MEASURING SOFTWARE PROCESSES PERFORMANCE BASED ON FUZZY MULTI AGENT MEASUREMENTS

M.A. SEYYEDI, M. TESHNEHLAB AND F. SHAMS

ABSTRACT. The present article discusses and presents a new and comprehensive approach aimed at measuring the maturity and quality of software processes. This method has been designed on the basis of the Software Capability Maturity Model (SW-CMM) and the Multi-level Fuzzy Inference Model and is used as a measurement and analysis tool. Among the most important characteristics of this method one can mention simple usage, accuracy, quantitative measures and comparability. Fuzzy logic-based tools are designed to provide such functions.

1. Introduction

The productivity and quality of software products depend on three elements: people, technology and software processes [8]. The very different methods of management and development of human resources have a long history and numerous defined methods exist in this field. The fields of technology software production tools are indeed growing fast, following a specific path. Software processes are considered to be the most essential context for improving productivity and quality. However, software process models are constant undergoing development. The present article focuses on the methods of measuring the maturity and quality of software processes in one software organization. This method, which has adapted the SW-CMM model and applied it for requirement management key process area, can be generalized and applied in other processing areas as well.

Many studies have been carried out on software process measurement based on the SW-CMM model, of which the most outstanding one is the report provided by Baumert and Mcwhinnel [4]. This report includes a series of parameters (combined parameters) compatible with the key measurement and analysis practices of the SW-CMM model [8]. These parameters are categorized according to the quantitative properties of a product, and not on Key Process Areas (KPA). The Raynus Report points out the relationship between the qualitative numbers of processes and the SW-CMM model and emphasizes the fact so as to avoid confusion with the convention that measuring a process can enhance organizational behavior in a software producing organization [9].

This method does not exclusively discuss the goals defined in the key process areas of the SW-CMM model and is in fact a general and complex method of application in other areas as well. Nevertheless, the main objective of this article is to present a specific

---

Received: March 2005; Accepted: September 2005

Key words and phraces: Software capability maturity model, Goal/ Question / Metric method, Key process areas, Fuzzy system, Multi level fuzzy inference model.

measurement method for any key process areas which can be easily comprehended and applied [7].

2. The Requirement Management Process in the SW-CMM Model

The Software Capability Maturity Model (SW-CMM) is a model of software processes which provides guidelines required by software organizations for constant improvement at the different levels defined therein.

This model consists of five levels as shown in Figure 1: Initial, Repeated, Defined, Managed and Optimized. Each level includes various key process areas. This article discusses the requirement management process, one of the key process areas of the SW-CMM model, as a sample process to fully carry out the measurements of the chosen model.

The requirement management process aims to create mutual understanding between customers and the software project organization with respect to their needs. In other words, these requirements must be comprehensive, documented and controllable for the designed software product to meet the needs of customers. In many cases, these requirements may vary throughout the software lifecycle, while the newly originated demands cannot be controlled adequately. The requirement management process activities include the collecting and compiling of information, arriving at agreements with customers as well as analysis, review and management of alterations in requirements. In other words, such activities are performed to ensure that a software product meets the real requirements of its customers [8].

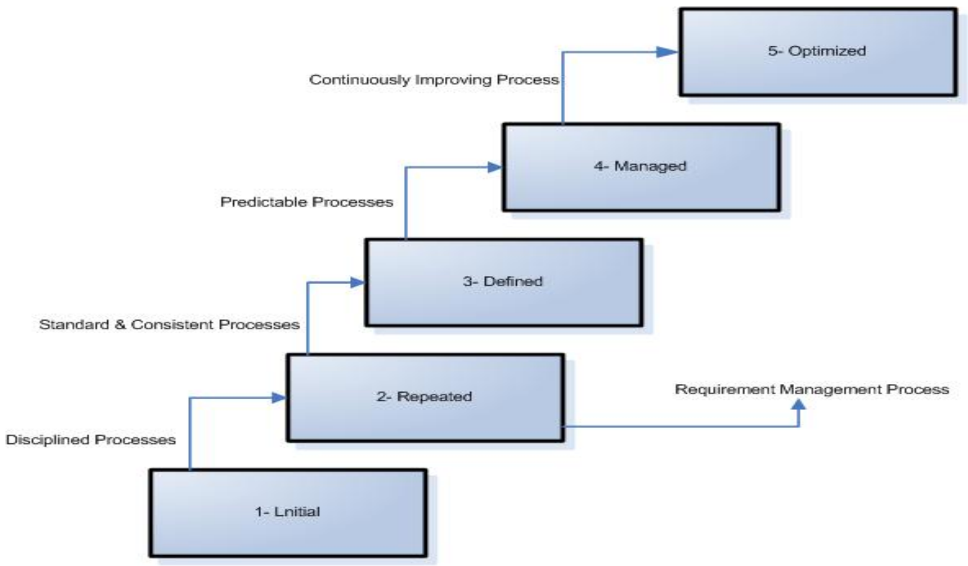

FIGURE 1. SW-CMM Key Process Areas

3. **The Goal/Question/Metric Method (GQM)**

The GQM method helps an organization focus on its goal measurements. According to this method, before the collection of input data, a series of predetermined goals are set for the organization, which are not of course definite and unchangeable. Based on these predefined goals a set of questions and measurable answers are designed. These questions are metric units and the pertinent answers form the necessary input data for measuring the achievement of the preset goals [10].

The GQM method consists of the following major stages:

- Defining a series of goals based on the requirements of the organization or processes as well as identifying the issues to be evaluated or optimized. The goals can be defined through three headings, namely the purpose, the perspective and the environment, shown in Figure 2.

The purpose: including the analysis of items such as processes, products and/or experimental models in order to distinguish, evaluate, predict and improve the following:

The perspective: takes the issues such as cost, elimination of problems, modifications, reliability & relationship with user, etc., into consideration from the viewpoint of the people involved such as the user, costumer, manager, software developer, company, etc.

The Environment: this is propounded in fields such as “problematic elements, human elements, resource elements, process elements, etc.”

FIGURE 2. Defining a Goal in the GQM Method

- Creating a set of quantitatively measurable questions according to the various guidelines provided for the classification of product-based or process-based questions. These questions are applied for the analysis of the relevant input data of different goals.
- Defining a series of metrics which provide the quantitative measurable information required to answer each question individually. At this stage, the metric units, which provide the necessary data to answer the predefined questions, are defined and attributed to each question. Nevertheless, a single predefined metric unit may be used to answer several questions. Therefore, the answer to each question is a combination of a series of predefined metric units. By defining the above stages once, the necessary input data to answer measurable questions are collected and analyzed in accordance with the organizational objectives and goals.

4. **The Applications of the GQM Method in Software Capability Maturity Model**

The GQM and SW-CMM can be easily combined with one another, as shown in Figure 3. The SW-CMM defines one or a number of goals for each key process area and all these goals can be used in the first stage of the GQM [5]. The requirement management process is chosen to analyze this issue. The CMM defines two separate goals for the Requirement Management Process. The first goal is described as follows:

"The requirements allocated to the software are to be controlled to establish a software engineering baseline and application management. [8]*"*

The above mentioned goal focuses on the creation of a controlling baseline for the requirements. The lack of control over existing requirements would make it impossible to offer a clear picture of how the final product could meet such requirements.

The second goal defined for the Requirement Management Process is as follows:

"The consistency of software plans, products and activities must be controlled with regard to the given and allocated requirements of the software."

This goal mainly focuses on establishing a control over the consistency between the requirements and the software product that has been produced according to the former. Such a consistency would enhance the product design and would meet customer demands.

In the GQM model the above mentioned goals may be expressed as follows:

Goal 1*: "To analyze the system requirements allocated to the software, and thus to establish a requirement-oriented baseline founded from the viewpoint of the software manager of the organization where the requirement management process is to be carried out."*

Goal 2: *"To analyze the software plans, products and activities with the aim of controlling their consistency with respect to the requirements founded from the viewpoint of the software manager of the organization where the requirement management process is to be carried out."*

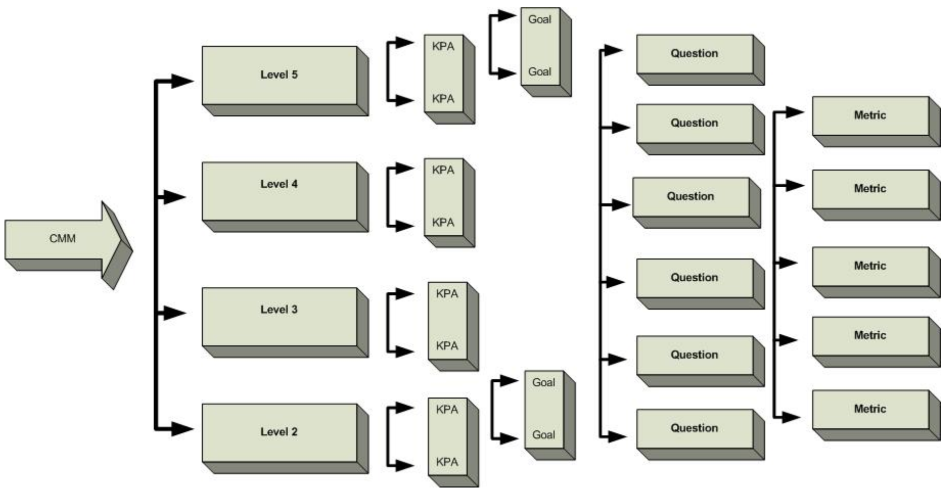

FIGURE 3. The Relationship between CMM & GQM.

4.1 Questions Regarding the First Goal of the Requirement Management Process

The Table 1 contains several questions regarding the first goal of the requirement management process. It must be pointed out that the answers to the questions are given on the basis of the software size and software project estimation. Obviously, the complexity of software processes is in direct relationship with the size and the complexity of the projects. The parameters of the project size should be taken into consideration in our measurements models.

QUESTIONS	FUZZY MEMBERSHIP DOMAIN	PARAMETERS
1-What is the level of requirement stability?	U: [0..1]	RS1= Number of final requirements-number of initial requirements / Number of final requirements
	U: [0..10]	RS2= number of requirement changes
	U: [4,6,8]	RS3= changes depth
2- What are the reasons behind requirement changes?	U: [0..1]	RCR1=RS1
	U: [0..10]	RCR2=RS2
	U: [0..10]	RCR3= Test cases per requirement
	U: [4,6,8]	RCR4=RS3
	U: [4,6,8]	RCR5= phase of the demanded changes
3- What is the cost of change of requirements?	U: [0..1]	CC= cost of change / cost of each requirement
4- Are the needed changes manageable?	U: [0..1]	CM1= proposed changes - rejected changes / number of proposed changes
	U: [0..1]	CM2= proposed changes - occurred changes / Proposed changes
	U: [0..1]	CM3= occurred changes - essential changes / changes occurred
	U: [0..1]	CM4= configuration items affected by changes / configuration items
	U: [0..1]	CM5= number of requirements affected by one change / number of requirements
5- Do the number of changes decrease with time?	U: [0..1]	CR= number of final changes-number of initial changes / number of initial changes
6- How many incomplete, inconsistent and missed requirements are identified?	U: [0..1]	RIIM1= incomplete requirements / number of requirements
	U: [0..1]	RIIM2= consistent requirements / number of requirements
	U: [0..0.5]	RIIM3= missed requirements / number of requirements

TABLE 1. The Questions and Parameters Regarding the First Goal of the Requirement Management Process

4.2 Questions Regarding the Second Goal of the Requirement Management Process

The second goal refers to the maintenance of consistency between the requirements and the software project. This is achieved through a tracking capability throughout the software records. By applying the requirement changes, other records will also be affected by these changes. Therefore, when any change takes place with regard to the requirements, plans, products and predefined activities, planned and recorded activities must be revised.

QUESTIONS	FUZZY MEMBERSHIP DOMAIN	PARAMETERS
1- Does the software product meet the requirements?	U: [0..1]	RS1= number of final requirements – initial requirements / number of final requirements
	U: [0..10]	RS2= number of changes of each requirement
	U: [4,6,8]	RS3= rate of changes
2- What are the effects of requirement changes on the software project?	U: [0..0.3]	PEF1= rate of effort made for requirement management / rate of effort made for the project
	U: [0..0.7]	PEF2= number of configuration items affected by one change / project configuration items
	U:[0..0.7]	PEF3= rate of effort made for the application of changes / rate of effort made for the project
3- Are all products consistent with the requirements?	U: [0..0.7]	RCO= number of inconsistencies / number of products

TABLE 2. Questions and Parameters Regarding the Second Goal of the Requirement Management Process

5. Distributed Artificial Intelligence

The fabrication of prototypes and artificial intelligence science were first propagated by Bon Gresor in 1989 and included two fields of activities [1]:

Solving Distributed Artificial Intelligence Problems:

This field consists of breaking down a problem along with its related sciences into subdivisions. Each subdivided problem is solved based on its relevant scientific knowledge and by combining the obtained solutions by which the key problem is solved.

The significant matter in this method is the way the key problem is broken down into subdivisions which are solvable in accordance with their relevant sciences and such that when the obtained solutions are combined, the key problem is not affected by new circumstances. The final solution should be applicable and efficient.

Multi- Agent Systems:

These systems consist of the performance of a series of automatic agents with the objective of arriving at a solution to a problem. The output of each agent is a suitable input to the next agent. Elements such as agent class, agent structure, knowledge, inference ability and compatibility, and learning are defined for each agent. However, each agent can be used or designed in either a static or dynamic form.

A group of agents, when put together, include elements such as group organization, coordination, cooperation, guidance, analogous behavior, harmonized planning and objectives and finally, internal relationships within the group. The model discussed in this article is the Fuzzy Multi-parametrical System which itself is a sample of the multi-agent system.

5.1 Designing Fuzzy Systems

Fuzzy systems are used to create prototypes of human knowledge regarding a particular issue. Human knowledge can fit into two categories: conscious and unconscious. Conscious knowledge can be clearly described in words. Unconscious knowledge, on the other hand, includes situations where experts know what they are doing but are unable to express it in words [6]. To express unconscious knowledge, an expert behaves differently in different situations. In this case, an expert is symbolically considered to be a black box, the inputs and outputs of which are then measured. Therefore, an expert’s behavior is converted into a series of inputs and outputs. The fundamental issue here is to build fuzzy systems based on input-output pairs. The common methods of creating fuzzy systems are:

- The Simple Heuristic Method
- Adjustment of parameters by teaching descendant gradients
- Adjustment of parameters by teaching the Minimum Recursive Squares

In designing fuzzy systems, choosing the number of appropriate rules is very important. Choosing excessive number of rules would complicate the system even further and such a complexity might be unnecessary in solving the problem. On the other hand, choosing very few rules would lead to the creation of a weak fuzzy system.

5.2 Multi-Agent Fuzzy Logic

The main objective is to develop methods and techniques for distributed calculations and process controls in multi-input systems through the fuzzy logic theory models. Consequently, distributed fuzzy rules could be proper tools for control systems with different modes (inputs) as well as for imprecise models (fuzzy).

The fuzzy system discussed in this article aims to control and measure both fuzzifier and defuzzifier processes. It consists of four steps:

- Fuzzifier
- Inference Engine
- Rule Base
- Defuzzifier

The Inference Engine includes a group of Fuzzy relations within the inference engine which are extendable from two dimensional environments to multi-dimensional ones. This is done by defining each of the fuzzy inputs and its respective effect on other fuzzy sectors [1].

6. Multi-Agent Fuzzy Logic for the Measurement of Software Processes

The outcome of the functions of the GQM method on the goals of key process areas of the SW-CMM model is a series of questions and parameters for answering them. As shown in Tables (1) and, (2) a fuzzy membership domain can be defined for each parameter. Therefore each parameter forms an input into the first agent of the fuzzy logic being discussed (as shown in Figure 4). The first agent output (the first level in Figure 4) provides the final quantity of answers to questions. This quantity forms an input into the second agent of the relevant fuzzy logic. The resulting output (the second level in Figure 4) determines to what extent a goal is achieved in key process areas and the third agent output determines the growth of key process areas based on the goals. Figure 4 shows the Multi-agent fuzzy logic used for the first goal of the requirement management process. The first agent in this system consists of 18 inputs and 6 outputs and the second agent consists of 6 inputs and one output which indicates how far the first goal of the requirement management process has been achieved. The aforementioned system includes a single fuzzifier, product inference engine and a centroid defuzzifier.

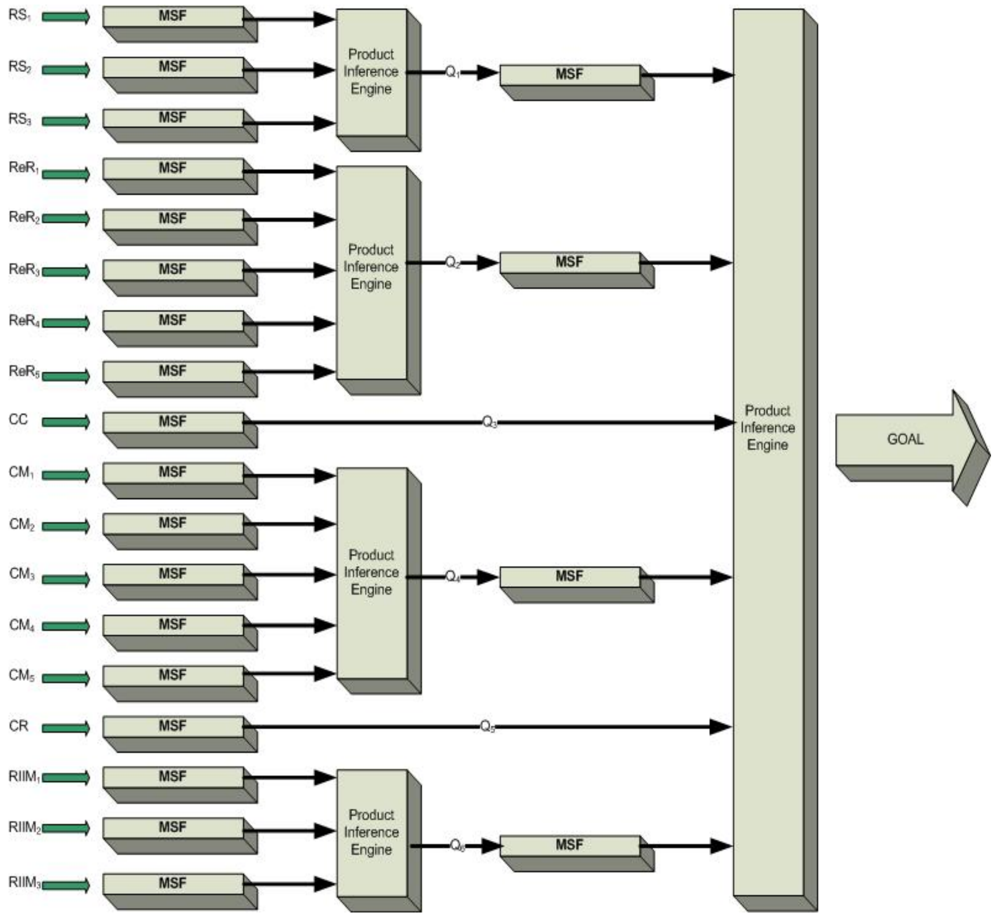

FIGURE 4. Multi-parametrical Fuzzy System for the Achievement of the First Goal of the Requirement Management Process

7. Measurement and Analysis Tools

The multi-agent fuzzy logic described in the previous section has been applied as a measuring and analyzing tool for software processes. Amongst the most significant characteristics of this tool are its easy and comprehensive usage; the automation of fuzzy calculations; the possibility of storing the results of each measurement and analysis and of further comparing and analyzing the result of various measurements for studying the maturity and improvement of software processes of the organization. Therefore this tool provides the possibility of analyzing and predicting the effect and outcome of changes in processes carried out in software organizations. This tool can facilitate the provision of a database regarding the maturity and improvement of software processes of contractors.

8-A Case Study:

The software processes measurement model presented in this article emphasizes the measurement of the achievement of direct goals based on a formal model. As mentioned in the previous sections of the said article, none of the existing methods have directly measured the performance of software processes but have mainly analyzed the software processes through parameters such as cost, time and man power utilized to produce the final product. Therefore the results discussed in this article are not comparable with other existing methods. This section follows the results obtained by carrying out the discussed method of measurement in the field of requirement management in a software development project for mobile customer care of TeleCommunication of Iran (TCI). This project was carried out at the Industrial Faculty of the Khajeh Nasir-e-din Toosi University (Table3).

In this project the requirements management process has been performed by project manager. Project engineering team considerations about requirement management process of this project is as follows:

- Requirements identification and control to establish a software engineering baseline is performed accurately. Defect rate and customer satisfaction reports also prove this issue.

- Requirements change requests have been applied directly in source code without any consistency control with other software products. Thus inconsistencies between software products increased. High maintenance costs also give credence to this issue.

The issues concluded from the results of examining SPMF method on requirements management process, which is shown in Table 3, are as follows:

- first goal achievement rate is acceptable. So, it can be said that identification and control of requirements in this project has been performed successfully

- low rate of second goal achievement indicates that project team has not performed well in consistency control of requirement changes and software products.

Therefore, the issues mentioned above correspond to the considerations of project team about requirement management process. Examinations on other sample projects as well as this case study approve SPMF measurement results.

Goal	Question	Metric	Response to Question	Achievement of Goal
First	Level of stability of requirements?	RS1=0.32 RS2=7 RS3=5	0.48	0.88
	What are the changes in requirement?	RCR1=0.32 RCR2=7 RCR3=9 RCR4=5 RCR5=7	0.14	
	What is the cost of changes in requirement? (CC)	CC=0.47	0.47	
	Can these changes be managed? (CM)	CM1=0.71 CM2=0.25 CM3=0.9 CM4=0.4 CM5=0.05	0.5	
	Is the number of changes reducing? (CR)	CR=0.92	0.92	
	What number of requirements has not been identified as incomplete & incompatible? (RIIM)	RIIM1=0.1 RIIM2-0.94 RIIM3=0.05	0.85	
Second	1- Does the software product meet the requirements?	RS1=0.32 RS2=7 RS3=5	0.48	0.36
	2- What are the effects of requirement changes on the software project?	PEF1=0.1 PEF2=0.4 PEF3=0.47	0.3	
	3- Are all products consistent with the requirements?	RCO=0.05	0.05	

TABLE 3. Results of Measurement of the First Goal of the Requirement Management Process of the TCI Mobile Customer Care Project

8. **Conclusion**

The fuzzy system of measuring and analyzing software processes presented in this article is the result of combining the SW-CMM model, the GQM method and the principles for designing multi-agent fuzzy systems. Among the most important characteristics of this system, its direct emphasis on the goals of software processes; application of direct measuring parameters; its general use in other SW-CMM key process areas; easy use; automation of calculations and the creation of a formal environment based on fuzzy logic in an informal environment, can be mentioned.

If the fuzzy controls used in this model are capable of learning and expanding their own rule-base, then there would be no time limitation with regard to change in the membership functions and the existing knowledge in the rule-base. As time changes, the rule-based knowledge would naturally develop. In other words, applying the nervous system methodology in the fuzzy systems mentioned in this article helps to adjust and improve the rules and membership functions, leading us to a more precise model.

REFERENCES

[1] Alexander, *Distributed fuzzy Control of ultivariable systems*, Klawer academic Publishers, 1996.

[2] V. R. Basili, C. Caldiera, and D. Rombach, *Experience Factory*, Encyclopedia of Software Engineering Volume 1, ,Jogn Wiley & Sons, (1994) 469-476.

[3] V. R. Basili and H. D. Rombach, *The TAME Project: Towards improvement – oriented software environments*, in IEEE Transactions on Software Engineering, **14** (**6**), (1988) 758-773.

[4] J. H. Baumert and M. S. McWhinney, *Software easures and the capability Maturity Model*. Software Engineering Institute Technical Report, CMU/SEI-92-TR-25, ESC-TR-92-0, 1992.

[5] C. Bergstrom, 2000, *process Metrics for Ericsson Erisoft AB-a proposal*, Umea University Report Umnad 292/2000, Umea, Jan.

[6] Gregory E. Kersten, *Stan Szpakowiz*, Negotration in distributed artificial intelligence, IEEE 1994 .

[7] P. Kuvaja and A. Bicego, *BOOTSTRAP-Europe's Assessment Method*, IEEE Software, May (1993) 83-85.

[8] M. C. Paulk, C. V. Weber, S. Garcia, M. B. Chrissis and M. Bush, *key Practices of the Capability Maturity Model Version 1.1*, Software Engineering Institute Technical Report, CMU/SEI-93-TR-25 ESC-TR-93-178, Pittsburgh, PA, 1993.

[9] J. Raynus, *Software Process Improvement With CMM*, Artech House Publishers, 1999, Boston.

[10] R. Van Solingen and E. Berghout, *the Goal/Question/Metric Method- A Practical Guide for Quality Improvement of Software development*", McGRAW-Hill Companiew, London 1999 .

MIR ALI SEYYEDI*, COMPUTER - SOFTWARE DEPARTMENT OF SCIENCES & RESEARCH, TEHRAN, IRAN

E-mail address : **seyyedi@behpardaz.net**

MOHAMMA TESHNEHLAB, DEPARTMENT OF CONTROL, KHAJEH NASIR TECHNICAL UNIVERSITY, TEHRAN, IRAN

E-mail address : **Teshnehlab@eet.kntu.ac.ir**

FEREIDOON SHAMS, COMPUTER - SOFTWARE DEPARTMENT OF SCIENCES & RESEARCH, TEHRAN, IRAN

E-mail address : **F.shams@agri-jahad.org**

* CORRESPONDING AUTHOR