MCReF: A Metric to Evaluate Complexity of Functional Requirements

Carlos Roberto Paviotti
São Paulo Federal Institute of Education, Science and Technology, IFSP
Capivari, Brazil
e-mail: carlinhos@ifsp.edu.br

Luiz Eduardo Galvão Martins
Institute of Science and Technology
Federal University of São Paulo, UNIFESP
São José dos Campos, Brazil
e-mail: legmartins@unifesp.br

Abstract— **The high sophistication of software systems has lead to an increase in the requirements complexity. Currently, there are metrics to evaluate the functional size of the software such as metrics of function point and use case points which are used with good results. However, a metric for the complexity for software requirements specifically had not yet been proposed. Identifying this gap, this paper proposes a Metric of Complexity of Functional Requirements (MCReF is an acronym composed by Portuguese words: *Métrica de Complexidade de Requisitos Funcionais*) indicated to evaluate and classify the complexity of software requirements. MCReF was developed from an empirical study based on a questionnaire that collected the opinion of 20 professionals from the requirements area to determine the weights of the factors that influence the requirement complexity. The responses were tabulated and given a statistical treatment to assess the weights of the complexity factors and their respective ranges of values for classification. A case study using MCReF is also presented in this paper.**

Keywords-Requirements Engineering; Complexity of Requirements; Requirement Metrics.

I. INTRODUCTION

Being part of the system engineering phases, Requirements Engineering consists of a set of techniques employed in the processes involved in the development of system requirements, i.e., eliciting, detailing, documentation and validation of the requirements [11]. The result of the set of requirements is a Software Requirements Specification Document, where the degree of understanding and accuracy of the provided description tend to be proportional to the degree of quality of the generated product. The definition of the software requirements occurs in the early development phases. Requirements Engineering provides methods, techniques and tools that help requirements engineers to define and classify what must be implemented in the software before starting building the system to be, i.e., the earliest phases of the software life cycle. Several processes models advocate such a procedure, for example: Requirements Definition in the Waterfall Model [13], Requirements Design in the Spiral Model [11], Requirements Gathering in the Prototyping Model [15], Requirements Workflow in USDP [13], etc. Among the ways of realizing the requirements complexity of a given system, regardless of the process model to be adopted, the Use Cases provide help in this issue, helping to formalize the scope of the system and facilitating the communication between developer teams and stakeholders. The presentation of requirements in a Use Cases Diagram is a simplified and less complex form of representation than the requirements description in natural language, enabling to estimate the project size and realize the system's complexity in a global way. Being one of the important factors to generate a software product with quality, a Software Metric corresponds to quantitative measures on one or more relevant features of the software [7][8][10], which allows developers to have a more refined view on the software process or related documentation, along with being an important management tool that contributes to preparation of time schedule, more accurate costs and more plausible goals, thus facilitating the decision making process and its consequent results.

Among the existing metrics, focusing on functionalities and not on a software system requirements, there are Function Points [13] and Use Cases Points Metrics [15], in both, the specified complexity factors are classified as subjective since they link the measures to "its value to the user".

Some related studies have been performed involving the requirements complexity, with presence in researches and empirical studies [12]. However, as many of them are focused on software quality, the necessity of involving the complexity factor in achieving the final result of the study remains, which generally refers to the system or project complexity in relation to their functionalities and not their requirements.

Kanjilal, Sengupta, and Bhattacharya [1] developed an approach based on metric model which aims to quantitatively estimate the requirements complexity for the object-oriented methodology, using project models like Sequence Diagram and Classes Diagram in the aid of validating the estimates in the project phases and long term project management.

Zhao, Tan, and Zhang [2] created a method to estimate costs through the requirements designing, proposing a new term named Path Complexity, which indicates a metric to measure the effort of the software complexity based on E-R Diagram (Entity-Relationship Diagram), showing the whole database structure in which an entity that can reach other entities due to its relationship and obtaining data on it.

Aiming the complexity related to requirements, an empirical study performed by Regnell, Svensson, and Wnuk [3] describes a case of system engineering in the field of mobile telephony, based on experiences used at Sony

 ISBN: 978-1-61208-304-9

Ericsson, which demonstrates the existing complexity of requirements in mobile telephones development.

The result of this study is called by the authors Very Large–Scale Requirements Engineering (VLSRE), suggesting a new order of magnitude applied to requirements, focusing on the size of the requirements set (the number of requirements is used, among other variables, to represent the complexity and it is strongly related to the nature of interdependencies among requirements), which are managed by a system developer company.

Complexity is an attribute that allows measuring if a software, usually part of it (module, method or function) is easy to read (comprehension), or else how complex it can become, if it contains a large number of nesting of laces and decision commands in a given program or functionality [8].

According to McCabe (1976) in Pressman [13] complexity is the quantification of the number of interdependent paths in a program, which provides an indication of its maintainability and testability. It is important to note that these definitions of complexity were built with the software as object in question and not the software requirements [10]. Another issue, also reported by Regnell, Svensson, and Wnuk [3] is that one of the factors responsible for the increasing of the requirements complexity is the large and diversified set of stakeholders, both internal and external to the organization. Based on the research performed in the literature and on the case studies, it is possible to characterize the requirements complexity as the degree of difficulty to interpret, specify, understand and implement a set of requirements, which is directly influenced by the amount of variables and procedures relevant to the requirements, as well as by the dependency relationships or coupling among them.

Currently, there is not available among the Requirements Engineering techniques, a metric aimed specifically to evaluate the requirements complexity. Such metric is of fundamental importance for the software development teams to have a reference concerning to the degree of complexity a requirement can present. Based on a metric of requirements complexity, the developer teams may build their own productivity indicators, which will be of great value to accurately estimate variables such as effort, time and cost of software development.

The aim of this study is to contribute to the software development in industries that employs the Requirements Engineering concepts and techniques, by proposing a metric to evaluate the complexity of functional requirements, even before start building the systems, in which this complexity is already recognized in the early phases of the life cycle of the software development.

To achieve the proposed metric, the adopted methodology was divided in four phases: (i) Development of case studies focusing the requirements elicitation, specification and validation, based on real contexts, including: a) Creation of a requirements specification document using the template *Volere*, referring to a system for monitoring and capturing heart rates to evaluate the heart autonomic function (in human beings); b) Creation of a requirements specification document using one of the templates from the *IEEE STD 830-1998* recommendation [9], regarding to the system for technical and physical monitoring of athletes in all the categories of a Brazilian professional soccer club [16]. These case studies were used as a "laboratory" to identify the factors that influence the requirements complexity. (ii) Creation of a Requirements Complexity Metric, identifying: a) main variables that influence the requirements complexity; b) Relationships among these variables; c) Weight of these variables, obtained through the application of a questionnaire to the software development professionals; d) Classification of the requirements complexity. (iii) Application of the proposed metric in three case studies which were software projects whose requirements had already been raised and previously documented. (iv) Analysis and discussion of the results obtained with the application of the metric in the case studies.

The rest of this paper is organized as follows: MCReF metric is explained in the section II. The empirical study that grounded the proposed metric is presented in section III. A case study is discussed in the section IV. Conclusions are presented in the section V.

II. MCReF Metric

A. Proposal

The revolution of software systems, where the increasing complexity and the size of their set of requirements are inherited factors of this progress, has motivated the improvement of already existing methods, techniques and tools in the Requirements Engineering.

Currently, there are metrics to estimate the software size and functionality [8][13][15], something that was a challenge to software companies in past decades. However, a metric for complexity of software requirements had not already been proposed. Motivated by such a gap, this paper presents the Metric of Complexity of Functional Requirements (MCReF).

MCReF is a metric proposed to evaluate the complexity of functional requirements, enabling to classify how complex is the functional requirement, focusing especially in information systems requirements. To apply the proposed metric it is necessary to obtain from the Requirements Specification Document, the generated artifacts or diagram, enabling to know the main factors that influence the complexity of functional requirements, namely: treatment and identification of functionalities, input and output variables, dependencies and couplings, decompositions, constraints and number of stakeholders involved in. Once performed the identification of these factors, it is necessary to specify them a little more, and thus to classify the sub-factors that influence the complexity of functional requirements on which is applied the weight attributed to each subfactor of complexity, enabling to obtain the degree of complexity in a single requirement.

B. Case study Development

To assist identifying the factors that influence the complexity of the information system requirements, two case

 ISBN: 978-1-61208-304-9

studies were carried out, each one having as a result a requirements specification document, being in different templates, which allowed a wider view of the functionalities and the objectives to specify and document the requirement correctly. The requirements specification documents included the following systems: (i) Monitoring and Heart Frequency Capturing System to evaluate the heart autonomic function (in human beings) developed in collaboration with Department of Physiotherapy at UNIMEP (Methodist University of Piracicaba – Brazil), using Volere template [14]; (ii) Technical and Physical Follow Up System to all categories of a professional soccer club in Brazil, which is discussed in a previous work [16].

C. Metric Development

Based on case studies performed to support the MCReF metric it was possible to identify in the Requirement Specification Document [16], the main factors of the complexity that influence the functional requirements, which are described in the following subsections.

1) Input and Output Variables

Represent values to be treated or used to meet the requirement represented by the identifiers, i.e., a label for each variable. They are classified as: (i) Input variable – existing variable in the requirement that will receive information from one agent or another system and making necessary to treat the value of this input, for example, an input variable of genre: "f" for female or "m" for male. (ii) Output variable – a variable of the result of the requirement. After processing the variable, the resulting information will be presented to the applicant and such value must be treated by the application, for example: the information "f" obtained from a field that stores data referring to genre must present the result "female" to the user requesting. It is possible to identify this factor of complexity in the Requirement Specification Document due to: the large number of variables, which will possibly have a greater complexity when comparing to requirements with a few variables, because these, whether input or output, need to be treated to present the results they were intended; the amount of constraints on the variables of the requirement, for example: input variables where the date of birth cannot be greater than or equal to the current; height and weight cannot receive negative values; output variables where age is obtained from date of birth stored; etc. Among the artifacts produced in a Requirements Specification Document, there is the factor of complexity in analysis in: Class Diagram, identifying the attributes of classes; Data Flow Diagram, obtaining the amount of data (input, output, query, internal file and external file); Entity-Relationship Diagram, identifying the attributes of the Entities and the attributes of the Relationships; Context Diagram, through the amount of data sent or received by the external entities, among others.

2) Number of Types of Stakeholders Involved

As reported by Regnell, Sevensson, and Wnuk [3], one of the factors responsible for the elevation of the complexity in Requirement Engineering is the large and diversified set of stakeholders, both internal and external to the system. However, regardless of the counting of stakeholders, there is a need of classifying these types involved.

It is possible to identify in the Requirements Specification Document such factors of complexity due to: number of actors representing given types of stakeholders – possibly a wide range of stakeholders attributed to the requirement will have a greater complexity when comparing to requirements with fewer stakeholders involved, because these will be related, at least, with one system functionality, demanding to be treated to present the results intended; quantity of existing hierarchic levels for the actors – each hierarchic level created indicates the need to specify and treat the available functionalities.

Among the artifacts produces in a Requirements Specification Document, there is the factor of complexity in analysis in: Use Cases Diagram, represented by the Actors and Hierarchic Levels existing among the actors (generalization relationships).

3) Number of External Interfaces

The external elements, with which the software in question must interact, such as IN/OUT hardware or even other systems, are considered external resources to the software and must be treated at the requirement level. It is possible to identify the influence of this factor of complexity analyzing: number of actors representing devices, such as sensors, actuators, etc. which demand treatment to interact with the system; number of actors representing other systems; other software or systems that receive or send information to the software in question. Among the artifacts produced in requirements specification, there is the factor of complexity in analysis in the Use Cases Diagram through the identification of the Actors and Data Flow Diagram by means of external and internal entities.

4) Functionalities Identification/Treatment

Functionality can be defined as a behavior or an activity for which a beginning and an end can be viewed, that is, something capable of being executed. For example, the simple execution of a functionality called "perform order" refers to the activities to be performed (create order, verify customer, link product, verify stock, calculate discount, define delivery time, etc.) resulting in the creation of an instance of the entity/class called "Order". It is also recommended to present, in the description, the set of preconditions (for example, customer already registered), to implement functionality, and post-conditions (product delivered, product warranty after sale etc.) which may arise from this implementation.

It is possible to identify in the Requirements Specification Document this factor of complexity by analyzing: the number of existing functionalities to perform a requirement; necessary conditions set out in the requirement preconditions, necessary conditions set out in the requirement post-conditions, requirements that involve dependency or coupling of the functionality of other requirements. Among the artifacts produced in a Requirements Specification Document, there is the factor of

 ISBN: 978-1-61208-304-9

complexity in analysis in: Classes Diagram, represented by the operation of classes; Data Flow Diagram, represented by the processes; Use Cases Diagram represented by the Use Cases, Requirement Specification Form, obtained from the conditions to perform a requirement; number of validations to perform a requirement, number of results obtained from the performance (main flow, requirement alternative(s) and exception (s)), among others.

D. *The weights of Factors of Complexity and their Subfactors*

In Table I, the factors and subfactors of the complexity proposed for MCReF are presented along with their respective weights, obtained from the results of the empirical study performed with 20 professionals from the requirements area. The factors and subfactors are objects of study and were obtained through bibliographic review of the Requirement Engineering area along with the development of case studies focused on requirements elicitation, specification and validation based on real context, among them: a) Creation of a requirement specification document, using the template Volere, referring to a monitoring and collection of a heart rate system to assess the autonomic function of the heart (in human beings); b) Creation of a requirement specification document using the templates recommended by IEEE STD 830-1998, referring to a technical and physical monitoring of athletes system on all categories of a professional soccer club [16]. To define each Weight Attributed to the Subfactors of Complexity of the Requirement, as presented in Table I, it was necessary to base on the responses obtained on the empirical study conducted with the professionals from the area. Based on the responses obtained from this study, the arithmetic average of the respondents answers were obtained for each subfactor of complexity and thus defining the subfactor Average.

TABLE I. WEIGHTS OF THE FACTORS OF COMPLEXITY OF THE REQUIREMENTS

Factor/Sub-Factor	Sub-Factor Average	*Weight attribute to the sub-factor of complexity of the requirements*
Q1 – Input and Output Variables		
Q1.1 – Number of Input Variables	3.60	***0.85***
Q1.2 – Number of Output Variables	3.30	***0.78***
Q1.3 – Number of Constraints of Input Variables	3.90	***0.92***
Q1.4 – Number of Constraints of Output Variables	3.60	***0.84***
Q2 – Stakeholders		
Q2.1 – Number of human actors	3.65	***0.86***
Q2.2 – Number of hierarchic levels	2.85	***0.67***
Q3 – External Interfaces		
Q3.1 – Number of actors that represent devices	3.15	***0.75***
Q3.2 – Number of actors that represent others systems	3.40	***0.80***
Q4 – Functionalities		
Q4.1 – Number of Functionalities	4.10	***0.97***
Q4.2 – Number of hierarchic levels of Functionalities	3.80	***0.90***
Q4.3 – Number of Pre-Conditions	3.55	***0.84***
Q4.4 – Number of Post-Conditions	3.40	***0.80***

To define the weight attributed to the subfactor of complexity, it was necessary to conduct, for each one, a division of the average of the subfactor obtained by the sum of the subfactors of complexity generated. With the value of the assessment of each factor of complexity, it is obtained the result, which must be multiplied by 10 (ten), to be applied in a 0-10 scale, as suggested by the metric proposed. During the empirical study, it was needed to define a weight to the factors of requirements complexity along with their subfactors of complexity, however, it was verified that only the responses attributed to the subfactors of requirements complexity would be of real interest, discharging the values obtained to the factors of requirements complexity.

The amount identified of each subfactor of requirement complexity must be multiplied by the weight attributed to the Subfactor of Complexity (SfC), resulting in the Complexity of the Subfactor of the Requirement (CSfR) and allowing them to receive their respective classification of complexity. The degree of importance of the composition to the subfactor of requirement complexity, in this study called weight of the subfactor, is the result of the empirical study conducted with the professionals of the area.

The classifications of the CSfR is the result of empirical tests conducted, and the rating value “Low” was assigned by the MCReF’s developers, based on their professional expertise; “Medium” corresponds to twice the value attributed to low classifications, “High” corresponds to higher values than the average and less than “inappropriate”. The classification “Inappropriate” indicates that the amount of elements defined for the SfC in the requirement multiplied by the weight of the factor of requirement complexity exceeds the value attributed to the value “high”. For the complexity of the subfactor of the requirement that is not identified or used in the requirement, there should be used a value of zero (0). In case there is not a CSfR classified as “Inappropriate”, it is possible to obtain the classification of the requirement by the sum of the complexities of the subfactors referring to the requirement in question, thus obtaining a “Complexity of the Requirement” (CR). This Complexity of the Requirement must be related with Table II to receive a Classification of the Complexity of the Requirement (CCR). When the CSfR is classified as “Inappropriate”, it is recommended to restructure the requirement or, “Complexity Inappropriate Requirement” must be attributed to the requirement in question, i.e., it will maintain the structure of the functional requirement in analysis, even with one or more subfactors of complexity classified as inappropriate. All Complexity of Inappropriate Requirement (CiR) indicates that one or more subfactor of complexity of the requirement was diagnosed as a number of elements defined for the SfC of the requirement that, when multiplied by the weight of the factor of requirement complexity, exceeds the value attributed to the classification “High”, then this requirement is given the Complexity Inappropriate Requirement (CiR) and its weight is the highest value shown in Table II multiplied by the number of times the SfC of requirement for the functional requirement in question was classified as inappropriate. Therefore, the **Complexity of the Requirement** is obtained by the result of the sum of the CSfR and its **Classification of the Complexity of the Requirement** is achieved through the application of the Complexity of the Requirement checked with Table II. The Classification of the Complexity of the Requirement (CCR) is the result of empirical tests grounded on the development of case studies focused on elicitation,

 ISBN: 978-1-61208-304-9

specification and validation of requirements based on real contexts [16]. To define the classification as "Very Low" it also takes under consideration the classification "inappropriate" where both have a scale of 10 (ten) points, i.e., less than 10 points are classified as "Very Low" and the 10 points less than 100 points are "Inappropriate".

TABLE II. CLASSIFICATION OF THE COMPLEXITY OF THE REQUIREMENT

CCR		
From	To	Classification
0	10	Very Low
11	26	Low
27	42	Middle Low
43	58	Middle
59	74	Middle High
75	90	High
91	.....	Inappropriate

III. THE EMPIRICAL STUDY THAT GROUNDED THE PROPOSED METRIC

The empirical study, which aimed the application of a questionnaire concerning to the requirements complexity identified along with the professionals of the area provided the database to obtain the weights for each factor of complexity studied. The results are shown through the following analysis: data from the participants, degree of importance attributed to the factors and subfactors of requirements complexity and reliability of the instrument of data collection.

A. *Data from the participants*

It was possible to obtain a profile of the interviewed through the part of the questionnaire "Professional Identification". The results indicated that 100% of the participants in the empirical study were professionals with a high level of academic education, distributed in master (30%), mastering (55%) and Ph.D (15%). Regarding the time working in the area of requirements, 80% of the participants have carried out activities for 5 years or more, while only 10% has had less than a year in the area.

B. *Degree of importance attributed to the factors and subfactors of complexity of the requirement*

For the specific purpose of obtaining weights to the factors and subfactors of complexity, it was used the basic tool for data collection: a questionnaire consisting of 4 factors subdivided in 12 subfactors with 5 alternatives each, whose measures were based on Likert scale [6]. The factors considered in the empirical study were obtained by reviewing the literature about the complexity of requirements and also by the case study developed along the research using the templates Volere and IEEE STD 830-1998 to document the requirements with the factors: input variables and output of the system, Stakeholders, external interfaces to the system and system functionalities. Through this instrument to collect data, the participants were able to express their opinion about each of the affirmatives.

C. *Analysis of the Reliability*

Finished the tabulation of the research data using the statistic software SPSS (Statistical Package for the Social Sciences- version 13.0), the instrument used to collect data was subjected to a reliability evaluation through Cronbach's Alpha coefficient analysis which works the relationship between internal covariance and variances of the measures. The value of Alfa can range between zero and one (0 - 1) and the higher this value, the greater the internal consistency of the instrument evaluated. Authors differ on the minimum acceptable value to Cronbach's Alpha Coefficient. Hair et al. [4] said that to have an acceptable reliability, Cronbach's Alpha must have a value of at least 0.70. However, as this is not considered an absolute value, lower values are accepted if the research is exploratory in nature. According to Malhorta [5], the minimum value of Cronbach's Alpha to ensure the reliability in a research must be 0.60.

Using Cronbach's Alpha in this study aimed to evaluate the internal consistency of the instrument used (questionnaire), and check if there is consistency in the variation in the participants' responses, examining each factor and subfactor of complexity considered in the research. Table III presents the results of Cronbach's Alpha coefficient for subfactors grouped by their factors of requirement complexity in question, i.e., involving Q1.1, Q1.2, Q1.3 and Q1.4 for Input and Output variables, Q2.1 and Q2.2 for Stakeholders, Q3.1 and Q3.2 for External Interfaces and Q4.1, Q4.2, Q4.3 and Q4.4 for functionalities.

According to the presented in this table, it is possible to observe the Alpha values obtained for each one of the factors of complexity considered in the empirical study. It is observed that the lower Alpha value produced was for the factor of Input and Output Variables (0.532) and the highest result was for the factor External Interfaces (0.834). Analyzing the general Alpha and considering all factors, it is noticed that the value generated was very satisfactory. The result indicates that the instrument used in the research is highly reliable since reached a maximum value of 1 (one), an Alpha of 0.808 was obtained. This value can be presented as an indicator of efficiency and reliability of the instrument in evaluating the factors of requirement complexity.

TABLE III. RESULTS OF CRONBACH'S ALPHA FOR FACTORS OF COMPLEXITY OF REQUIREMENT

Statistics by scale	Average	Variances	Standard Deviation	Number of variations
	3.5208	0.9117	0.9548	4
Cronbach's Alpha for Factors of Complexity				
Input and Output Variables				0.532
Stakeholders				0.759
External Interfaces				0.834
Functionalities				0.718
General Cronbach's Alpha				0.808

IV. CASE STUDY

The intent of this section is to present the applicability of the metrics of complexity of functional requirements – MCReF - in a case study. The context of such study was a system to monitor and capture heart rate to evaluate the autonomous function of the heart.

 ISBN: 978-1-61208-304-9

A. *Monitoring and Heart Rate Capturing System*

The documentation of requirements specification referring to the Monitoring and Heart Rate Capturing System to evaluate the autonomous function of the heart (in human beings) was developed by students of the Computer Science Master Degree at UNIMEP – Methodist University of Piracicaba, Brazil - related to the practical work using the Template Volere and presented to the discipline of Requirements Engineering. The documentation consists of 21 functional requirements, 15 new ones and 6 from the previous system. Table IV shows the results of the application of MCReF.

TABLE IV. RESULTS OF THE APPLICATION OF MCREF – MONITORING AND HEART RATE CAPTURING SYSTEM

ID	Requirements	COMPLEXITY	INAD	CLASSIFICATION	COMPL INAPPR
FRN001	Keep patient's basic data		1	Inappropriate	91,00
FRN002	Create profile of patient's registry	32,49		Middle low	
FRN003	Create profile of signal collecting	19,11		Low	
FRN004	Create experimental protocol	19,28		Low	
FRN005	Perform experiment		1	Inappropriate	91,00
FRN006	Maintain experiment	5,21		Very low	
FRL007	Collecting signal	17,08		Low	
FRN008	Create profile of signal cleaning	8,73		Very low	
FRL009	Analyzing graphic signal	6,90		Very low	
FRL010	Cleaning signal	10,68		Low	
FRL011	Statistically analyze the experiment	5,93		Very low	
FRN012	Maintain signal collecting	16,24		Low	
FRN013	Create items (blanks) for patient profile	8,42		Very low	
FRN014	Define identification of experiments	2,47		Very low	
FRN015	Check the experimental protocol	9,83		Very low	
FRL016	Generate graphic time domain	11,13		Low	
FRL017	Generate graphic frequency domain	11,30		Low	
FRN018	Testing beep	5,23		Very low	
FRN019	Maintain physiotherapist		1	Inappropriate	91,00
FRN020	Maintain material data		1	Inappropriate	91,00
FRN021	Maintain equipment data		1	Inappropriate	91,00

Legend:
FRN – New Functional Requirements
FRL – Legacy Functional Requirements

1) *Analysis and Discussion of the results obtained with the application of MCReF in the Monitoring and Heart Rate Capturing*

Investigating the subfactors that classify FRN001 complexity as "inappropriate", it is observed that the subfactor "number of functionalities", which presents 21 functionalities, multiplied by the weight 0.97 results to the subfactor a complexity equal to 20.37 (weight adopted according to Table I), which is higher than the stated in the classification of complexity given to the subfactor applied in the metrics, i.e., higher than 5 and less than 10.

In the analysis of the subfactors that classify the complexity of FRN002 as "middle low", it was observed that the subfactor "number of input variables" stated with 22 variables, which multiplied by the weight 0.85 results in a complexity of 18.7 to the subfactor defined as "High" in the classification of complexity.

Besides this subfactor, it was found that the subfactor "Number of Constraints to Input Variables" presents 7 variables, which multiplied by the weight 0.92 generates a complexity of 6.44 to the subfactor also defined as "High" in the classification of complexity.

Evaluating the classifications of complexity produced by the MCReF from the experience of the analyzer considering their own productivity indicator, it is observed that the result of the application of the proposed metric reflects the reality in the implementation of a software requirement, i.e., the results of the complexity obtained for the requirements corresponds to the necessary resources identified for their development and enable their identification in functional requirements of factors and subfactors of higher complexity. It is also noticed that the results obtained with the application of MCReF assist in the tasks to estimate the effort (people and professional), time and cost for development, ranging from the functional requirement of lower complexity, the FRN014, until the highest complexity, the FRN002.

V. CONCLUSION AND FUTURE WORK

With the evolution of Software Engineering techniques, it became possible to improve the software quality through standardization and definition of development processes – in accordance with the requirements – to ensure a final product that meets the customer's expectations, as agreed.

The tasks of classifying and measuring software are present from the conceptual stage (requirements) to product delivery. However, little has been explored in the Requirements Engineering area about the use of metrics of complexity. Briefly, only two studies about the subject could be identified [1][3]. Currently, there is not available, among Requirements Engineering techniques, a metric aimed specifically to measure the complexity of requirements. Such metric is of fundamental importance for software development teams in industries to have references about the degree of complexity a requirement can present. Based on a metric of complexity of requirements, the development teams can build their own productivity indicators that will be of great use to predict, with precision, variables as effort, time and cost of software development. These requirements must preferably be specified in standard documents, based on, for example, the template *Volere* or templates available in *IEEE STD 830-1998* recommendation, allowing distinguishing their main features, artifacts or diagrams contained therein, namely: treatment of functionalities; input and output; dependencies or coupling, constraints and number of stakeholders involved. With the definition of the subfactors of complexity and their respective weights and classification, it has been applied in real requirements context already specified the metric of complexity proposed. With the complexity and classification obtained for the requirements it became possible to compare the results among requirements and check the efficiency of the proposed metric. For the specific purpose of obtaining weights to the factors of complexity, it has been used a basic instrument of collecting data, a questionnaire composed of

 ISBN: 978-1-61208-304-9

four factors of complexity, divided in 12 subfactors with 5 alternatives whose measures were based on Likert scale.

The factors considered by the empirical study were obtained through a literature review about the complexity of requirements, and also through the case studies developed along this research. Through the instrument of collecting data, the participants could express their opinion about each of the statements. The instrument used to collect data was subjected to an evaluation of reliability through Cronbach'Alpha coefficient.

Besides the evaluation of the general consistency of the instrument, Cronbach`s Alpha was employed to analyze each issue (factor and subfactor of complexity) considered in the research. Therefore, the current paper assists the development of system that use the Requirements Engineering techniques and concepts, through a metric of complexity of requirements, i.e., with the capacity of measuring how complex a requirement is, even before starting building it, identifying such complexity in the early stages of a software development life cycle. It is envisioned the possibilities of expanding this research and suggested as future works the development of a method to obtain the complexity of a set of existing requirements in a project, enabling classify the complexity of a system as a whole.

It is also suggested the development of a software to support the proposed metric. Besides such suggestions, this metric could: become a tool to estimate the cost of the software, because of the complexity involved in the requirement, being charged by the degree of difficulty for its implementation; predict the time of development of the requirement presented by the complexity associated to the resources required for implementation; estimate the delivery time of the modules of the system; establish the necessary resources (hardware, software, professionals, etc.) and qualify the software through the way of treatment of the requirement complexity. The study presented in this paper points out for the necessity of new researches in the Requirements Engineering metrics.

REFERENCES

[1] A. Kanjilal, S. Sengupta, and S. Bhattacharya, "Analysis of complexity of requirements: a metrics based approach", Proceedings ISEC'09, pp.131-132, Pune, India, 2009.

[2] Y. Zhao, H. B. K. Tan, and W. Zhang, "Software cost estimation through conceptual requirement"; Proceedings of the Third International Conference On Quality Software (QSIC'03), p.141, 2003.

[3] B. Regnell, R. Svensson, and K. Wnuk, "Can We beat the complexity of very large-scale requirements engineering?", Proceedings of the 14th International Working Conference on Requirements Engineering: Foundation for Software Quality (REFSQ 2008), pp.123-128, Montpellier, France, june/2008.

[4] J. F. Hair, Multivariate Data Analysis, 4th ed., New York: Prentice-Hall, 1995.

[5] N. K. Malhotra, Marketing Research: An Applied Orientation. New Jersey: Prentice Hall, 1996.

[6] R. Likert, "A technique for the measurement of attitudes". Archives of Psychology, 1932.

[7] B. W. Boehm and P. N. Papaccio, "Understanding and controlling software costs", IEEE Transactions on Software Engineering, p.1462-1477, vol. 14, no. 10, 1988.

[8] N. E. Fenton and S. L. Pfleeger, Software Metrics – A Rigorous and Practical Approach, 2nd ed., PWS Publishing Company, 1997.

[9] IEEE, IEEE Std 830-1998 Software Requirements Specification, The Institute of Electrical and Electronics Engineers, New York, 1998.

[10] S. H. Kan, Metrics and Models in Software Quality Engineering; Addison–Wesley, 2002.

[11] G. Kotonya and I. Sommerville, Requirements Engineering: Processes and Techniques; John Wiley & Son, Chichester, England, 1998.

[12] S. Park and J. Nag, "Requirements management in large software system development"; IEEE International Conference on Systems, Man and Cybernetics, pp. 2680-2685, vol. 3, 1998.

[13] R. S. Pressman, Software Engineering: A Practitioner´s Approach; McGraw-Hill, 7th edition, 2009.

[14] J. Robertson and S. Robertson, Volere: Requirements Specification Template, Edition 14, 2009.

[15] I. Sommerville, Software Engineering; Addison Wesley, 9th edition, 2010.

[16] C. R. Paviotti and L.E.G. Martins, MCReF – Métrica de Complexidade de Requisitos. Revista Conteúdo, vol.1, no.6, pp.1-26, ago/dez 2011.

 ISBN: 978-1-61208-304-9