International Journal of Industrial Engineering & Production Research

September2015, Volume 26, Number 3
pp. 193-211

http://IJIEPR.iust.ac.ir/

pISSN: 2008-4889

A Three- Stage Algorithm for Software Cost and Time Estimation in Fuzzy Environment

FatemeZareBaghiabad , Hassan KhademiZare*

Ms.c student yazd university - - yazduni - fatemezarebaghi@yahoo.com
Associated professor yazd university - - yazduni - hkhademiz@yazduni.ac.ir

KEYWORDS

software,
cost and time estimation,
fuzzy logic,
network models,
analytical hierarchy process,

ABSTRACT

The software productions are very expensive projects with uncertainty. For these reasons, the estimation of time and cost of software isvery important for both producers and consumers. In this paper an efficient three- stage algorithm is developed for software production cost and time estimation. In the first stage, the required person- month for implementation of software production are obtained by COCOMO and function point methods. We integrated these two methods to consider all aspects in software production and increase estimation accuracy. In the second stage the required duration for completion of each step of production (planning, analysis, design and so on) is obtained by paired comparisons matrix. In third stage, tables of complete time and cost of software are concluded by GERT network in project control and work break structure (WBS). In whole of all stages of this paper, triangular fuzzy numbers are used to express uncertainty existed in succession and repetition of each production step, time of beginning, ending, the duration of each task and costs of them. Retrieved results examined by 30 practical projects and conclude accuracy of 93 percent for time estimation and 92 percent for cost one. Also suggested algorithm is more accurate than COCOMOII 2000 algorithm as 50 percent based on examined problems.

1. Introduction

Generally, there are five methods for project cost and time estimation. Model-Based method is based on mathematic models. Its principals are mainly derived from actual data sets of implemented projects. Expertise-Based method is based on experts' opinions who have gained especial skills over implementing different projects. Learning-Oriented method estimates project cost and time by simulating previous projects. Dynamic-Based method clearly recognizes characteristics of labor effort, skills, software project costs and their changes during project. Composite method uses the combination

* **Corresponding author:***Hassan khademizare*
Email:*hkhademiz@yazduni.ac.ir*

of the above methods for the cost and time estimation of project[1]. There are different methods for cost and time estimation of software development. The accuracy of the estimation is of significant importance for the organization. Primary estimation models are based on regression analysis or mathematical derivations and current models are based on Simulation, Neural Network, Genetic Algorithm and Soft Computation[2]. To name a few of the most important methods, Estimation by Analogy, Expert opining, Delphi, Work Break Structure, Top-Down and Bottom-Up, Cased-Based Reasoning, Neural Network, Fuzzy Logic Model, Function Point Analogy, COCOMO and COCOMO can be mentioned[3]. Estimation by Analogy is appropriate when similar implemented projects are available in the same field[4]. Expert Opining method is quite useful when there is similar historical information. For cost calculation, an expert system is designed and implemented using experts' knowledge in similar projects. Delphi method uses direct communications with experts for cost and time estimation. It's very convenient when organization does not have proper data sets for cost estimation. WBS method has various applications in general software projects by break of hierarchical of tasks, systems and subsystems for scheduling and budgeting[5]. Top-Down method begins estimating from system level by testing of product macro functions and interrelation between sub functions. The cost of system level activities such as integration, configuration management and documentation is estimated and added to accounts[6, 7]. Bottom-Up method begins from components level and ends to system. Efforts needed for development of any components is computed and then added to effort costs needed in total system[8]. Case-Based Reasoning method belongs to the category of Machine Learning methods of cost estimation techniques[3]. One of the important steps of this technique is feature selection which can be highly precise in computational complexities[9]. Neural Network can model complicate relationships between depended variables (effort) and independent variables (cost component) and also use training records to be generalized for new data and then achieve acceptable results[10]. In the same context, by using Probabilistic Neural Network can simultaneously reach estimations of development software parameter (size or effort) and probability which real value of parameter is less than estimation one[11]. Fuzzy Logic Model is also categorized under Machine-learning techniques. It may begin with Short-Scale programs and the codes collected can be used as inputs for a fuzzy model of software development effort estimation[3]. One of the other vastly used methods for software cost estimation is Function Point Analogy based on Source Lines of Code. Due to its full quantity control capability at the end of estimation, this method is the most intuitive one for cost estimation. This method considers the whole software as a set of functions whose quantity determines the approximate program size[12].

COCOMO (Constructive Cost Model) approach is designed for cost calculation of large software systems. COCOMO is a developed version of COCOMO. COCOMO is a hierarchy of

estimation models and is applicable to more modern kinds of more volatility software development models such as business software, object-oriented software and software which uses spiral or additional models[13]. COCOMO uses Function Points or Lines of Code to estimate a software system size. However, Lines of Code can't be estimated in the early software development stages[14]. In COCOMO and COCOMO, 12 and 17 system properties are examined, respectively[15]. This algorithm analyzes characteristics of the four effective factors in software system including product, computer, human resource and project. Product characteristics comprises of ordered software reliability, database size, software project complexity. Computer characteristics include run time standards, main memory standards, boot time, dummy work capability. Personnel characteristics include analytical capability, work plan experience, programming capability, dummy business and programming language experience. Project characteristics contains documented programming operations, software tools usage and constrains on the implementation schedule[16].

There are 22 factors in COCOMO including Development Flexibility, Team Cohesion, Develop of Reuse capability, Unprecedented, Architecture and risk Resolution, Platform Experience, Data Base size, Required Development Schedule, Language and tools Experience, Process Maturity, Storage Constraint, Use of Software tools, Platform Volatility, Application Experience, Multi-site Development, Documentation Match to Life Cycle Needs, Required Software Reliability, Personnel Capabilities, Time Constraint, Programmer Capability, Analysis Capability and Product Complexity[17].

Software cost and time estimation models can also be categorized into algorithmic and non-algorithmic approaches. Non-algorithmic approaches often estimate cost and time by using either previous project information or people comments and experience instead of complex mathematical formula and equations, but algorithmic approaches have more complex mathematical computations and are based on mathematical models. These models attempt to relate effort to one or two project characteristics. The main cost component of the model is usually software size (such as number of lines of code). General forms of these models are linear or nonlinear regression[3].

In this paper, we computed required person-month for implementation of production projects by developed COCOMOII method. In COCOMO model, network factors including geographical extends of system, system reliability in critical conditions, and number of simultaneous users, accessibility to system support and coordination level with existing systems have less important; in this model less system functions are also used. So we added network factors to COCOMOII method and named it developed COCOMOII. Also we considered number of function points as software size in COCOMOII model. We raised accuracy of estimation by integrating of these two models and reducing their deficits. We used analytical hierarchy process and expert opinions to determine the influence of each of 33 project factor (mentioned in developed

COCOMOIImodel) on each of 8 production steps (Determine required systems, Understand the system requirements, analysis, design, system implementation, test and delivery, Maintenance and development, installation) in second stage. To maintain flexibility of developed algorithm in various conditions, the coefficients of model were determined as fuzzy number by fuzzy LIKERT spectrum. Coefficients used in this paper are designed for specific place conditions.

1.1- Production scheduling techniques in uncertain conditions

Determining of tables of scheduling and programming of software production is one of the basic elements of project management of software production. Inappropriate and unreal estimations of tables of production program make wrong assessment in cost, required resources and scheduling; also leads to customer unreliability, applying costs in contracts and contradiction in projects progress reports. One of the reasons of these unreal estimations is using inappropriate estimation techniques. Software projects have specific properties which distinguish it from other projects and approach software projects to research ones. These properties encompass uncertainty in definition, sequence and time of activities, required resources, existence of loops, and lack of reproducibility in project activities and so on. Three properties of uncertainty in time of activity, sequence and reproducibility are more important than other properties because of their direct effect on project required time and resources. We also apply fuzzy GERT[2] technique to determine the influence of project sequence and reproducibility on project time and cost.

In certain network techniques, network parameters such as activity definition, sequence and time are definitive. In stochastic network only parameter of activity time is uncertain estimated by probabilistic distribution. This technique obtains more real results than certain ones but can't consider cycles and uncertainty in activity definition and sequence. The cyclic stochastic network with logical nodes, stochastic paths and repetitive loops are appropriate tools for modeling of software production. This technique can consider cycles and uncertainty in activity times. Chanas and Kamburowski[18] and Henry Prade[19] were the first to use fuzzy logic to schedule project development. Many techniques have been offered for fuzzy scheduling so far which can be categorized into three categories based on their application: fuzzy time, fuzzy network and cyclic fuzzy network[20]. In project schedule technique with fuzzy time, only the network time parameter is fuzzy. Cyclic Fuzzy network are the same cyclic stochastic network which fuzzy parameters replace with stochastic ones. Itakura and Nishakava[21] in 1984 first suggested cyclic fuzzy networks. Ozdamar and Alanya[22] in 2000 examined software production projects and suggested modeling a problem of mathematical nonlinear 0-1 programming. Their model includes uncertainty in activity duration and network architecture. Activities can be done in parallel. Objective function is to minimize total project duration.

In this paper parameters of activity duration and number of loop iteration were shown as fuzzy

[2] Graphical Evaluation and Review Technique

triangular numbers in GERT networks. Output of this algorithm was shown as a table of production programming for all steps of software production that was depicted with fuzzy numbers. To test and examine validation of this algorithm, thirty real software projects were considered and estimated results were compared with real information.

1.2- Functions of Software Systems

Function points approach estimates cost by determining of software size. Software systems retain five functions: Internal Logic File (ILF), External Interface File (EIF), External Input (EI), External Output (EO) and External queries (EQ). Each function has three complexity degrees of Low, Average and High. Number and weight of each function is calculated according to standard Tab 1[23]. We adjusted this standard table proportional to expert opinion as fuzzy numbers. Numbers inside parenthesis shows weights of number of functions. Each function has three degree of complexity: low, average and high shown with fuzzy triangular numbers. For example, software experts conclude if the numbers of internal logic files are between 10 and 50, *N* will be 7 with degree of complexity of 0.9.

Tab 1: number of functions

Group	Low	Average	High
ILF	(10, 30, 50) $\Rightarrow$ N= 7 (W=0.9)	(30, 100, 200) $\Rightarrow$ N=10(W=1.05)	(100, 300, 500) $\Rightarrow$ N=15(W=1.3)
EIF	(10, 30, 50) $\Rightarrow$ N=5 (W=0.8)	(30, 100, 200) $\Rightarrow$ N=7 (W=0.9)	(100, 300, 500) $\Rightarrow$ N=10(W=1.05)
EI	(5, 25, 40) $\Rightarrow$ N=3 (W=0.7)	(25, 75, 125) $\Rightarrow$ N=4 (W=0.75)	(100, 200, 300) $\Rightarrow$ N= 6(W=0.85)
EO	(5, 25, 40) $\Rightarrow$ N=4 (W=0.75)	(25, 75, 125) $\Rightarrow$ N=5 (W=0.8)	(100, 200, 300) $\Rightarrow$ N=7(W=0.9)
EQ	(10, 30, 50) $\Rightarrow$ N=3 (W=0.7)	(30, 80, 130) $\Rightarrow$ N=4 (W=0.75)	(100, 200, 300) $\Rightarrow$ N=6(W=0.85)

As a result, to estimate software size, in addition to the number of functions, their complexities must also be estimated. Software size is named as unadjusted function point (UFP) and computed according to the Eq.1[5]:

$$\mathrm{UFP} = \frac{\sum_{i=1}^{5}\sum_{j=1}^{3} N_{ij} W_{ij}}{\sum_{i=1}^{5}\sum_{j=1}^{3} W_{ij}} \qquad (1)$$

N_{ij} = Number of functions type i with complexity level j

W_{ij} = Weight of number of functions type i with complexity level j

UFP = Number of unadjusted functions

We use UFP as size of software to estimate required person- month for implementation of production projects.

2. Effective factors for software cost and time estimation

In primary computers, software costs constituted a minor percentage of total computer system costs,

thus some errors in software cost estimation had relatively low effect on total system cost, but nowadays software is the most expensive component in computers so that error of estimation can be equal to the gap between benefit and loss.

Cost and time estimation is usually fuzzy and an imprecise science. Countless variables such as personnel, engineering issues, environmental conditions and government policies can affect final software cost and the work done for development. The main component of software development cost is due to man and his efforts. Most cost estimation methods focus on this aspect and estimate costs as person-month phrase. In order to achieve a reliable cost and work estimation of a software project, either a set of systematic steps must be designed or the software must be broken down into different functional components so that to have a sound estimation with an acceptable level of risk.

There is not a simple relationship between project costs and the final price that the costumer is charged. This is because it depends on different factors including organization validity, travels, required trainings, software engineers' salary, project characteristics, personnel, network and hardware which last during project life.

The model's practitioners must properly recognize product characteristics first, and then minimize error probability by using numbers and cost coefficients. In this method, project cost estimation is often implemented parallel to the project planning step so that to achieve conceivable project characteristics and incorporate sound budget numbers into the model. Developed model has an exponential factor for the most probable, optimistic and pessimistic conditions. Effective factors in determining power of weights of information systems (WIS) include available experience, production flexibility, project management capability, teamwork integrity and project control. Each factor has five states which are effective in determining WIS power: very high, high, medium, low and very low.

2.1-Factors and characteristics of information technology systems

There are generally 33 cost and time factors which determine software production time and cost. These factors are categorized into five categories: characteristics of product, computer, personnel, project and network. The coefficient mentioned in this table is determined based on condition of production system and characteristics of five above factors. Required information and coefficients are summarized is in Tab2. The coefficients of Tab2 are obtained based on field studies, experts' sessions and practical experiences available in three major centers of design, production and support. Each question in this table has five states: very high, high, medium, low and very low. Due to the diversity of questions, in some cases very much and in some cases very less is appropriate.

Selecting any of the state influence software time and cost estimation. For an even more accurate estimation, at least 5 experts are required to fill this table. The mean score of the expert's scores will be the basis of time and cost computation. To perception details of Tab2, see appendix A.

Tab 2: Summery of information and coefficients

Row	Title	Cost Factors	Very high	High	Average	Low	Very low
S1	Product Characteristics (S)	Database Size	1.86	1.28	1.08	0.94	0.5
S2		Complexity of program structure	1.3	1.15	1	0.85	0.7
S3		Flexibility & predictability capability	1.5	1.3	1.1	1	0.8
S4		Importance of user information security	1.4	1.15	1	0.88	0.75
S5		Importance of compatibility with relevant applications (system Integration)	1.3	1.2	1.1	1	0.8
S6		Observation of software standards	1.1	1.05	1	0.7	0.3
S7		Usage of modern software tools(required)	1.5	1.3	1.3	0.8	0.5
S8		Number of inputs and outputs	1.4	1.3	1.3	0.9	0.7
C1	Computer Characteristics (C)	Required memory to run	1.21	1.06	1	0.5	0.25
C2		Compatibility with existing hardware	1.3	1.15	1	0.87	0.3
C3		Computer response time(run time)	1.15	1.07	1	0.87	0.35
C4		Usage of modern hardware tools	1.3	1.11	1	0.5	0.25
C5		Diversity of existing hardware	1.1	1	0.9	0.7	0.3
H1	Personnel Characteristics (H)	Analyst group capability (infer & provide requirements)	0.71	0.86	1	1.19	1.46
H2		Designer group experiences about considered system	0.82	0.91	1	1.13	1.29
H3		Programmer group capability	0.7	0.86	1	1.19	1.46
H4		Group knowledge about present program(hardware& software)	0.5	0.9	1	1.1	1.2
H5		Group capability in system integration	0.5	0.95	1	1.07	1.14
P1	Project Characteristics (P)	Usage by professional programmers	0.7	0.86	1	1.17	1.42
P2		Project scheduling changes	1.1	1.04	1	1.08	1.23
P3		Project delivery time	0.82	0.91	1	1.12	1.3
P4		System validity and verification	1.3	1.11	1	1.11	1.3
P5		Software response level	0.7	1	1.1	1.2	1.5
P6		Ease of Install & launch	1.3	1.1	1	0.8	0.5
P7		Software documentations	1.3	1.1	1	1.1	0.5
P8		Support Period (maintenance, modification, user training)	1.5	1.3	1.2	0.8	0.5
P9		Number of main software functions	1.4	1.3	1.2	1	0.4
P10		system development capability	1.3	1.11	1.05	0.5	0.25
N1	Network Characteristics (N)	Geographical Extends of system	1.5	1.3	1.2	0.7	0.5
N2		Number of simultaneous users	1.4	1.15	1	0.58	0.3
N3		Access to system support	1.4	1.2	1.1	0.7	0.5
N4		Reliability in Critical situations	1.3	1.15	1	0.87	0.3
N5		Coordination with present systems	1.4	1.3	1.1	0.8	0.25

3. First Stage of Hybrid Algorithm: Time and Cost Estimation Algorithm

After studying and examining benefits, shortcomings and applications of each method of software cost estimation, our algorithm steps which are a combination of Functional and COCOMOIItechniques are as follows:

Step 1: Calculate software size by function points approach (UFP).

In this step, first Tab1 is completed by n experts. Weight average of function points (UFP) is calculated as Eq.1. If Eq.1 is completed by more one expert UFP will compute by average of all UFPs. Our purpose of n experts is that people who are aware about software functions and work practical software production projects.

Step 2: Determine each coefficient of Tab2. (n experts must complete this table).

Step 3: Multiply the resulted weights in all rows ofEach coefficient of Tab2 selected by an expert

is called effort multiplier (EM). The obtained number is weights of information systems (WIS) and obtained as Eq.2 [24]. If Tab2 is completed by more one expert, average of WIS for all experts must be computed.

$$WIS = \prod_{i=1}^{33} EM \qquad (2)$$

Step 4: Calculate the minimum, maximum and most likely project duration by Eq.3, Eq.4 and Eq.5. Resulted numbers express required person-month for project implementation. These equations are the same standard relations in COCOMOII [25].

$$Person-month_{\min} = 2.8(WIS)(UFP)^{1.05} \qquad (3)$$

$$Person-month_{nom} = 3(WIS)(UFP)^{1.12} \qquad (4)$$

$$Person-month_{\max} = 3.2(WIS)(UFP)^{1.2} \qquad (5)$$

Step 5: Distribute each of the calculated times between project personnel according to Tab3.

Tab 3: Staff cost and time percentage

Personnel Characteristics	Firm management	Project management	MA	Undergraduate	Technician
Time percentage	4.9, 5, 5.1	14.8, 15, 15.2	24.7, 25, 25.3	19.8, 20, 20.2	34.5, 35, 35.5

4. Second stage of hybrid algorithm: Paired comparisons matrix of analytical hierarchy process (AHP) for determining relationship between factors and steps

The world around us is full of multi- criteria problems which we must make decision. One of the most complete approaches for making decision is analytical hierarchy process which first suggested by Saaty in 1980. This process is a graphic view from real complex problem. General objective of problem is in head of this process and criteria, sub criteria and alternatives are in next levels. Qualitative and quantitative objectives are measured as paired comparisons for comparing alternatives to criteria and criteria to objectives[26]. We used the fuzzy spectrum with five options in these paired comparisons as Tab 4 and Fig. **1**. In this paper, we assumed factors (product, project, computer, personnel and network) are independent from each other.

Tab 4: The relationship for paired comparisons

Alternative	Alternative weight	Symbol	Linguistic phrase
Excellent	(0.75, 1, 1)	E	Very high
Good	(0.5, 0.75, 1)	G	High
Fair	(0.25, 0.5, 0.75)	F	Medium
Poor	(0, 0.25, 0.5)	P	Low
Very Poor	(0, 0, 0.25)	V	Very low

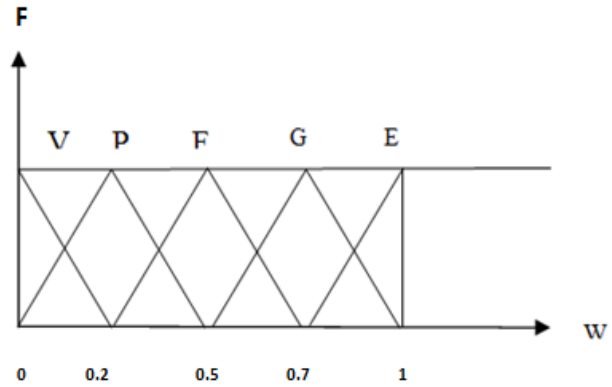

Fig.1: Review of fuzzy numbers

Early weights of alternatives to criteria (steps) and criteria to general objective were calculated by dividing retrieving weights on each factor of questionnaire ($\frac{w_i}{w_j}$) and comparing every element of each level to other levels. Final weight of each criterion and alternatives to each other was calculated by applying analytical hierarchy process and Expert Choice (EC) software. Chart of analytical hierarchy process was shown as Figure 2 and calculated weights depicted as Tab 5.

To increase accuracy and validity of this algorithm, we used opinions of 150 software experts. To collect information we design a questionnaire with three sections. In first section we determine importance of each alternative to factors. In second section we determine importance of factors to steps and in third section determine the influence of each step on software cost and time estimation. Related experts were from three major centers of design, production and support. These experts were capable and possessor of at least three years work experience; thus in this paper statistical sample was equal to statistical society.

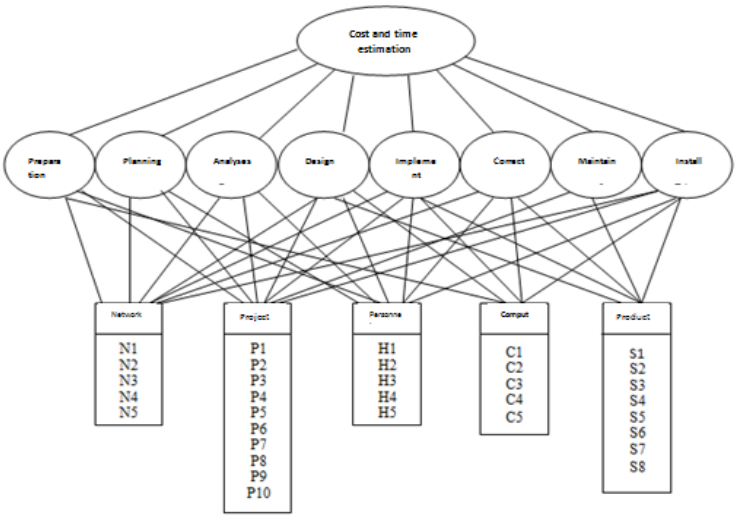

Fig.2: Analytical hierarchy process

Tab 5: Time percentages of steps

The influence of each factor on each production step								
Factors: Product Computer Personnel Project Network	Planning		Analysis	Design	Implement		Support	
	Determine required systems	Understand the system requirements			System implement	Test and Deliver	Maintenance and development	Installation
Time percentage	2.9, 3, 3.1	4.9, 5, 5.1	19.9, 20, 20.1	16.8, 17, 17.2	24.8, 25, 25.2	23.8, 24, 24.2	10.8, 11, 11.2	4.9, 5, 5.1

Total duration of project was calculated by following steps 1 to 3 in explained algorithm. Then the duration of each production step was obtained by multiplying total duration to time percentage of each step. The durations of each step were inputs of third stage of the algorithm. In third stage total time and cost of project was obtained by calculations of fuzzy GERT networks and development algorithm.

4.1- System development steps

Cycle of Software system development has different methodologies in macro levels. The most important and complete methodology is information system development life cycle (ISDLC), since it has numerous applications and encompasses all steps of planning, analysis, design, implementation and software support. Different models are designed for implementing information system development life cycle steps. The most important methodologies used in software production include: Linear Sequential model, Waterfall model, Prototyping model, Rapid Application Development model, Incremental model, Spiral model, Parallel Development, Object Oriental Component model, Formal Methods model and Fourth Generation techniques[27]. In this paper in order to analyze and evaluate the developed algorithm, Waterfall method is used.

5. Third stage of hybrid algorithm: Development of solving method for GERT networks in uncertain condition

After determining the duration of each step of software production, we must define nodes, paths and production loops for reaching tables of production programming. To model software projects explained in this paper, we use cyclic fuzzy graphical evaluation and review technique (GERT) networks which parameters are shown as fuzzy sets[28]. First projects information were estimated based on definitions and assumptions, then network of software steps were depicted and in next step cyclic fuzzy network was solved. These steps were explained in[28]. Output of this algorithm includes tables of production programming, implementation steps and completion time of project as fuzzy numbers.

5.1- Analysis of results of production programming

Table of production programming including

implementation time of each step of software production obtains by evaluating each node and activity. Since calculations are based on fuzzy numbers, in cyclic fuzzy networks the completion time of project is a fuzzy number instead of a definitive one. This uncertain numbers adapt to reality and decreases stresses in project control sessions. Also the cost is estimated more real. Computational model developed in this paper is designed in Excel environment and simply extended for other software. This model is capable of upgrading in new conditions. Fuzzy GERT chart used in this paper depicted steps of design, production and software support with loops is shown in Fig. 3.

5.2- Validation and verification of developed algorithm

To examine validity and verification, 30 real projects were considered implemented in three major centers of design, production and support. Time and cost of these projects were estimated by developed algorithm and method of COCOMOII2000 and then compared with real time and cost. Steps of cost and time estimation for software were explained as below:

Step1. Determine value of UFP by **Tab1** and expertise comments

Step2. Complete Tab2 by expertise comments and compute WIS

Step3. Compute minimum, maximum and most likely values of required person- month for project implementation by Eqs.1, 2 and 3:

Person-Month= (person-month $_{Min}$, person-month $_{Nom}$, person-month $_{Max}$)

Step4. Compute time percent of each production step by last row of Tab 5: For this purpose, values of person- month were multiplied by values of last row of Tab 5 and results have been in second column of Tab 6. In Tab 6, number and probability of loop were determined by experts.

Step5. Do network computations and complete Tab.7 and Tab.8: These computations were done based on computations of paper [28]. The completion time of each production step and completion time of project was determined by Tab.7 and Tab.8. The completion time of project was computed by Eq.6:

$$\textbf{The completion time} = \frac{(F8_{min} + (F8_{nom} * 4) + F8_{max})}{6} \quad (6)$$

Step6. Compute project cost by Tab. 9: In this table, participation percent and salary of each work group affected on software production was determined and by help of this, software cost was computed.

Step7. Compare estimated time and cost of software production with real ones: values of last row of Tab 10 expressed improvement on production time and cost of software in suggested algorithm relative to real values and COCOMOII 2000 method.

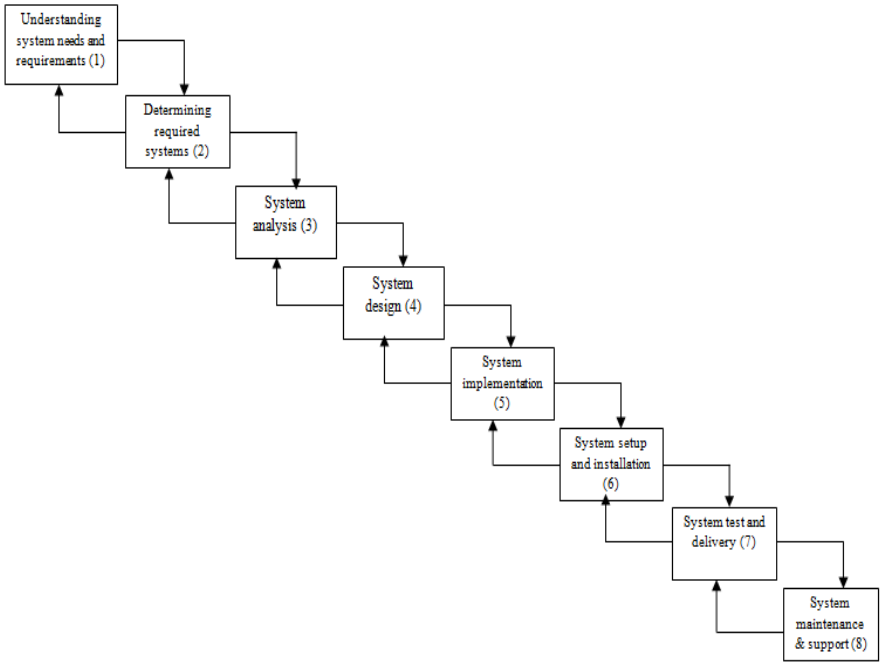

Fig.3: model of software production

Tab 6: Network loops and activities

Rank	Activity code (loop)	Activity description (loop)	Activity duration (month) Loop frequency (number)	Activity occurrence probability (loop)
1	1	Understanding system needs and requirements	(a_1, b_1, c_1)	1
2	2	Determining required systems	(a_2, b_2, c_2)	1
3	3	System analysis	(a_3, b_3, c_3)	1
4	4	System design	(a_4, b_4, c_4)	1
5	5	System implementation	(a_5, b_5, c_5)	1
6	6	System setup and installation	(a_6, b_6, c_6)	1
7	7	System test and delivery	(a_7, b_7, c_7)	1
8	8	System maintenance and support	(a_8, b_8, c_8)	1
9	2 – 1	Loop 2 – 1	(2, 3, 4)	0.14
10	3 – 2	Loop 3 – 2	(1, 2, 3)	0.11
11	4 – 3	Loop 4 – 3	(1, 2, 3)	0.12
12	5 – 4	Loop 5 – 4	(2, 3, 4)	0.11
13	6 – 5	Loop 6 – 5	(3, 4, 5)	0.15
14	7 – 6	Loop 7 – 6	(1, 2, 3)	0.14
15	8 – 7	Loop 8 – 7	(2, 3, 4)	0.12

Tab.7: Activity duration including iteration loops

Activity	Iteration loops	Loop occurrence probability	Activity duration	Frequency loop	Activity frequency	Final activity frequency	Final activity completion time
1	2 – 1	0.14	(a_1, b_1, c_1)	(2, 3, 4)	(0.3, 0.4, 0.6)	(1.3, 1.4, 1.6)	B1= $1.3a_1, 1.4\ b_1,)$ $(1.6c_1$
2	2 – 1	0.14	(a_2, b_2, c_2)	(2, 3, 4)	(0.4, 0.6,	(1.4, 1.6, 1.9)	B2=

	3 – 2	0.11		(1, 2, 3)	0.s9)		$1.4a_2$, $1.6\ b_2$,) ($1.9c_2$
3	3 – 2 4 – 3	0.11 0.12	(a_3, b_3, c_3)	(1, 2, 3) (1, 2, 3)	(0.2, 0.5, 0.7)	(1.2, 1.6, 1.7)	B3= $1.2a_3$, $1.6\ b_3$,) ($1.7c_3$
4	4 – 3 5 – 4	0.12 0.11	(a_4, b_4, c_4)	(1, 2, 3) (2, 3, 4)	(0.3, 0.6, 0.8)	(1.3, 1.6, 1.8)	B4= $1.3a_4$, $1.6\ b_4$,) ($1.8c_4$
5	5 – 4 6 – 5	0.11 0.15	(a_5, b_5, c_5)	(2, 3, 4) (3, 4, 5)	(0.7, 0.9, 1.2)	(1.7, 1.9, 2.2)	B5= $1.7a_5$, $1.9\ b_5$,) ($2.2c_5$
6	6 – 5 7 – 6	0.15 0.14	(a_6, b_6, c_6)	(3, 4, 5) (1, 2, 3)	(0.6, 0.9, 1.2)	(1.6, 1.9, 2.2)	B6= $1.6a_6$, $1.9\ b_6$,) ($2.2c_6$
7	7 – 6 8 – 7	0.14 0.12	(a_7, b_7, c_7)	(1, 2, 3) (3, 4, 5)	(0.5, 0.8, 1)	(1.5, 1.8, 2)	B7= $1.5a_7$, $1.8\ b_7$, 2) (c_7
8	8 – 7	0.12	(a_8, b_8, c_8)	(3, 4, 5)	(0.4, 0.5, 0.6)	(1.4, 1.5, 1.6)	B8= $1.4a_8$, $1.5\ b_8$,) ($1.6c_8$

Tab.8:Software production schedule

Node	Input activity	Final activity completion	Activities begin time	Activities finish time
1	Understanding system needs and requirements	B1	R1 = (0,0,0)	F1 = R1 +B1
2	Determining required systems	B2	R2 = B1 + R1	F2 = R2 + B2
3	System analysis	B3	R3 = B2 + R2	F3 = R3 + B3
4	System design	B4	R4 = B3 + R3	F4 = R4 + B4
5	System implementation	B5	R5 = B4 + R4	F5 = R5 + B5
6	System setup and installation	B6	R6 = B5 + R5	F6= R6 + B6
7	System test and delivery	B7	R7 = B6 + R6	F7 = R7 + B7
8	System maintenance and support	B8	R8 = B7 + R7	F8 = R8 + B8

Tab.9: cost of software production

Required time for project completion	**Percent participation in project time (**Error! Not a valid result for table.**)**				
B1 + …+ B8	**Company manager (4.9, 5, 5.1)**	**Project manager (14.8, 15, 15.2)**	**MA (24.7, 25, 25.3)**	**BA (19.8, 20, 20.2)**	**Associate degree (34.5, 35, 35.5)**
Sum up monthly time	(B1+ …+ B8)* (4.9, 5, 5.1)	(B1+ …+ B8)* (14.8, 15, 15.2)	(B1+ …+ B8)* (24.7, 25, 25.3)	(B1+ …+ B8)* (19.8, 20, 20.2)	(B1+ …+ B8)* (34.5, 35, 35.5)
Monthly salary (monetary unit)	A	B	C	D	E
The cost of working group	(B1+ …+ B8)* (4.9, 5, 5.1)* A	(B1+ …+ B8)* (14.8, 15, 15.2)* B	(B1+ …+ B8)* (24.7, 25, 25.3)* C	(B1+ …+ B8)* (19.8, 20, 20.2)* D	(B1+ …+ B8)* (34.5, 35, 35.5)* E
Sum up total project costs	(B1+ …+ B8)*[(4.9, 5, 5.1)* A+ (14.8, 15, 15.2)* B+ (24.7, 25, 25.3)* C+ (19.8, 20, 20.2)* D+(34.5, 35, 35.5)* E]				

5.3- Compare estimated results with real information

Time and cost for 30 projects were estimated by two methods of COCOMOII 2000 and algorithm developed in this paper. Estimated results were compared with real time and cost as Tab 10, Fig. 4 and Fig. **5**. For comparing time and cost

software, Eq. 7 was used.

$$MRE = \frac{|\ Estimated - Actual\ |}{Actual} \quad (7)$$

Tab 10: compare estimated results with real information

Project	Real cost and time		COCOMOII2000 method				Hybrid / developed method			
	Cost (monetary unit)	Time (Person-hour)	Cost (monetary unit)	Percent deviation	Time (Person-hour)	Percent deviation	Cost (monetary unit)	Percent deviation	Time (Person-hour)	Percent deviation
P1	125	1300	144	0.15	1450	0.12	139	0.11	1405	0.08
P2	175	1700	191	0.09	1870	0.1	190	0.05	1768	0.04
P3	280	2950	347	0.24	3540	0.2	311	0.11	3245	0.1
P4	135	1420	151	0.12	1620	0.14	146	0.08	1534	0.08
P5	920	9800	970	0.06	10585	0.08	966	0.05	10290	0.05
P6	825	8500	743	(0.9)	7820	(0.08)	756	(0.06)	8075	(0.05)
P7	740	7500	703	(0.05)	6975	(0.07)	718	(0.03)	7275	(0.03)
P8	150	1450	165	(0.1)	1625	0.12	178	0.08	1551	0.07
P9	200	1900	156	(0.23)	1520	(0.2)	180	(0.1)	1691	(0.11)
P10	120	1300	111	(0.08)	1015	(0.22)	114	(0.05)	1144	(0.12)
P11	560	5700	672	0.2	6670	0.17	627	0.12	6270	0.1
P12	565	5450	706	0.25	6705	0.23	635	0.12	4796	0.12
P13	420	4150	370	(0.12)	3735	(0.1)	400	(0.05)	3942	(0.06)
P14	310	3200	242	(0.22)	2435	(0.24)	273	(0.12)	2848	(0.11)
P15	650	6400	552	(0.15)	5440	(0.15)	579	(0.11)	5888	(0.08)
P16	340	3300	306	(0.1)	2937	(0.11)	326	(0.04)	3200	(0.03)
P17	595	5900	660	0.11	6726	0.14	625	0.06	6372	0.08
P18	780	7900	880	0.15	8850	0.12	855	0.1	8295	0.05
P19	675	6800	580	(0.14)	5985	(0.12)	601	(0.11)	6324	0.07
P20	480	5100	422	(0.12)	4590	(0.1)	446	(0.07)	4845	0.05
P21	350	3400	410	0.17	3875	0.14	392	0.12	3740	0.1
P22	950	9650	810	(0.15)	8396	(0.13)	855	(0.1)	8685	0.1
P23	160	1700	176	0.1	1887	0.11	173	0.08	1785	0.05
P24	890	8700	935	0.05	9405	0.09	917	0.03	8352	0.04
P25	850	8650	757	(0.11)	7612	(0.12)	807	(0.05)	7698	0.11
P26	195	2600	175	0.1	2540	0.02	188	0.04	2513	0.03
P27	215	2850	190	0.12	2710	0.05	204	0.05	2753	0.03
P28	200	2670	220	(0.1)	3020	(0.13)	207	(0.03)	2775	(0.04)
P29	225	2980	205	0.08	2900	0.03	213	0.05	2853	0.04
P30	180	2400	220	(0.18)	2970	(0.24)	199	(0.1)	2669	(0.11)
Sum	13260	137320	13167.5	4.74	137408	3.86	13219	2.27	134581	2.13
Average	442	4577.3	438.92	0.16	4580.3	0.13	440.63	0.08	4486	0.07

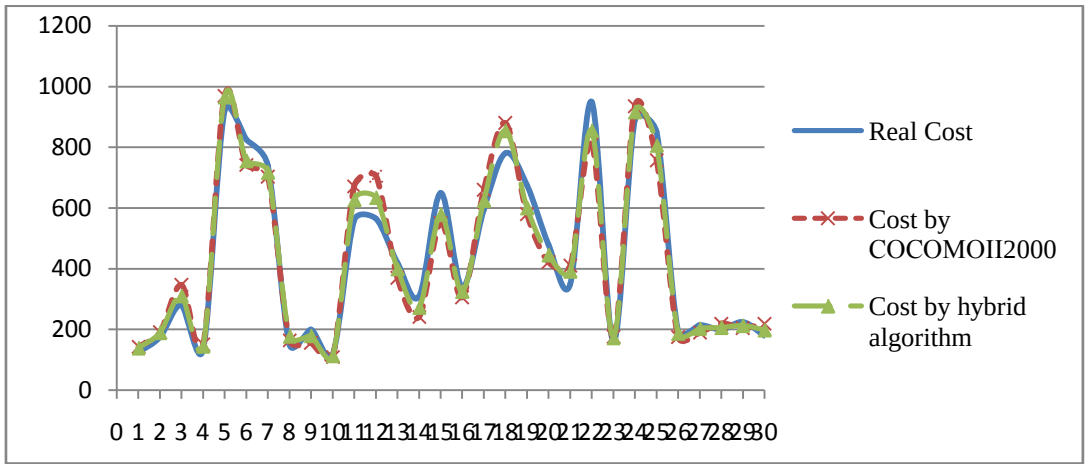

Fig.4: Comparing real cost with cost estimated by COCOMOII2000 and hybrid algorithm

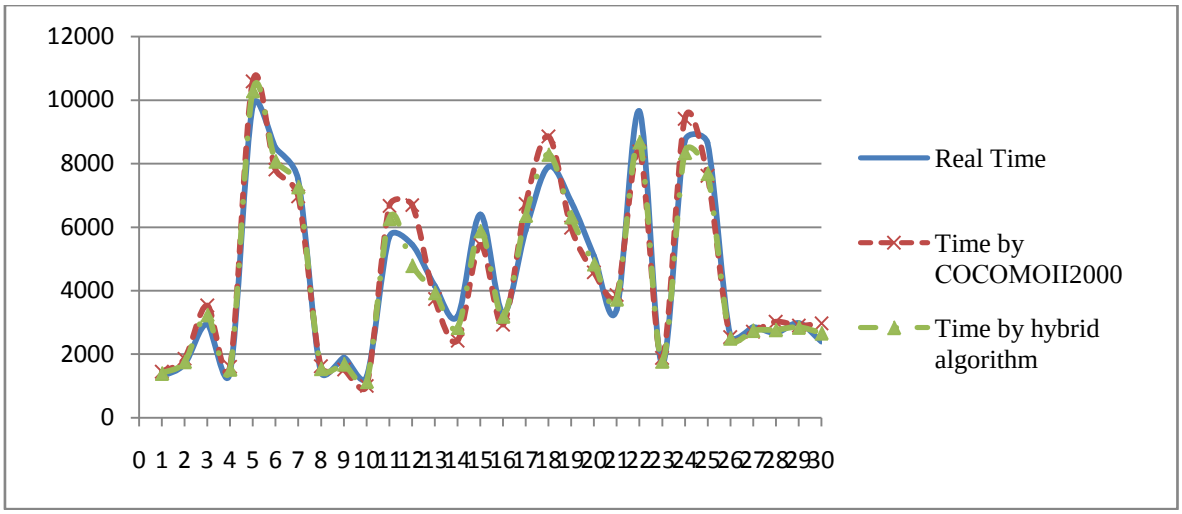

Fig.5: Comparing real time (person- hour) with time estimated by COCOMOII2000 and hybrid algorithm

6. Conclusions and Future Researches

In this paper a three- stage algorithm was developed to estimate cost and time of software production projects in uncertain conditions. First stage includes obtaining required person- month for implementation of production projects by combining of two models of function points and COCOMOII. In second stage time relationship between specifications of five factors (product, computer, personnel, project and network) and eight steps of software production were determined by paired comparisons matrix of AHP model.

In this stage, the duration of each step was determined for iterance to third stage. In third stage, the final time and cost was computed by work break structure and fuzzy GERT networks. Results of comparing real information of 30 implemented projects with estimations of suggested algorithm express accuracy of 93 percent for time estimation and 92 percent for cost one. The maximum relative error for when only COCOMOII 2000 method used, is twice relative to suggested algorithm. The accuracy of 93 and 92 for estimation of software time and cost with high uncertainty is very worthy.

As future researches, we suggested:

1. To determine specifications of software systems, we used methods of COCOMOII and Function points. Other methods can be suggested to distinguish important specifications which effect on time and cost estimation.
2. In this paper calculation of fuzzy GERT network was used only for the competition time and then we estimated cost based on the time. Other approaches can be suggested to estimate cost directly by network and fuzzy GERT calculations.
3. It’s useful to design a method which can estimate software sales and its effect on cost and profitability of software.

Resources

[1] Dolado, J.J., "*On the Problem of the Software Cost Function*", Inform Software Tech, 43 (1), 2001, pp.61- 72.

[2] Choudhary, K., "*GA Based Optimization of Software Development Effort Estimation"*, International Journal of Computer Science and Technology, 1 (1), 2010, pp.38-40.

[3] Lopez-Martin, C., "*A fuzzy logic model for predicting the development effort of short scale programs based upon* twoindependent *variables"*, Applied Soft Computing, 11(1), 2011, pp.724–732.

[4] Idri A., K.T.M. *Fuzzy Analogy: A New Approach for Software Cost Estimation*. in *International workshop on* software *measurment*. 2001. Aachen, Germany.

[5] Ghahramani, B., "*Software Reliability Analysis: a Systems Development Model"*, Computers and Industrial Engineering, 45 (2), 2003, pp.295-305.

[6] XU, Z., Khoshgoftoar, T.M., "*Identification of Fuzzy Models of Software Cost Estimation",* Fuzzy Sets and Systems, 145 (1), 2004, pp.141-163.

[7] Huang X., H.D., Ren J., Capretz L.F., "*Improving the COCOMO model using a neuro-fuzzy approach",* Applied Soft Computing, 7, 2007, pp.29-40.

[8] Izyumov B., K.E., Wagenkecht M., *Software Tools for Regression Analysis of Fuzzy Data*, in *Proceeding of 9th* Zittau *Fuzzy Colloquium*. 2001. p. 221-229.

[9] Li, Y.F., Xie, M., Goh, T.N., "*A study of mutual information based feature selection for case based reasoning in* software *cost estimation",* Expert Systems with Applications, 36 (3), 2009, pp.5921–5931.

[10] Bhatnagar, R., "*Software Development Effort Estimation – Neural Network Vs. Regression Modeling* Approach*",* International Journal of Engineering Science and Technology, 2 (7), 2010, pp.2950-2956.

[11] Pendharkar, P.C., "*Probabilistic estimation of software size and effort",* Expert Systems with Applications, 37 (6), 2010, pp.4435-4440.

[12] Musilek P., P.W., Succi G., Reformat M., "*Software Cost Estimation with Fuzzy Models",* Applied Computing Review, 8 (2), 2000, pp.24-29.

[13] Dickson, G., *Software Cost Estimation* Canada: Bericht, Faculty of Computer Science–Faculty of Engineering, University of New Brunswick. 2007.

[14] Ali A., Q.S., Shah Muhammad S., Abbas J., TariqPervaiz M., Awan S., "*Software cost estimation through* entity *relationship model",* J. Am. Sci., 6 (11), 2010, pp.47-51.

[15] Boehm B., A.C., Brown A. W., Chulani S., Clark B.K., Horowitz E., Steece B. , *Software Cost Estimation with* COCOMO *II*Prentice- Hall: 2000.

[16] Kazemifard, M., Zaeri, A., Ghasem-Aghaee, N., Nematbakhsh, M.A., Mardukhi, F., "*Fuzzy Emotional* COCOMO *II Software Cost Estimation (FECSCE) using Multi-Agent Systems"*, Applied Soft Computing, 11 (2260–2270), 2011.

[17] Sicila M.A., G.E., Galvo T., "*An Inqairy-based Method for Choquet Integral-based Aggregation of Interface* Usability *Parameters"*, Kybernetika, 39 (5), 2003, pp.401- 414.

[18] Chans., K.J., "*The use of Fuzzy Variables in pert"*, Fuzzy Sets and Systems, 5 (1), 1981, pp.11-19.

[19] Prade.H, "*Using Fuzzy set theory in a Scheduling Problem: A Case Study"*, Fuzzy sets and Systems, 2 (2), 1979, pp.153-165.

[20] Wang, J., "*A fuzzy robust scheduling approach for product development projects"*, European Journal of Operational Research, 152 (1), 2004, pp.180–194.

[21] Itakura, H., Nishikawa, Y., "*Fuzzy Network technique for technological Forcasting"*, Fuzzy Sets and Systems, 14 (2), 1984, pp.99-131.

[22] Ozdamar, L., Alanya, "*Uncertainty modeling in software development projects (with case study)"*, Annals of Operations Research, 102 (1-4), 2000, pp.157–178.

[23] Al-Hajri, M.A., Abdul Ghani, A.A., Sulaiman, M.N., Selamat, M.H., "*Modification of standard Function Point* complexity *weights system"*, The Journal of Systems and Software, 74(2), 2005, pp.195–206.

[24] Boehm, B., Valerdi, R., lane, J.A., Winsor Brown, A., "*COCOMO Suite Methodology and Evolution, Software* Engineering *Technology"*, Journal of Defense Software Engineering, 2005, pp.20-25.

[25] Jung, P., ed. *Software Cost Estimation and COCOMO II*. 1997, Systems Engineering Research Institute (SERI): Taejon, Korea.

[26] Ataii, M., *Fuzzy multi- critria making decision* Iran: Damghan university. 2011.

[27] Kargari M., K.H., *Information Systems Technology*Iran: Elmosanat university. 2005.

[28] Khademi Zare, H., Fatemi Ghomi, S.M.T., Karimi, B., "*Developing a heuristic algorithm for order production planning using network models under uncertainty conditions"*, Applied Mathematics and Computation, 182 (2), 2006, pp.1208–1218.

Appendix A

Computation of coefficients of Tab2:Each component of Tab.11 is ranking by n experts based on the importance of them for estimating time and cost. The concept of each ranking is

shown in Tab.12. For each component of Tab2, very high, high, average, low and very low coefficients are computed based on Eq. 8- 12. The resulted numbers are the same coefficients of Tab2.

Tab. 11: Computation of coefficients of Tab2

Row	Title	Cost Factors	Importance of each component								
S1	Product Characteristics (S)	Database Size	2	1.75	1.5	1.25	1	0.75	0.5	0.25	0
S2		Complexity of program structure	2	1.75	1.5	1.25	1	0.75	0.5	0.25	0
S3		Flexibility & predictability capability	2	1.75	1.5	1.25	1	0.75	0.5	0.25	0
S4		Importance of user information security	2	1.75	1.5	1.25	1	0.75	0.5	0.25	0
S5		Importance of compatibility with relevant applications (system Integration)	2	1.75	1.5	1.25	1	0.75	0.5	0.25	0
S6		Observation of software standards	2	1.75	1.5	1.25	1	0.75	0.5	0.25	0
S7		Usage of modern software tools(required)	2	1.75	1.5	1.25	1	0.75	0.5	0.25	0
S8		Number of inputs and outputs	2	1.75	1.5	1.25	1	0.75	0.5	0.25	0
C1	Computer Characteristics (C)	Required memory to run	2	1.75	1.5	1.25	1	0.75	0.5	0.25	0
C2		Compatibility with existing hardware	2	1.75	1.5	1.25	1	0.75	0.5	0.25	0
C3		Computer response time(run time)	2	1.75	1.5	1.25	1	0.75	0.5	0.25	0
C4		Usage of modern hardware tools	2	1.75	1.5	1.25	1	0.75	0.5	0.25	0
C5		Diversity of existing hardware	2	1.75	1.5	1.25	1	0.75	0.5	0.25	0
H1	Personnel Characteristics (H)	Analyst group capability (infer &provide requirements)	2	1.75	1.5	1.25	1	0.75	0.5	0.25	0
H2		Designer group experiences about considered system	2	1.75	1.5	1.25	1	0.75	0.5	0.25	0
H3		Programmer group capability	2	1.75	1.5	1.25	1	0.75	0.5	0.25	0
H4		Group knowledge about present program(hardware& software)	2	1.75	1.5	1.25	1	0.75	0.5	0.25	0
H5		Group capability in system integration	2	1.75	1.5	1.25	1	0.75	0.5	0.25	0
P1	Project Characteristics (P)	Usage by professional programmers	2	1.75	1.5	1.25	1	0.75	0.5	0.25	0
P2		Project scheduling changes	2	1.75	1.5	1.25	1	0.75	0.5	0.25	0
P3		Project delivery time	2	1.75	1.5	1.25	1	0.75	0.5	0.25	0
P4		System validity and verification	2	1.75	1.5	1.25	1	0.75	0.5	0.25	0
P5		Software response level	2	1.75	1.5	1.25	1	0.75	0.5	0.25	0
P6		Ease of Install & launch	2	1.75	1.5	1.25	1	0.75	0.5	0.25	0
P7		Software documentations	2	1.75	1.5	1.25	1	0.75	0.5	0.25	0
P8		Support Period (maintenance, modification, user training)	2	1.75	1.5	1.25	1	0.75	0.5	0.25	0
P9		Number of main software functions	2	1.75	1.5	1.25	1	0.75	0.5	0.25	0
P10		system development capability	2	1.75	1.5	1.25	1	0.75	0.5	0.25	0
N1	Network Characteristics (N)	Geographical Extends of system	2	1.75	1.5	1.25	1	0.75	0.5	0.25	0
N2		Number of simultaneous users	2	1.75	1.5	1.25	1	0.75	0.5	0.25	0
N3		Access to system support	2	1.75	1.5	1.25	1	0.75	0.5	0.25	0
N4		Reliability in Critical situations	2	1.75	1.5	1.25	1	0.75	0.5	0.25	0
N5		Coordination with present systems	2	1.75	1.5	1.25	1	0.75	0.5	0.25	0

Tab. 12: The concept of ranking of Tab.11

Importance	Ranking
Absolutely low	0
Ultra low	0.25
Very low	0.5
Low	0.75
Normal	1
High	1.25
Very high	1.5
Ultra high	1.75
Absolutely high	2

Very high = Max (comments experts) (8)
High= AVE (comments higher than the average) (9)
Average = AVE (comments experts) (10)
Low = AVE (comments lower than the average) (11)
Very low = Min (comments experts) (12)