# Mining and Clustering Textual Requirements to Measure Functional Size of Software with COSMIC.

**Conference Paper** · January 2009

Source: DBLP



**3 authors:**

Ishrar Hussain
Concordia University Montreal


Olga Ormandjieva
Concordia University Montreal


Leila Kosseim
Concordia University Montreal

# Mining and Clustering Textual Requirements to Measure Functional Size of Software with COSMIC

**Ishrar Hussain, Olga Ormandjieva and Leila Kosseim**
Department of Computer Science and Software Engineering, Concordia University, Canada

**Abstract -** *The importance of early effort estimation, resource allocation and overall quality control in a software project has led the industry to formulate several methods for functional size measurement (FSM) of software, which are obtained from its requirements document. The main objective of this research is to develop a comprehensive workbench that would facilitate and automate early functional size measurement process from requirements documents written in unrestricted natural language. For the purposes of this research, we have chosen to use the COSMIC FSM method developed by the Common Software Measurement International Consortium (COSMIC) and now adopted as an international standard by the International Standardization Organization (ISO).*

**Keywords:** Effort Estimation, Software Requirements Specification, Functional Size Measurement, COSMIC, Text Mining, Natural Language Processing.

## 1 Introduction

Measuring correctly the functional size of the software and estimate correctly the effort are two key factors to successful software project management [7]. These variables available right from the start in a project would give the project manager an advantage about any future course of action, since many of the decisions made during development depend on, or are influenced by, the initial estimations. Measuring the functional size of the software has become in integral part of a software project where the functional user requirements of the Software Requirements Specification (SRS) document are primarily used. Starting from the size measurements and using the adequate mathematical equations, required effort and resources can be estimated precisely at the earliest phase of software development life cycle.

The importance of early effort estimation, resource allocation and overall quality control in a software project has led the industry to formulate several methods for functional size measurement (FSM) of software straight from its requirements document. Currently, there are five FSM models namely, NESMA [20], MKII [24], COSMIC [3,14], IFPUG [13], and FiSMA which are recognized as ISO standards. Size measures are expressed as single numbers function points (FP), designed to reflect how many of certain types of items there are in a system [22].

To the best of our knowledge, the existing literature does not report on automation of any of the existing FSM processes from SRS text. Similarly to all FSM methods, COSMIC method requires a pre-processing of the textual requirements to extract certain information pertinent to the COSMIC models, which is currently done manually by experts in COSMIC functional size measurement, which process is time consuming and highly specialized thus requires measurers' expertise and experience (a certification exam for COSMIC Expert measurer exits). The effectiveness of measuring the functional size in COSMIC largely depends on the accuracy of processing the textual requirements, which can be very tedious if the requirements are ambiguous or incomplete. SRS is one of the earliest artifacts produced in the lifecycle of software development. It is, in most cases, written in natural language and holds the clients' point of view of the software to be developed. The main objective of our work is to develop a comprehensive workbench that would facilitate and automate the process of extracting COSMIC models from the preprocessed SRS documents written in natural language organized and re-written in a complete and unambiguous fashion. Our proposed approach involves using a supervised text mining and an unsupervised text clustering techniques, in conjunction with specific natural language processing (NLP) tools, to extract automatically from the textual requirements all the information that is considered necessary in the COSMIC FSM. To our knowledge, ours is the first attempt automate the COSMIC FSM process from unrestricted natural language text.

The remainder of this paper is organized as follows: Section 2 provides a background on related work and the COSMIC method. Section 3 discusses the role of text mining and text clustering in the proposed research. Section 4 introduces the objectives and the methodology, and section 5 describes our tool support for text mining. Section 6 concludes the paper with a discussion on its applicability and the outline of the future work.

## 2 Functional Size Measurement with COSMIC

For the purposes of this research, we have chosen to use the COSMIC FSM method developed by the Common Software Measurement International Consortium (COSMIC) and now adopted as an international standard (ISO/IEC 19761 [14]). We chose this method in particular because it conforms to all ISO requirements (ISO 14143-1 [15]) for functional size measurement, focuses on the "user view" of functional requirements, and is applicable throughout the development life cycle from the requirements phase right through to the implementation and maintenance phases.

The process of measuring software functional size using the COSMIC method implies that the software functional processes and their triggering events be identified. In COSMIC, the unit of measurement is the data movement, which is a base functional component that moves one or more data attributes belonging to a single data group. It is denoted by the symbol CFP (Cosmic Functional Point). Data movements can be of four types: Entry, Exit, Read or Write. 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 subprocesses of each functional process are sequences of events; 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 attributes through software from a functional perspective.

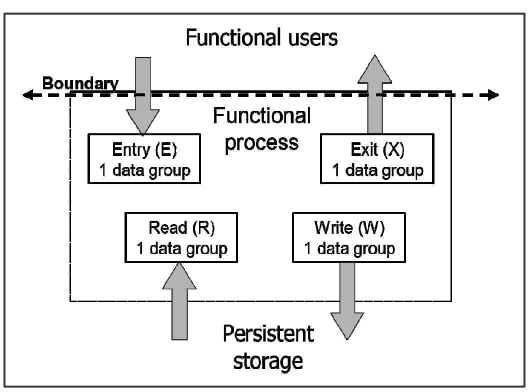


Figure 1: Generic flow of data attributes through software from a functional perspective [14]

## 3 Related Work

In recent years, there have been some notable studies in this area where researchers attempted to automate the process of functional size measurement. But, none of the studies to our knowledge addressed the problem where they would take the textual requirements as inputs to start the measurement process. Instead, they all relied on conceptual models to be manually built from the requirements, so that their automated approach can be adopted. For example, one of the leading work done in this area is by Diab *et al* [5], where they developed a comprehensive system called, µcROSE, which accepts state charts of as inputs to measure the functional size of real-time systems only. We find their work to be largely dependant on a set of hard-coded rules for mapping different objects of interest to different COSMIC components, and also require C++ code segments to be attached with the state transitions and supplied as inputs too, so that data-movements can be identified. They presented a very brief validation of their work by an expert, testing their system against only one case study, where it performed poorly in detecting the data groups, resulting in some erroneous measurement outputs. Candori-Fernández *et al* [2], on the other hand, performed another study, where they presented step by step guidelines to first derive manually the UML modeling artifacts, e.g. the use case models and system sequence diagrams from the requirements, and then, apply their set of rules for measuring the COSMIC functional size of the system from the UML models. Their approach was validated on 33 different observations, showing reproducible results with 95% confidence.

## 4 Role of Text Mining and Text Clustering

An effective application of COSMIC FSM process on SRS documents requires the measurer to overcome some crucial challenges. Our work addresses most of these challenges by means of text mining and text clustering techniques.

Text mining is an Information Extraction technique that is generally applied to classify textual information into several given categories, based on statistically learned patterns. Whereas, text clustering breaks a textual document into groups having common patters within. Although recent publications in the area of extracting information from SRS documents [1,4,8] have used these natural language processing techniques successfully, none, to our knowledge, so far applied these techniques to overcome the challenges of applying COSMIC FSM process on SRS documents.

In this section, we will list these challenges along with a brief explanation of how text mining and text clustering can provide vital solutions.

## 4.1 Ambiguity in SRS

Since SRS documents are commonly written in form of unrestricted natural language texts, they may suffer from the problem of ambiguity that is inherent to natural language specification [10,21]. Thus, to apply COSMIC on unrestricted SRS text, the measurer has to face the challenge of cleaning up the text from all its ambiguities, which can be monotonous for long SRS documents when done manually, resulting in poor quality SRS documents contributing to poor quality measurement results for COSMIC.

To combat this challenge, our previous work [10,21] showed that using a trained text mining system we can successfully classify requirements text into ambiguous and unambiguous sentences. As requirements development process is iterative in nature [12], such text mining system can aid the requirements analyst to extract the ambiguous sentences so that they can be elaborated or rephrased or dropped by the specifier and then reanalyzed by the system. Thus, the iteration continues until no ambiguous statements are left in the document, proving a good quality SRS document, ready for COSMIC FSM to be applied.

On the other side, the incompleteness or incorrectness of the requirements may become an issue when measuring the functional size. A methodology for a semi-automatic validation of the requirements in an interactive manner was proposed by the authors in [18] with the aim of reducing the incompleteness and incorrectness of the requirements.

## 4.2 Extraction of FUR

Since COSMIC method only works on functional user requirements (FUR), the next challenge that a measurer faces is to extract the functional user requirements manually from the SRS document, where both functional and non-functional requirements can be mixed together in same paragraphs.

Our previous work [11] showed that, again, a text miner can effectively classify the requirements text into functional and non-functional user requirements with high accuracy. Thus, it can help the measurer extract the FUR automatically, so that COSMIC FSM can be applied.

## 4.3 Identification of Functional Processes

In a collection of unrestricted functional user requirements text, the sentences may not appear in groups that correspond to individual functional processes. Thus, the COSMIC FSM process leaves the task on the measurer to group these sentences manually corresponding to each of the functional processes.

Our approach documented in this paper will present the idea of using an unsupervised text clustering technique to do the same task.

## 4.4 Identification of Data Groups

Identification of data groups and the mapping of requirements sentences to different data groups are also tasks that a measurer has to do manually. Here, we propose a text clusterer again to aid the tasks.

## 4.5 Classification of Data Movements

A measurer finally has to classify the requirements sentences into four different types of data-movements: (1) Entry, (2) Exit, (3) Read, and (4) Write — where one sentence can belong to multiple classes. This is again a classification problem that we propose to deal with supervised text mining technique. Also, we send the sentences, which fail to get classified by a well-trained text miner into any of the four classes, back to the specifier indicating that they are not in the appropriate level of granularity (LoG), as trained to the miner. Thus, the next iteration can be more accurate in the classification task.

Thus, having all these information extracted from the SRS, we can automatically measure the functional size of software using the COSMIC method from its SRS document. Section 4 will explain the process in details.

# 5 Research Objectives and Methodology

The objective of our work is to develop a comprehensive workbench that would to facilitate the process of extracting all the required information for calculating the COSMIC size measure of a specific software automatically from its SRS document. Setting this target, we select the following sub-goals related to achieve our main objective:

1. Our system will accept a requirements document of any quality as its input. Therefore, we can have text containing ambiguous requirements, which we will be filtering out automatically before further processing.

2. To generate COSMIC models we need to extract the Functional User Requirements (FUR) statements only. Therefore, if Non-Functional User Requirements (NFR) statements exist mixed together with FUR statements an SRS document, our system will automatically separate the FUR's from the NFR's.

3. To calculate the COSMIC size, we need to count its function points for each of its functional processes. To locate the group of statements in the requirements text that belong to the same functional process, we will have to devise a requirements clusterer and include it with our system.

4. To calculate the COSMIC size, we need to count its function points in CFP, which requires the statements expressing four different types of data-movements

(Entry, Exit, Read and Write) to be extracted and counted. We, therefore, will have to develop classifiers that can extract the Entry, Exit, Read and Write statements separately, so that the corresponding CFP can be counted later.

The methodology of our work follows six distinct steps. We will be presenting them here in details.

## 5.1 Resolve Ambiguity

First, we will be using the Ambiguity Checker tool, as presented in [10,21] to facilitate a semi-automated environment to detect and resolve ambiguity in the SRS text. The tool outputs a collection of sentences in the document as Unambiguous, and the rest, classified as Ambiguous, are fixed manually by the requirements specifier. Thus, the document moves to the next iteration resolving ambiguity on the way, and the process ends when all the sentences in the document are classified as Unambiguous.

## 5.2 Extract FUR

We take the unambiguous SRS text that we received from the previous step as input to this step, and we use the text miner, presented in [11] to classify SRS text into functional user requirements (FUR) and non-functional user requirements (NFUR). Thus, set of FUR sentences are extracted fully automatically that already demonstrated a very high accuracy in the results, shown in [11].

## 5.3 Identify Functional Processes

We developed a text clusterer [16] that cans automatically group sentences in an SRS document into sets of sentences that correspond to individual use cases. Since COSMIC functional processes are, in some manner, analogous to use cases, we will be applying the same clusterer on the unambiguous FUR sentences that we received from the previous step to group them into different sets of sentences, where each set corresponds to a distinct functional process.

## 5.4 Classify COSMIC Data-movements

For each of the sets of sentences belonging to a distinct functional process that we received as an output from the previous step, are then classified into four sets: “ENTRY”, “EXIT”, “READ” and “WRITE”. Here, a sentence can belong to more than one class, and, again, a sentence can below to none of the classes.

To solve this multi-class classification problem, we will be developing four different text miners: (1) ENTRY classifier, (2) EXIT classifier, (3) READ classifier, and (4) WRITE classifier — where each of these will be trained by its corresponding binary corpora, generated from the data collected by the online annotation system, as described in section 3. Thus, each of the four text miners will receive the same set of sentences as input, and will classify them into positive and negative classes. Thus, the ENTRY classifier, for example, will output two classes of sentences: (1) sentences that were used to express ENTRY data-movements (i.e. positive), and (2) sentences that were NOT used to express ENTRY data-movements (i.e. negative). The same holds for the other classifiers as well. Therefore, this will allow one sentence to be classified as positive by one or more classifier, allowing it to be classified into more than one class. On the other hand, this also allows one sentence to be classified as negative by all the classifiers, indicating that the sentence is not at the appropriate level of granularity (LoG) to be classified by any of the classifiers. We will also provide an interface, so that the measure can fix these sentences and rerun the step.

## 5.5 Identify Data Groups

Again, for each of the sets of sentences belonging to a distinct functional process, as extracted by the step described in section 4.3, we use a text clusterer along with a co-reference resolver to identify the different data groups participating in the functional process, and the mapping of its sentences to each of these data groups.

## 5.6 Count CFP

Finally, for each of the sets of sentences belonging to a distinct functional process, as extracted by the step described in section 4.3, we multiply the number of sentences in each of the four types of data-movements with the number of data-groups that maps to those sentences to get the count of COSMIC Function Point (CFP) for that type of data-movement within that functional process. We repeat this step for each of the four types of data-movements to get their respective counts of CFP. Summing up the CFP’s of all four data-movements will result in the CFP count for that functional process.

CFP is the unit of functional size in COSMIC. We will repeat this step for each of the functional processes to get their respective CFP count. Again, summing them all up will indicate the functional size of the whole system.

Figure 2 illustrates our methodology performing cosmic size measurement from the functional requirements.

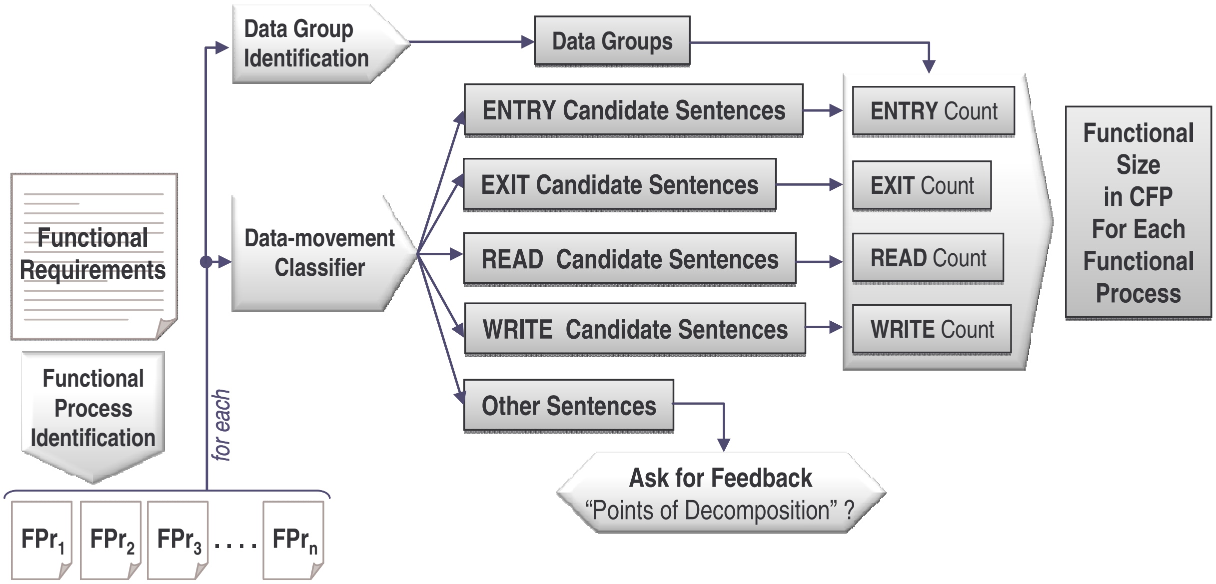


Figure 2: Our approach of automating the Functional Size Measurement process of COSMIC

## 6 Corpora Collection and Annotation

For the sub-goals 1, 2 and 4, mentioned in section 4, we will need to train the classifiers in a supervised learning approach. This requires a large number of requirements instances to be annotated by experts. Therefore, we are currently developing an online corpora collection and training data generation system that would allow requirements engineers and experts all around the world to contribute SRS documents and to collaboratively annotate requirements instances, so that the system can automatically generate data for training text mining systems that are to be used in the area of COSMIC FSM.

Figure 3 shows the architecture of the online annotation system that generates the training data for the text miners used in your work.

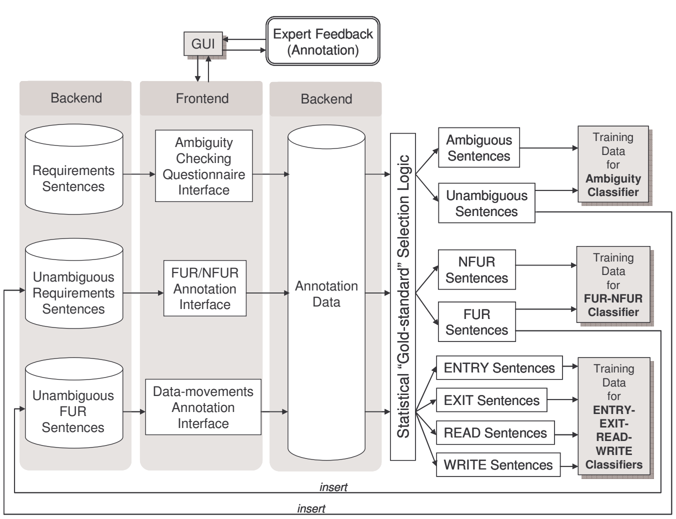


Figure 3: Architecture of Online Annotation System

## 7 Conclusions and Future Work

Measuring automatically the functional size with COSMIC from SRS documents as presented in our approach falls under the Count, Compute, Judge estimation technique [17], which means, basically, that the first course of action in measuring consists of counting and computing. If there is a way to directly count and compute some value to provide the estimate, this should be the best option, since it usually provides the most accurate result. Such accuracy can guarantee the quality of the measurement data and its consequent use, for instance, for the purposes of estimating effort or average productivity for a new project from size measured in CFP. In such cases, historical data on the effort and productivity of the organization with similar projects is required. In general, productivity is defined as a ratio of size (CFP) to effort (expressed in person-months, or pm). The historical average NFR productivity can then be used to obtain a good estimation of the NFR effort. For instance, if the average productivity is known to be 9 [CFP/staff-month], then the estimated effort for implementing the performance in the above example will be 27 [CFP] / 9 [CFP/ staff-month] = 3 [staff-month].

The research work presented in this paper will result in a new corpus which will be used for training the test classifier for COSMIC size measurement purpose. Our approach is applicable to the functional requirements operationalized through functions/ processes which could be mapped to the COSMIC model. Nevertheless, we note that the goal-oriented RE community [19,9,23] considers the importance of considering the effect of nonfunctional requirements on size and effort. Examples are global nonfunctional requirements like survivability, reporting, and customizability.

Our future work includes further investigation of the effect of the nonfunctional requirements on the software size measurement process and its automatic assessment. In order to validate our methodology, we will design an experiment similar to the approach adopted in [6] on the COSMIC case studies published in [18]; the validation will be addressed in our future work.

# 8 References


[1] Ambriola, V., Gervasi, V., "Processing natural language requirements," Proceedings of Automated Software Engineering (ASE'97): 12th IEEE International Conference, November 1-5, 1997, pp. 36-45.

[2] Candori-Fernández, N., Abrahão, S. & Pastor, O. (2007). "On estimation of the functional size of software from requirements". *Journal of Computer Science and Technology*, 22(3), May 2007, pp. 358-370.

[3] COSMIC, *The COSMIC Functional Size Measurement Method – Version 3.0*, The Common Software Measurement International Consortium, 2007. URL: http://www.gelog.etsmtl.ca/cosmic-ffp/updates/COSMIC_method_V3.zip (*Last accessed: May 11, 2009*)

[4] Cybulski, J. L., & Reed, K., "Requirements Classification and Reuse: Crossing Domain Boundaries", Lecture Notes in Computer Science, vol. 1844, Proceedings of Sixth International Conference on Software Reuse, Vienna, Austria, June 27-29, Springer-Verlag, 2000, pp. 190-210.

[5] Diab, H., Koukane, F., Frappier, M., and St-Denis, R., "$\mu_c$ROSE: Automated Measurement of COSMIC-FFP for Rational Rose Real Time", *Information and Software Technology*, 47(3), 2005, pp. 151-166.

[6] Díaz I., Moreno L., Fuentes, I., & Pastor O. "Integrating Natural Language Techniques in OO Method", Proceedings of the 6th International Conference on Intelligent Text Processing and Computational Linguistics (CICLing'05), LNCS Springer Verlag.

[7] Ebert, C., Dumke, R., Bundschuh, M., Schmietendorf, A., *Best Practices in Software Measurement: How to use metrics to improve project and process performance*, Springer, 1st ed., 2004.

[8] Fantechi, A., Gnesi, S., Ristori, G., Carenini, M., Vanocchi, M., and Moreschini, P., "Assisting requirement formalization by means of natural language translation," Formal Methods in System Design, vol. 4, Springer, 1994, pp. 243-263.

[9] Glinz, M., "Rethinking the Notion of Non-Functional Requirements", Proceedings of the 3rd World Congress for Software Quality, 2005.

[10] Hussain, I., Ormandjieva, O., & Kosseim, L., "Automatic Quality Assessment of SRS Text by Means of a Decision-Tree-Based Text Classifier". Proceeding. of the Seventh International Conference on Quality Software (QSIC 2007), pp. 209-218, 2007.

[11] Hussain, I., Kosseim, L., Ormandjieva, O., Using Linguistic Knowledge to Improve the Detection of Non-Functional Requirements Specifications. Lecture Notes in Computer Science, vol. 5039, E. Kapetanios et al. (Eds). Proceedings of the 13th International Conference on Applications of Natural Language to Information Systems (NLDB-2008), Springer-Verlag, 2008, pp. 287-298.

[12] IEEE, *Guide to the Software Engineering Body of Knowledge*, Abran, A. & Bourque, P. (Eds), IEEE Computer Society Press, 2004.

[13] IFPUG 4.1, Unadjusted Functional Size Measurement Method - Counting Practices Manual, ISO/IEC 20926:2003, first edition, 2003-10-01 www.ifpug.org

[14] ISO/IEC 19761, Software Engineering: COSMIC-FFP – A functional size measurement method, International Organization for Standardization – ISO, Geneva, (2003).

[15] ISO 14143-1, Functional size measurement – Definitions of concepts, International Organization for Standardization – ISO, Geneva, 1988.

[16] McConnell, S. (2006). Software Estimation: Demystifying the Black Art, Microsoft Press, 2006.

[17] Moradi Seresht, S., Ormandjieva, O., Sabra, S. (2008) "Automatic Conceptual Analysis of User Requirements with the Requirements Engineering Assistance Diagnostic (READ) Tool". Proceedings of 6th International Conference on Software Engineering Research, Management and Applications (SERA 2008), pp.133-142.

[18] Moradi Seresht, S., Ormandjieva, O., "Automated Assistance for Use Cases Elicitation from User Requirements Text", Proceedings of the 11th Workshop on Requirements Engineering (WER'08), 2008.

[19] Mylopoulos, J., "Goal-oriented Requirements Engineering", Keynote speech at the 14th IEEE International Conference on Requirements Engineering, IEEE Computer Society Press, 2006.

[20] NESMA, NESMA Functional Size Measurement method compliant to ISO/IEC 24570, 2006, www.nesma.nl

[21] Ormandjieva, O., Hussain, I., Kosseim. L., "Toward Text Classification System for Quality Assessment of Software Requirements Written in Natural Language". Proceedings of The 6th Joint Meeting of the European Software Engineering Conference and the ACM SIGSOFT Symposium on the Foundations of Software Engineering, 2007, pp. 39-45.

[22] Stensrud, E., "Alternative Approaches to Effort Prediction of ERP projects", *Journal of Information and Software Technology*, 43 (7), 2001, pp. 413-423.

[23] Wieringa, R., "The Declarative Problem Frame: Designing Systems that Create and Use Norms", Proceedings of the 10th IEEE Int'l Workshop on Software Specification and Design, 2000, IEEE Computer Society Press, 2000, pp.75-85.

[24] UKSMA, *Estimating with Mark II*, v.1.3.1., ISO/IEC 20968:2002(E), www.uksma.co.uk