupings are adequate and reflect on actual requirements documentation.

IV. CASE STUDY DESCRIPTION

A. Finding an organization for this case study

As part of the research program that includes this case study, several organizations had been contacted for partnership. Because of this, we knew that managers of Axon-ID were interested in answering the following questions:

1. What estimation models can be derived from our projects historical data?
2. Which improvements could be made on the RE process in order to facilitate early estimation?

The means to answer those questions included activities such as measuring the functional size of projects, collecting project effort data and assessing the quality of the RE process. Since these activities are part of this case study design, this made Axon-ID a suitable subject for the case study.

B. Context of the organization

This study was conducted at Axon-ID, a small software development organization located in North America, specializing in integration and modernization projects for applications in the finance domain. Axon-ID was founded in 1998 (16 years ago) and is currently employing 25 people. Over the years, they have also developed an expertise in functional test automation. Since its foundation, the organization has always been pursuing continuous improvement goals in order to improve their products, services and processes. One of the results of their organizational culture was the adoption of Agile and Lean software development principles and values since 2005. This has led Axon-ID to be composed of several cross-functional teams to which projects are assigned. This approach to software development processes has since then proven beneficial to the organization by improving its productivity and reducing costs, according to its managers.

Another benefit of adopting Agile and Lean software development processes is to improve estimates accuracy over time through analysis of past productivity of the software development team. This means that the estimates rely on which team a project will be assigned to. However, in an organizational context, the manager may not know at contract negotiation time which team will be assigned to the project. Hence there is a need to decouple the development effort estimate of a project from the development team productivity that will be assigned to the project. Estimates accuracy also relies upon other factors such as: type of client, previous experience with the business domain, size of the team, technology, and so on.

Axon-ID improves its software process by targeting areas for improvement, setting goals, applying actions, measuring progress and evaluating the efficiency of these actions. Their targets are aligned with business needs and revised periodically or on a needed basis.

At the time this case study begun, the organization had identified six areas for improvement as depicted in Figure 2. Progress measurement was made by using tasks completion against tasks planned for each of the improvement areas. The score variation among the six areas can be explained by the fact that improvement activities have started at different moments and that they do not progress at an equal pace.

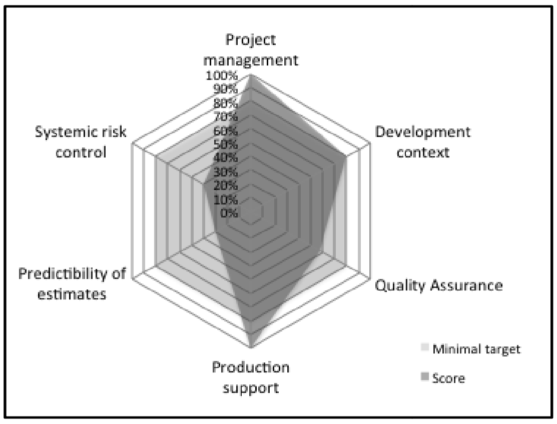

Figure 2. The organization's improvement measures per area of improvement.

With this internal progress assessment, the organization's managers felt confident that the improvement actions in most areas were under control, except predictability of estimates. They knew about quantitative methods to derive estimation models, although they had neither the know-how nor the resources availability to develop such models.

For an organization to accurately estimate the development effort of any given project, particularly at contract negotiation time, it should be able to rely on its historical data which take into account the factors that influence estimates accuracy cited previously. Therefore, this historical data should not be reliant on a particular team's estimated effort but on an objective functional size measure of projects.

Axon-ID's interest in functional size measurement was mainly in its objectivity and predictability. They were most interested in using these objective measurements in order to support strategic decisions related to the functional size of future projects and gaining from the benefits of the measurement process on software requirements engineering and software project estimation.

C. *Agile Approach and methods*

To be considered Agile, an approach to software development must be consistent with the four values and the 12 principles of the Agile Manifesto [13].

Since its introduction, authors of the Agile approach claim that it is able to deliver business value to clients, improve quality, reduce release time delays, conform to budget and delivery dates, and improve collaboration and communication between the client and the development team [14]. The implementation of the Agile approach is referred to as an Agile methodology. There exist several methodologies implementing the Agile approach, such as Scrum [14], eXtreme Programming (XP) [15], Kanban [16], and Test-Driven Development (TDD) [17] or Acceptance Test-Driven Development (ATDD) [20], an extension of TDD where functional and acceptance tests are automated amongst the most popular methodologies.

The decision of using an Agile method to manage a project usually relies on the following criteria: volatility of requirements, customer availability, project duration, project risks level, management support, project criticality and size of the team [18].

Axon-ID applies Scrum or Kanban, combined with ATDD for its software development projects.

D. *RE process and RE process output in Agile*

Scrum is an iterative and incremental software development method that is used to manage projects and requirements. Scrum recognizes that the requirements of a software project are likely to change during its development and addresses this challenge by organizing its activities in short time-boxed periods called 'sprints'. At the end of each of these periods, a Sprint Review, during which the work done during the sprint is presented to the stakeholders, takes place [14].

Scrum defines three primary roles:

- *Product Owner (PO)*: Responsible for defining the project scope and planning the execution of the project;
- *Development Team*: Responsible for building the product;
- *Scrum Master*: Responsible for keeping the process effective and efficient and continually improving it.

In Scrum, the task of documenting the requirements is the responsibility of the PO. In fact, any team member can document the requirements and Scrum encourages all members to be involved in the RE process. However, it is the responsibility of the PO to approve and prioritize these requirements.

A process using an Agile approach for RE values working software over exhaustive documentation [13]. Documentation is important as a communication tool but an Agile team should bear in mind to keep this documentation as lean as possible so its effort is better spent on building the right software for the customer. Agile RE and new technologies have given rise to new forms of documentation that were seldom used or not available several years ago. These new forms of documentation include digital pictures, wikis, intranet, mind maps, etc. Another form of documentation used in RE by Agile practitioners is automated tests. Written in an adequate language level, these tests can be used to describe expected behaviour and results of a piece of software.

A type of document that has been vastly adopted by Agile teams as part of their RE process is User Stories (US). US are artefacts used to document and discuss expected functionalities that need to be implemented. Typically, a US will have one of the following naming scheme [29]:

- "As a <role>, I want to <goal>"
- "As a <role>, I want to <goal>" so that <benefit>
- In order to <benefit> as a <role>, I want to <goal>

As proposed by Cohn [29], US should also have the following characteristics: independent, negotiable, valuable, estimable (measurable), small and testable (INVEST). US can be expressed at different levels of detail. Figure 3 illustrates the expected hierarchy of Agile requirements. US can be exploded in smaller actionable work items called tasks. A US describing high numbers of functionalities that will later be exploded in smaller US is called an Epic. A Feature, e.g. "Credit card payment", can in turn include many Epics. Finally, related features can be grouped together by Theme.

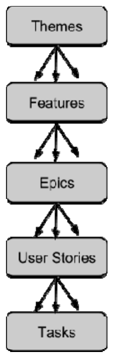

Figure 3. Hierarchy of Agile requirements.

At Axon-ID, all requirements are being written as US and stored in a web-based tool (Jira) specialized for managing Agile projects and Agile project requirements.

V. Measurement and observation conduct

A. *Startup measurement project activities*

Over the course of the first week, access to the premises was granted and a meeting with the organization's managers and key personnel took place. Managers at Axon-ID listed the projects that matched the defined project selection criteria. The initial list contained 23 projects. Upon closer examination of projects data, it was discovered that 10 of those projects would have to be discarded, for the following reasons:

- No development effort had taken place (e.g. they were analysis projects without the realization part, or were infrastructure projects);
- The developed software was no longer available when the FUR were inadequate for measurement.

The final scope of the measurement project was therefore reduced to 13 projects. Axon-ID's managers provided a priority amongst these projects to align the measurement effort with their interest in the resulting size and relative effort.

B. *Measurement and observation activities*

At an early stage of this measurement project, the assigned measurer was trained on the COSMIC method by a certified COSMIC measurer with extensive industry experience. Then, the COSMIC method was used to measure the size with each US or change request recorded in the RE recording tool. The measurement data was recorded in a spreadsheet tool. At the end of each project's measurement, the quality rating of the RE artefacts were recorded. Finally, measurement effort was recorded for each project.

Measurements were made on recent projects that reflect the current software development process of the organization. At the time of writing of this paper, the functional size of five projects out of 13 have been measured and accounted for preliminary results presented in the following sections. The measured projects were characterized with the following attributes: technology, business domain, development methodology, and project type. Table 2 shows the attributes of each project part of this case study.

TABLE 2. ATTRIBUTES OF SELECTED PROJECTS.

Project	Technology	Business Domain	Type
A	Web / Java / PostgreSQL	Labour Union	New
B	Web /	Marketing	Evolution
C	Web / .Net / MSSQL	Human Resources	New
D	Web / Java / PostgreSQL / Solr	Art Merchant	New
E	Mobile Web / Android / .Net	Inventory Management	New

All selected projects applied the Scrum method to implement the Agile approach, combined with ATDD.

C. Verify data quality activities

1) Functional size data

We wanted to understand the completeness of requirements. It was observed that, even though not all requirements were in the expected format from a measuring perspective, all functional processes had been documented. It was shown that this completeness was the result of having the remuneration of any member of a project's team directly linked with the effort that was being signed off in the same tool that was used for requirements management. Therefore, since any effort had to be linked with a software requirement, this practice ensured that all requirements were included in the requirements management tool.

2) Project effort data

Also, this practice described above ensures that effort was correctly and consistently signed off, leading to accurate effort tracking by the organization.

VI. RESULTS

This section presents and discusses the collected data in this case study to the initial assumptions.

A. Measurement relative effort

Table 3 shows the measurement effort for each project in staff-hours, the project functional size, the relative measurement effort (in minutes per CFP), and the assessed quality rating.

TABLE 3. MEASUREMENT EFFORT PER PROJECT.

Project	Measurement Effort (staff-hours)	Project Size (CFP)	Relative measurement effort (min./CFP)	Quality rating
A	164	715	13.7	b
B	45	177	15.3	c
C	26	303	5.1	a
D	37	352	6.3	a
E	18	129	8.4	b

As these projects were measured in order A to E, the measurer had to overcome a learning curve over project A, a project considered 'big' with a size of 715 CFP. Looking at project A and E, which have the same quality rating of 'b', we see that project E as a relative measurement effort 39% lower than of project A.

Project B, having a quality rating of 'c', shows the highest relative measurement effort of 15.3 minutes/CFP.

Projects C and D, both having a quality rating of 'a', show the lowest relative measurement effort.

Discussion over the related assumption:

A1. Measurement productivity: *There is a direct relationship between requirements quality and the relative effort for sizing these requirements.*

In this case, data suggests a direct relationship between requirements quality rating and the relative effort for sizing these requirements. However, further measurement data is needed to ascertain this assumption.

B. Software development relative effort

Table 4 presents functional size, project effort and requirements quality rating for each project.

TABLE 4. REQUIREMENTS, MANAGEMENT AND MEASUREMENT METRICS BY PROJECT

Project	#User-stories	Project Effort (staff-hours)	#FP	Project Size (CFP)	Relative effort (hours/CFP)	Quality rating
A	440	8610	70	715	12.0	b
B	97	5565	27	177	31.4	c
C	51	7595	33	303	25.1	a
D	70	4417	33	352	12.5	a
E	41	2289	30	129	17.7	b

Table 4 shows that there are more user stories than the number of different functional processes (FP) for all measured projects. While it is true that one functional process can result in multiple user stories, it was observed that some of these US were used to manage change requests. Examples include user stories that captured new functionalities and change requests at the same time, but developed in different sprints. Since the tool used for RE tracked the effort on a per US basis, distinguishing development effort from change request effort proved to be impossible. However, the total effort for the project would remain the same.

The observations made while applying the measurement process also showed that a majority of user stories did not meet the expected format proposed by Agile methodologies.

When analyzing the effect of the quality of documentation on the project effort, it is expected that higher quality ratings would yield more cost effective results.

Data from Table 4 shows that project B, with a documentation quality rating of 'c', has the highest project relative effort.

Project C, with a quality rating of 'a', shows the second highest project relative effort. However, this project included parameterization of a Client Relationship Management

(CRM) component. It is suspected that with more projects measured, this particular project would be identified as an outlier as the team could not apply the same software process due to the nature of the software and its constraints.

Projects A and D, with quality ratings of ‘b’ and ‘a’ respectively, show comparable relative effort. However, larger projects might distribute their initial project start-up effort over the whole project size. This phenomenon explains why Project A, being 203% the size of project D, has a comparable relative effort.

Discussion over the related assumption:

A2. Development productivity: *There is a direct relationship between requirements quality and the project relative effort.*

In this case, data suggests a direct relationship between requirements quality rating and the relative effort for sizing these requirements. However, further measurement data is needed to ascertain this assumption.

C. Idenfication of defects in the requirements

Functional size measurement requires that software artefacts such as user stories, data models and applications be used as inputs in the measurement process. Aside from the actual measures, the measurement process can also be used to produce a qualitative evaluation of the requirements engineering process. Six types of defects that were discovered during measurement are described in this section.

1) Inadequate functional decomposition

The COSMIC method requires that a project's requirements be mapped to the COSMIC model in order to be measured. In this case study, during this mapping phase, the measurer was able to identify inadequate functional decomposition in project requirements. It is expected that each user story be mapped to only one functional process. On the other hand, a functional process may be mapped to multiple user stories, the first of which creates the functional process while all subsequent user stories modify this existing functional process, as user requirements change over time.

In this case study, inadequate functional decomposition was observed as several user stories were mapped to multiple functional processes, making it difficult to manage the project scope from the user's viewpoint. Thus, the user expects delivery of functionalities, not user stories, even though he/she might be knowledgeable about the user stories and their content.

2) Inconsistent terminology for data groups/objects

The COSMIC measurement method requires that data groups be identified. A data group must correspond to a single object of interest. A correct identification of these data groups is required in order to obtain an accurate measure. Therefore, careful attention was put in the analysis of requirements and software artefacts in order to correctly identify unique data groups and objects of interest.

In this case study, several terms were used to refer to the same objects of interest in each project. From a requirements engineering perspective, these terminology inconsistencies qualify as requirement defects.

3) Inconsistent user story naming

User stories commonly document the ‘who’, ‘what’ and ‘why’ of a requirement, and are identified by a short name giving an overview of the underlying functionality it describes. This short name, while of little value once a user story has been fully detailed, is of importance for measurement estimation activities. It was found, while examining user stories, that some of the names could not be used as a direct link to a functional process without taking into account the details of the user story. User story names should at least contain the name of the functional process it maps to in order to be correctly measured. This is due to the fact that before the beginning of a project or a sprint, not all user stories have been detailed. User story naming must then be taken as a requirement issue to be addressed.

4) Unavailability of data modeling artifacts

Data modelling is one of the activities commonly conducted in the development of business applications or any data-driven application. However, some data modelling artefacts were missing from the requirements documentation. After consulting with project managers, it appeared that although data modelling activities had taken place, no permanent record of these activities had been kept. Without data modelling artefacts, a measurer, while conducting functional size measurement activities, can come upon a requirement artefact that cannot be measured without making assumptions about the data groups. When faced with such a situation, physical data models had to be extracted from their actual databases and then mapped back to the COSMIC model in order to extract the data groups or clearly identify them. These activities could have been avoided had the data modelling artefacts been available and stored with the project documentation.

5) User story inner inconsistency

As stated earlier, it is expected that each user story be mapped to only one functional process. The user story should also document the functional requirements of a new functionality or modifications to be made to an existing functionality. In this case study, some user story descriptions were used to track change requests submitted by the product owner within the same user story used to create the functionality. From a measurement perspective, this was not an important issue since it was relatively easy to distinguish between the functional size of the original functionality and the size of the change request. However, from a project management perspective, this practice makes it hard or even impossible to correctly manage the scope changes of a project or to track down rework effort resulting from change requests.

6) Inconsistent RE process

In order to derive estimation models that behave predictably, the development processes on which they rely should also behave predictably. It was observed that development processes within the organization varied depending on the project and/or team. This discrepancy can be circumvented by having different estimation models for different development processes. This means that when measuring or estimating the functional size of a future project, the development process that will be used has to be known, which might not be the case.

A more problematic observation has been made while observing the development process within a project. In some cases, the development process changed over time between the start and the end of a project. While modifications to existing processes are expected in an organization that pursues continuous improvement, these modifications have to be closely monitored, and the results have to be measured. Changes to existing processes that yield a decrease in the ratio between the development effort required and the functional size should be kept, while those that yield an increase of this ratio should be rejected from subsequent development cycles.

Discussion over the related assumption:

A3. Extensibility: *The usage of the COSMIC method can be extended to identify defects in software functional requirements from the industry.*

Through the identification of these six RE defect types, functional sizing of projects using the COSMIC method has shown that it can be extended as a means to identify defects in requirements from industry (i.e. uncontrolled from a scientific viewpoint) written in other forms than use cases as in [11].

VII. Other findings

A. Measurement challenges uncovered during this case study

1) Unavailable requirements information for some projects

Since this case study focused on projects developed less than two year ago, requirement artefacts from these projects were available. However, we discovered that some RE artefacts' level of detail were not suitable for an accurate measure, as the data model was missing. One of the ways to overcome this obstacle involves the reverse engineering of data models from the software's database. For some of the completed projects, the database servers had been decommissioned by the organization. Gaining access to these decommissioned servers involved too much effort for the organization. As such, the priority for the measurement of theses projects was lowered. As no data model will be available for some remaining project, it is expected that relative measurement effort will be higher than for those projects where the data model was available.

2) Unavailable key personnel for projects

Faced with incoherent or inconsistent RE artefacts, the most effective way to understand and measure these artefacts is to consult its project members. For some of the projects, the key members who could provide the most insight either had left the organization or were assigned to other projects and access to them was limited.

B. Preliminary estimation model and its usage

Figure 4 presents the preliminary estimation model that was derived from the measured functional size and collected project effort data. While not a valid model at this time due to the scarce data points available, it shows early sign of reliability.

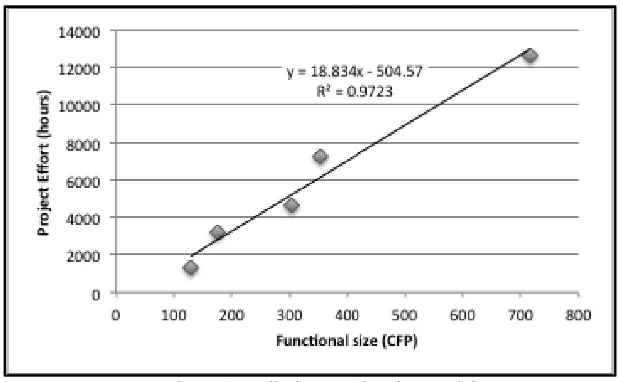

Figure 4. Preliminary estimation model.

It is of interest to report on its early use within the organization. When evaluating potential business opportunities, estimating the project effort at Axon-ID is currently an activity based on expert judgment. Managers at Axon-ID were interested in comparing the estimated effort using the estimation model in Figure 4 to the estimated effort based on internal expert judgment.

For a potential future project, the estimation activities from internal experts required the mobilization of two resources. The same estimation activities, using the estimation model, were conducted by one measurer over a period of six hours.

According to Axon-ID' manager, the difference between estimated project effort using the model was less than 10% of the average estimated project effort by internal experts. While this constitutes anecdotal evidence of the applicability of the model, managers were encouraged to pursue the measurement initiative and further data collection is on-going.

VIII. Threats to validity

The findings and data presented in this case study came from a single organization. At this time, it cannot be generalized that these findings apply to other organizations of a similar context. Further case studies are required at this time to refine the experiment design and validate potential generalization.

IX. Recommendations

Following the aforementioned findings, recommendations are targeting areas of improvement of the RE process and its outputs but also the measurement challenges uncovered during this case study.

1) Train functional analysts as measurers

As a means to reduce inadequate functional decomposition, functional analysts should be trained as measurers. This should lead functional analysts to align their RE activities with the subsequent measuring activities. Since the measuring activities mainly rely on the outputs of RE activities, such outputs should describe and correspond to what is expected from a measuring perspective.

2) Perform and record data modeling

As a means to reduce terminology inconstancies and also reduce the measurement effort, it is recommended that data modelling activities be conducted during each functional analysis activity of a project. In an Agile context, this implies that each sprint should result in a more refined and accurate logical data model of the client's business domain. These logical data models should also be part of the RE artefacts. This, in turn, should lead to increasingly precise estimation during the whole development life cycle. It will also decrease the effort required to measure the actual functional size of a project therefore enabling an easier refinement of estimation models.

It is expected that data models be kept as a project output on the organizational content management systems, as with any other software project and product artefact.

3) Organize the software improvement projects as any other project

While conducting this case study, it was found that although the organization was prepared to have researchers on site conducting measurement activities, they had not planned the impacts of such activities on their operations. As such, the introduction of unplanned activities had an effect on their previously planned operational activities. It is recommended that any improvement project be treated as any other project. Thus, resource allocation should be planned for and included in operational projects subject to process improvement activities, in order to sustain the measurement effort.

4) Include a measurer during sprint reviews and retrospectives

As stated earlier, measurement activities during any estimation phase of a project, whether done at the beginning of a project or at the beginning of a sprint, are of value. However, these measurements should also be validated once the software artefacts have been delivered. These validation activities serve a double purpose. Firstly, they can validate that the functional size of the product is aligned with the functional size of their corresponding requirements. Secondly, having a measurer present during retrospectives provides the team with a different source of input in their continuous improvement effort.

X. Conclusion and future work

As part of a broader research program that aims to improve methods for specifying requirements, a case study within a software organization was conducted. Software functional size, project effort and measurement effort were collected from projects within this organization. Quality of requirements documentation was also assessed. The results showed that inadequate functional decomposition, inconsistent terminology and documentation inconsistencies were revealed through functional sizing. Results suggested that a relation between the quality of requirements and project relative effort could be established. Finally, results also suggest that the quality rating used in this case study could be used as a predictor for measurement effort.

Two distinct axes of work are derived from this case study. The first axis concerns further measurement and analysis on the same case study. This includes measuring the remaining projects within the initial scope and collecting their project data. The second axis concerns work towards the broader scope of the current research program, where more experiments needed to be conduct in various software organizations applying different approaches to develop software. Collecting a larger sample of research data will bring us closer to our long-term objectives.

Acknowledgements

The authors are grateful to Yvan Bolduc, vice-president, and Jean-Pierre Parra, director at Axon-ID for generously sharing about their software process initiative and allowing this case study to be published. This work has been funded partially with a Mitacs grant, in partnership with Axon-ID.

References

[1] ISO/IEC 15504-2:2003 *Information technology — Process assessment — Part 2: Performing an assessment*, International Organization for Standardization, Geneva, Switzerland, 2003.

[2] Members Of The Assessment Method Integrated Team (2001), " Standard CMMI Appraisal Method for Process Improvement (SCAMPI), Version 1.1: Method Definition Document," Software Engineering Institute, Carnegie Mellon University, Pittsburgh, Pennsylvania, Handbook CMU/SEI-2001-HB-001. http://resources.sei.cmu.edu/library/asset-view.cfm?AssetID=5325.

[3] Derby, E., Larsen, D., and Schwaber, K. (2006), Agile retrospective: Making good teams great, Pragmatic Bookshelf, The Pragmatic Programmers series. ISBN: 0-9776166-4-9.

[4] CMMI Product Team, "CMMI for Development, Version 1.3," Software Engineering Institute, Carnegie Mellon University, Pittsburgh, Pennsylvania, Technical Report CMU/SEI-2010-TR-033, 2010. http://www.sei.cmu.edu/library/abstracts/reports/10tr033.cfm .

[5] Desharnais, J.-M., Abran, A., Assessment of the Quality of Functional User Requirements documentation using criteria derived

from a measurement with COSMIC- ISO 19761, International Workshop on Software Measurement – IWSM 2010, Stuttgart, Germany, November 2010, pp. 481-496.

[6] Lesterhuis, A., et al, The CSOMIC FSM Guideline for assuring the accuracy of measurements, The COSMIC Group, February 2011.

[7] Trudel, S. and Abran, A., Quick Reference Guide to the COSMIC Method for sizing Business Application Software, The COSMIC Group, 2012.

[8] Ungan, E. et al, An Experimental Study on the Reliability of COSMIC Measurement Results, In proceeding of: Software Process and Product Measurement, International Conferences IWSM 2009 and Mensura 2009, Amsterdam, The Netherlands, November 4-6, 2009.

[9] Trudel, S. and Abran, A., Functional Size Measurement Quality Challenges for Inexperienced Measurers, In proceeding of: Software Process and Product Measurement, International Conferences IWSM 2009 and Mensura 2009, Amsterdam, The Netherlands, November 4-6, 2009.

[10] Özkaya, A., Ungan, E., and Demirörs, O., Common Practices and Problems in Effort Data Collection in the Software Industry, In proceeding of: 2011 Joint Conf of 21st Int'l Workshop on Software Measurement and the 6th Int'l Conference on Software Process and Product Measurement, IWSM/Mensura 2011, Nara, Japan, November 3-4, 2011.

[11] Trudel, S., Using the COSMIC functional size measurement method (ISO 19761) as a software requirements improvement mechanism, Ph.D. Thesis, École de Technologie Supérieure, April 2012.

[12] The Institute of Electrical and Electronics Engineers, IEEE-Std-830-1998, IEEE Recommended Practice for Software Requirements Specifications, Software Engineering Standards Committee of the IEEE Computer Society, June 1998.

[13] Beck, K., et al (2001), Manifesto for Agile Software Development. http://www.agilemanifesto.org .

[14] Schwaber, K., and Beedle, M. (2002), Agile Software Development with SCRUM, Prentice-Hall. ISBN 0-13-067634-9.

[15] Beck, K., and Andres, C. (2004), Extreme Programming Explained: Embrace Change, 2nd Edition (The XP Series), Addison-Wesley Professional. ISBN : 978-0321278654.

[16] Kniberg, H. (2011). Lean from the Trenches: Managing Large-Scale Projects with Kanban, Pragmatic Bookshelf, The Pragmatic Programmers series. ISBN-13: 978-1934356852.

[17] Beck, K. (2002). Test Driven Development: By Example, Addison-Wesley, The Addison-Wesley Signature Series. ISBN-13: 978-0321146533.

[18] Cohn, Mike, Succeeding with Agile: Software Development using Scrum, Addison-Wesley, October 2009.

[19] Ambler, S. (2006), Introduction to Test Driven Development, Agile Data, http://www.agiledata.org/essays/tdd.html, Last accessed on March 23, 2014.

[20] Gärtner, M. (2012), ATDD by example: A practical guide to acceptance testing driven development, Addison-Wesley, The Addison-Wesley Signature Series. ISBN-13: 978-0321784155.

[21] A. Abran, J.W. Moore, P. Bourque, R. Dupuis, Guide to the Software Engineering Body of Knowledge (SWEBOK). IEEE Computer Society: Los Alamos, CA, 2004; 202 p. On Line at: http://www.swebok.org.

[22] S. Trudel and A. Abran, "Improving quality of functional requirements by measuring their functional size", International Workshop on Software Measurement -IWSM 2008, Metrikon 2008, and Mensura 2008, Springer, Munich, Germany, Nov. 2008, pp. 287-301, ISBN 978-3-540-89402-5.

[23] A. Abran, JM Desharnais, A. Lestherhuis, *et al*., COSMIC-FFP Measurement manual: the COSMIC implementation guide for ISO/IEC 19761:2009, version 3.0.1, Common Software Measurement International Consortium, January 2009.

[24] International Organization for Standardization, ISO/IEC 19761:2011, Software engineering -- COSMIC-FFP -- A functional size measurement method, Switzerland, 2011.

[25] A. Abran, Software Metrics and Software Metrology, IEEE Computer Society, Wiley, United States, 2010.

[26] International Organisation for Standardization, ISO/IEC 14143-5:2004, Information technology -- Software measurement -- Functional size measurement -- Part 5: Determination of functional domains for use with functional size measurement, March 2004.

[27] Desharnais, J-M., Bugra Kocaturk, and Alain Abran. "Using the COSMIC Method to Evaluate the Quality of the Documentation of Agile User Stories."*Software Measurement, 2011 Joint Conference of the 21st Int'l Workshop on and 6th Int'l Conference on Software Process and Product Measurement (IWSM-MENSURA)*. IEEE, 2011.

[28] Boisvert, Mathieu, and Sylvie Trudel. Choisir l'agilité-Du développement logiciel à la gouvernance: Du développement logiciel à la gouvernance. Dunod, 2011.

[29] Cohn, M. (2004), User Stories Applied: For Agile Software Development. Addison-Wesley Professional.