The Concept of Prediction of Software Characteristics and Software Projects Success Based on the Analysis of Software Requirements Specifications

Andriy Krasiy, Tetiana Hovorushchenko

Department of System Programming, Khmelnitsky National University, Institutska Str., 11, Khmelnitsky, 29016, UKRAINE,
E-mail: andriy-krasiy@yandex.ua, tat_yana@ukr.net

Abstract – The article shows the importance and the ability to predict the characteristics of software by analyzing the software requirements specification and shows the ability to predict the success of software projects in the early stages of the life cycle.

Keywords – software, software project, software characteristic, software life cycle model, software requirements specification (SRS), quantitative and expert quantitative indicators of specification, artificial neural network (ANN), integrative indicators of projects.

I. Introduction

Software Requirements Specification (SRS) is the foundation for software constructing. It includes the sets of functional and non-functional requirements [1].

Software project is a complex of interrelated activities aimed at achieving the tasks with clearly defined objectives within a specified period of time and with the required budget [2].

At present crisis in the field of software development is becoming more visible - large projects are behind schedule or exceed of the cost estimates, developed product does not have the required functionality, the software has low productivity, software quality don't satisfied the customers [3].

Statistics of software projects success according to The Standish Group International (Chaos reports) [4, 5] is presented in Table 1. Successful are projects, that delivered on time, on budget and have required features and functions; challenged – are projects, that late, over budget, and/or with less than the required features and functions; failed – are projects, that cancelled prior to competition or delivered and never used.

TABLE 1

THE SOFTWARE PROJECTS SUCCESS

	1994	2000	2006	2012
Succes-sful, %	16	28	35	39
Challen-ged, %	53	49	46	43
Failed, %	31	23	19	18

Analysis of the data in Table 1 gives the opportunity to see growth in the number of successful projects and fall in the number of failed projects, but at the same time share of challenged projects is pretty constant and is at least 43%. However, only a small number (up to 39% during 1994-2012 years) projects has the necessary functionality in existing restrictions on times and prices [5].

Software projects are often failed because of bugs at the early stages of the software life cycle, such as [6]: 1) inadequate formulation of requirements; 2) a bad design or ineffective planning; 3) incorrect understanding or insufficient analysis of the specification and of the project; 4) unrealistic project plans - on the cost and duration of the project, efficiency, simplicity and ease of use, quality, reliability and cross-platform (platform independent) developed software, i.e. concerning the main software characteristics; 5) incorrectly selected model of life cycle.

The specification analysis critically affect on software projects and their success. Therefore an analysis of the specification in order to obtain predictable assessments of the main software characteristics is important at the beginning of design stage. Obtained assessments of the main software characteristics will help predict the future success of the developed software project.

II.Analysis of software requirements specification

A large number of specifications and standards for their writing exists. The most comprehensive in terms of identifying key characteristics of software is the software requirements specification (SRS). It describes how software should behave in certain situations, which features should perform.

In analyzing the specification the quantity and quality indicators of information can be obtained. They are useful to identify the main software characteristics at the design stage. All qualitative information require conversion in quantitative information with the help of experts, because necessary quantitative values of software characteristics can be obtained only as a result of study of quantitative information of specification.

For future work, we introduce some definitions.

Definition 1. Quantitative indicators (QI) of SRS are indicators, that have precise quantitative values (for example, "average cost of bug - $ 89").

Визначення 2. Qualitative indicators of SRS are indicators, that are expressed linguistically (eg, "users know how to work with similar products").

Визначення 3. Expert quantitative indicators (EQI) of SRS are indicators, that are expressed linguistically in SRS, but have acquired the quantitative values after processing by the group of experts (eg qualitative indicator "users know how to work with similar products" converted in the EQI "average assessment of skills of users to work with similar products is 3.8 (on a 5-point scale)" after processing by the group of experts).

Analysis of the main software characteristics and indicators of SRS [7, 8] made it possible establish the relationship each main software characteristic with QI and EQI of specification [8, 10].

Therefore, obvious that the software requirements specification contains a significant amount of information useful to determine the quantitative values of the main software characteristics at the design stage.

III. The concept of intelligence prediction of software characteristics based on the analysis of specifications

For the solution of the task of prediction of software characteristics based on QI and EQI of SRS we need to solve the following system of equations [8]:

$$\begin{cases} C = f(X_C) \\ LcD = \varphi(X_{LcD}) \\ SLcM = \phi(X_{SLcM}) \\ EcEf = \gamma(X_{EcEf}) \\ CX = \sigma(X_{CD}) \\ U = \varsigma(X_U) \\ CP = \xi(X_{CP}) \\ Q = \psi(X_Q) \\ R = \omega(X_R) \end{cases} \quad (1),$$

where C - software cost; X_C - the set of quantitative and expert quantitative indicators of SRS, that affect on the assessment of software cost (by Fig.1); LcD - software project duration; X_{LcD} - the set of QI and EQI of SRS, that affect on the assessment of software project duration; $SLcM$ - type of software lifecycle model; X_{SLcM} - the set of QI and EQI of SRS, that affect on the selection of software lifecycle model; $EcEf$ - software effectiveness; X_{EcEf} - the set of QI and EQI of SRS, that affect on the assessment of software effectiveness; CX - software complexity; X_{CX} - the set of QI and EQI of SRS, that affect on the assessment of software complexity; U - software usability; X_U - the set of QI and EQI of SRS, that affect on the assessment of software usability; CP - software cross-platform; X_{CP} - the set of QI and EQI of SRS, that affect on the assessment of software cross-platform; Q - software quality; X_Q - the set of QI and EQI of SRS, that affect on the assessment of software quality; R - software reliability; X_R - the set of QI and EQI of SRS, that affect on the assessment of software reliability.

Theorem of Hecht-Nielsen [9] in the unconstructive form proves the possibility of solution of the task of representation of multivariate function of arbitrary form on neural network. Theorem of Hecht-Nielsen also points for each task the minimum number of neurons in the network needed for its solution. Thus, artificial neural networks (ANN) are versatile structures that allow realize any computational algorithm, summarize information and identify the relationships between the input and the resulting data. Therefore, ANN will be used to implement the function (1).

Then ANN should be developed, which will be process the sets of QI and EQI of specifications, will be approximate the indicators and will be provide the predictable quantitative assessment of software characteristics. Concept of prediction of the main software characteristics based on the analysis of QI and EQI of software requirements specification with using of artificial neural network is presented on Fig.1.

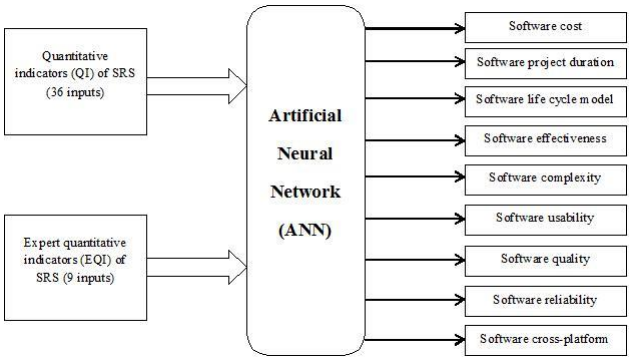

Fig.1 Concept of prediction of software characteristics based on the specifications analysis with using neural network's information technologes.

Difficulty formalizable task of prediction of software characteristics is determination of the weights and the mutual influences of QI and EQI of specifications within each software characteristic. This problem can be solved by using the trained ANN.

Selection of the structure and architecture of the neural network is performed according to the features and complexity of the task. Optimal configuration already exist order to solve certain types of tasks. Since the described task can not be reduced to any of the known types, the problem of synthesis of a new configuration is necessary to solve.

We must make the select of ANN's architecture, given the proposed concept of prediction of software characteristics, before development of neural network's model of prediction of software characteristics based on specification analysis.

Analysis of existing ANN architectures [10] showed that the multilayer perceptron is best suited to solving of the task of analysis of indicators of specifications and prediction of software characteristics. If you use another type of neural network to solving of this task, the nature of ANN will be artificially distorted, ANN's results will be inadequate and inappropriate.

We construct the following architecture of ANN to prediction of the software characteristics based on the analysis of specifications (Fig.2), given the relationship of software characteristics and the QI and EQI of specification, described in [8, 10], and the concept of prediction of software characteristics based on the analysis of specifications with using neural network's information technologies (Fig.1).

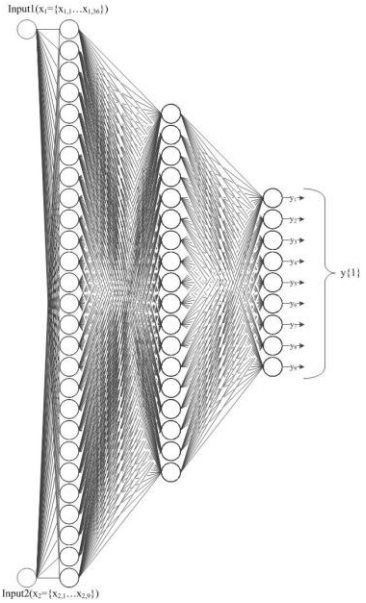

Fig.2 Architecrure of ANN to prediction of software characteristics based on the analysis of specifications.

Quantitative indicators of specification are fed to the set of inputs - Input1, expert quantitative indicators of specification are fed to the set of inputs - Input2. The output vector y{1} is a set of quantitative assessments of the main characteristics of the software (according to Figure 2) in the range [0, 1].

The problems during the realization of such neural network are: 1) establishing of criteria for the creation of expert quantitative indicators from the qualitative indicators of specification; 2) determination of the ranges of input vectors of ANN of prediction of software characteristics; 3) the collecting of a large number of QI, EQI and values of software characteristics from specifications of already developed projects; this data are necessary for the construction of training sample of ANN, represented on Fig.2 (to calculation of the necessary volume of training sample for the ANN, which must training with an error of order 10^{-1}, use the formula [11]:

$N > \frac{h \cdot g}{e_0} = \frac{45 \cdot 45}{10^{-1}} = 20250$ vectors of training sample,

where g – number of input neurons (inputs) of ANN ($g=45$); h – number of neurons in hidden layers of ANN ($h=27+18=45$), $e_0=10^{-1}$- acceptable error of learning).

IV.The concept of intelligence prediction of success of software projects

Some software characteristics require minimization. These characteristics are: software cost, software project duration, software complexity. Therefore their lowest values (nearly to 0) will show a high probability of success of software projects. However, other software characteristics (software effectiveness, software usability, software cross-platform, software quality and software reliability) require maximization. Therefore their highest values (nearly to 1) will show a high probability of success of software projects. Characteristic "life cycle model" is required only to make a decision on the correct choice of the life cycle model, so do not take part in the formation of any of integrative indicators.

Then we introduce two integrative indicators of project (IIP) - Min indicator and Max indicator.

For integrative indicator Min of project we will create a graph in the coordinate system, that has three basis axes (for three characteristics – software cost, project duration, complexity) - Fig.3.

Fig.3 Coordinate system for IIP Min.

For integrative indicator Max of project we will create a graph in the coordinate system, that has five basis axes (for five characteristics – software effectiveness, usability, cross-platform, quality, reliability) - Fig.4.

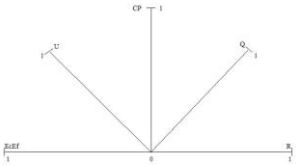

Fig.4 Coordinate system for IIP Max.

Suppose, for example, we have obtained the following values of the software characteristics from ANN for software project №1:

$C = 0.2; LcD = 0.1; CX = 0.15; EcEf = 0.8; U = 0.85;$

$CP = 0.87; Q = 0.89; R = 0.91$.

Then we will have the following graphic representations for integrative indicators Min (Fig.5) and Max (Fig.6) of software project №1.

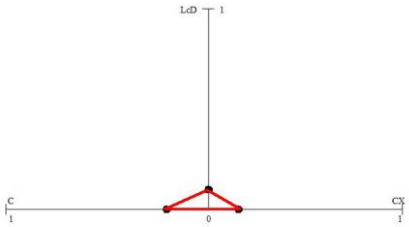

Fig.5 Graphic representation for integrative indicator Min of software project №1.

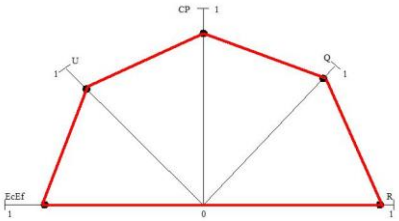

Fig.6 Graphic representation for integrative indicator Max of software project №1.

Integrative indicator Min of project №1 is area S_{min} of the highlighted triangle on Fig.5, and integrative indicator Max of project №1 is area S_{max} of the highlighted pentagon on Fig.6.

Integrative indicators provide the determination of probability of success of software projects.

So, for integrative indicator Min of project the probability of success of the project is:

$$P_{min} = 1 - \frac{S_{min}}{S_{bad}},$$

where S_{bad} - the area of maximum triangle (maximum possible IIP Min), marked by the dotted line on Fig.7.

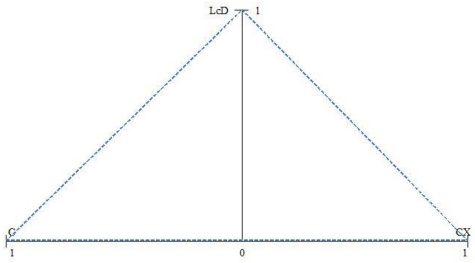

Fig.7 Graphic representation for integrative indicator Min of software project №1 and maximum possible IIP Min.

And for integrative indicator Max of project the probability of success of the project is:

$$P_{max} = \frac{S_{max}}{S_{best}},$$

where S_{best} - the area of maximum pentagon (maximum possible IIP Max), marked by the dotted line on Fig.8.

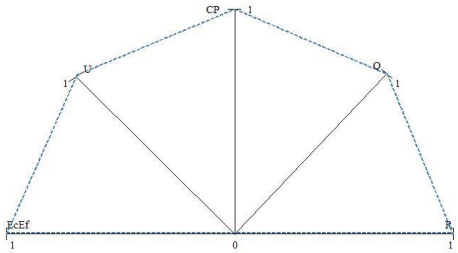

Fig.8 Graphic representation for integrative indicator Max of software project №1 and maximum possible IIP Max.

So, on the basis of the characteristics of the software for several different software projects, we can get the integrative indicators Min and Max for each of projects, designed to solve the same task. Integrative indicators provide to determine of the success of software projects, to compare software projects on the main characteristics and to perform the reasonable and grounded choice of software project for further realization.

Conclusion

The authors have proved that the analysis of specification provides the diversified quantitative and qualitative information for subsequent calculation of the main software characteristics.

The proposed concept of prediction of software characteristics based on the analysis of specifications provided an opportunity to develop a neural network's model for process of prediction of software characteristics. The possibility to take into account the importance (weights) of each specification indicators and the mutual influence of indicators within each software characteristic distinguishes this model from other models. Output ANN functionals, corresponding to the expected quantitative values of software characteristics, provide the assessment of the overall impact of the QI and EQI of specifications for the characteristics of the developed software.

Predictable quantitative values of software characteristics provides the calculation the integrative indicators for various projects, which are used to compare projects, prediction of the success of projects and grounded choice of the project for further realization.

References

[1] IEEE 830-1998. Recommended Practice for Software Requirements Specifications - New York: IEEE, 1998

[2] *R.T.Futrell, D.F.Shafer, L.I.Shafer*, Quality Software Project Management - New York: Prentice Hall PTR, 2003

[3] *O.Pomorova, T.Hovorushchenko*, Intelligent Assessment and Prediction of Software Characteristics at the Design Stage // American Journal of Software Engineering and Applications (AJSEA), 2013; 2(2) - pp. 25-31 // [Electronic resource] - Access mode: http://article.sciencepublishinggroup.com/pdf/10.11648.j.ajsea.20130202.11.pdf

[4] Latest study shows rise in project failures // [Electronic resource] - Access mode: http://kinzz.com/resources/articles/91-project-failures-rise-study-shows

[5] CHAOS Manifesto: Think Big, Act Small, 2013 // [Electronic resource] - Access mode: http://www.versionone.com/assets/img/files/CHAOSManifesto2013.pdf

[6] *O.V.Pomorova, T.O.Hovorushchenko*, Current problems of the software quality evaluation // Radio-electronic and computer systems - Kharkiv: NAU "KhAI", 2013 – № 5, pp.319-327

[7] *T.O.Hovorushchenko, A.V.Krasiy*, Definition of software characteristics and selection of software life cycle model based on the specifications analysis // Transaction of Khmelnitsky National University – Khmelnitsky: KhNU, 2013. - №6, pp.201-208

[8] *T.O.Hovorushchenko, A.V.Krasiy*, The mathematical modelling of software requirements specification and software characteristics // Radioelectronic and computer systsems – Kharkiv: KhAI, 2014 – № 5, pp.34-39

[9] *R.Callan*, The Essence of Neural Networks – Prentice Hall Europe, 2003. – 288 p.

[10] *A.V.Krasiy*, The modelling of process of prediction of software characteristics based on specifications analysis // Scientific journal "Computer-integrated technologies: education, science, industry" - Lutsk: Lutsk national technical university, 2014 - pp.66-76

[11] *F. Wasserman*, Neurocomputer Techniques: Theory and Practice [Russian translation]. Moscow: Mir, 1992. - 240 p.