Intelligent Support of Process of Evaluation and Prediction of Software Complexity and Quality

Oksana Pomorova, Tetyana Hovorushchenko

System Programming Department, Khmelnitskiy National University, Institutska Str., 12, Khmelnitskiy, 29016, UKRAINE,

e-mail: o.pomorova@gmail.com, tat_yana@ukr.net

Abstract – Research is dedicated to development of artificial neural network's method and intelligence system of evaluation and prediction of software complexity and quality, which provides the realization of comparative analysis of different project versions and selection of the best of them accordance its characteristics on the basis of design stage metrics analysis.

Key words – software, software metric, Safety Case methodology, artificial neural network.

I. Introduction

Safety Case methodology (Safety computer-aided software engineering) is developed over 20 years [1]. The primary object of Safety Case methodology is to minimize the software security and commercial risks by constructing a report, which should provides evidences, reasons and arguments that software is safe, and all requirements for the software is properly implemented. Currently, this methodology is generally accepted, but the level of its automation is still low.

The process of software developing for the Safety Case methodology depends on a large number of documents (requirements, standards, project specification), source code, software evaluation methods and analysis of their results, software testing results and the degree of its documentation. Figure 1 [2] represents the generic Safety Case model.

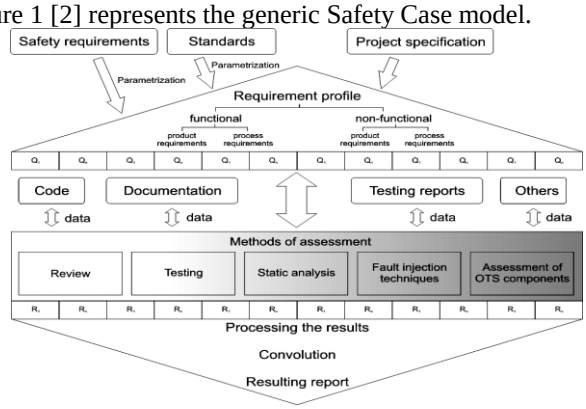

Fig.1 The generic Safety Case model

The basic parts of Safety Case model:

- *software requirements profile* - (including standards for software development, subject domain standards and customer requirements) - functional, inverse and non-functional requirements for software systems are analyzed; requirements for software safety and security are researched and completeness of all kinds of requirements is estimated;
- *software analysis results profile* - metric analysis results, source code and software test results are studied and analyzed;
- *evaluation of obtained software accordance to its requirement.*

One of the main tools of software quality analysis and evaluation is metric analysis. Software metric is a measure, that provides obtaining of numerical values of some software properties or its specifications.

Unsolved tasks of metric analysis results processing: 1) lack of unified standards for metrics, therefore each developer of "measurement" system offers own software quality evaluation measures and proper metrics; 2) the task of the metrics values interpretation are also difficult - for most users the metrics and its values are not informative; 3) at the software design stage the most attention at the project choice is paid to design cost and time, software company reputation and software engineering technologies, but the decisions, taken on the basis of these parameters, not guarantee software quality; 4) according to statistical data [3], only 1,5% software companies are trying to quantify evaluate the quality of processes and ready product using metrics, and only 0,5% software companies are trying to improve its work on the basis of software quality quantitative criteria with the purpose of defect-free products producing.

Considering the software evaluation methods analysis results conclusion was drawn, that the perspective research direction is development of intelligence methods and systems of evaluation and prediction of software complexity and quality.

II. Software Design Stage Metrics

9 software metrics of design stage with the exact values and 15 software metrics of design stage with the predicted values were chosen as the basic metrics for evaluation and prediction of software complexity and quality. The ranges of software design stage metrics values are determined in Table 1.

TABLE 1

SOFTWARE DESIGN STAGE METRICS VALUES RANGES

№	Design Stage Metrics with the Exact Values	Design Stage Metrics with the Predicted Values
(1)	(2)	(3)
1	Chepin's metric: -1, 0..32500	Expected Lines Of Code (LOC): -1, 0..50000
2	Jilb's metric (absolute modular complexity): -1, 0..2450	Halstead's metric: -1, 0..1562500
3	McClure's metric: -1, 0..120050	McCabe's metric: -1, 0..2402

(1)	(2)	(3)
4	Kafur's metric: -1, 0..497500	Jilb's metric (relative logical complexity): -1, 0..1
5	Cohesion metric: -1, 0..10	Expected quantity of program statements: -1, 0..50000
6	Coupling metric: -1, 0..9	Expected estimate of interfaces complexity: -1, 0..1
7	Metric of the global variables calling: -1, 0..1	Software design total time: -1, 0..520 (working days)
8	Time of models modification: -1, 0..46	Design stage time: -1, 0..182 (working days)
9	Quantity of found bugs during the models and prototype inspection: -1, 0..5000	Software design expected cost: -1, 0..25000 (usd)
10		Software quality audit expected cost: - 1, 0..2500 (usd)
11		Software realization productivity: -1, 0..5 (minutes for 1 line of code)
12		Program code realization expected cost: -1, 0..8750 (usd)
13		Expected functional points (FP): -1, 0..2945
14		Effort applied by Boehm's model: -1, 0..394 (man-months)
15		Expected development time by Boehm's model: -1, 0..520 (working days)

The design results evaluation and software complexity and quality characteristics prediction will be the results of processing of metrics from Table 1.

III.Artificial Neural Network's Method Of Evaluation And Prediction Of Software Complexity And Quality (NMEP)

NMEP provides the complexity and quality evaluation of the project and prediction of designed software on the basis of exact and predicted values design stage metrics of complexity and quality.

NMEP based on processing of the following sets:

- the set of the design stage complexity metrics with the exact values $CMEV = \{cmev_i \mid i = 1..4\}$;
- the set of the design stage quality metrics with the exact values $QMEV = \{qmev_j \mid j = 1..5\}$;
- the set of the design stage complexity metrics with the predicted values $CMPV = \{cmpv_k \mid k = 1..6\}$;
- the set of the design stage complexity metrics with the predicted values $QMPV = \{qmpv_n \mid n = 1..9\}$.

The results of these sets processing are:

- project complexity estimate PCE;
- project quality evaluation PQE;
- designed software complexity prediction SCP;
- designed software quality prediction SQP.

The basis to provide of project complexity estimate are elements of set $CMEV$. The basis to provide of project quality evaluation are elements of sets $CMEV$ and $QMEV$. The basis to provide of designed software complexity prediction are elements of set $CMPV$, but elements of sets $CMEV$ and $QMEV$ are taken into account. The basis to provide of designed software quality prediction are elements of sets $CMPV$ and $QMPV$, but elements of sets $CMEV$ and $QMEV$ are taken into account.

The problem of identifying the correlation between metrics values and quality and complexity of project and software should be solved for the software quality and complexity evaluation and prediction on the basis of metric analysis. One of the means, which makes it possible to summarize the information and identify dependencies between input data and resulting data, are artificial neural networks.

Metric analysis results we will to process using artificial neural network (ANN) of multilayer perceptron architecture.

Input data for ANN are the sets of design stage complexity and quality metrics with the exact values and the sets of design stage complexity and quality metrics with the predicted values.

The results of ANN functioning are 4 characteristics: project complexity estimate; project quality evaluation; designed software complexity prediction; designed software quality prediction.

NMEP concept is represented on Figure2.

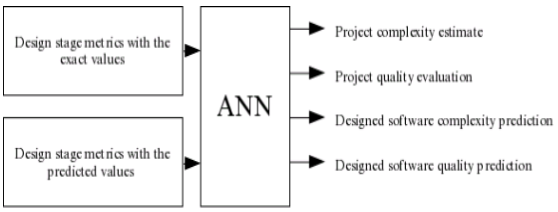

Fig.2 NMEP concept

ANN has 9 one type inputs for the quantitative values of design stage exact metrics and 15 another type inputs for the quantitative values of design stage predicted metrics. If a certain metric is not determined, then -1 is given on the proper ANN input.

Project complexity estimate, project quality evaluation, designed software complexity prediction, designed software quality prediction are values in the range [0, 1], where 0 - proper metrics were not determined, approximately 0 - the project or designed software has a high complexity or low quality and 1 - the project or software is simple or high quality.

The above dependences of the resulting evaluations on the input sets of metrics are taken into account at ANN training.

The conclusion about the project quality and complexity and the expected quality and complexity of designed software is based on an analysis of 4-th obtained results.

The ANN has 24 neurons of the input layer, 14 neurons of approximating layer and 8 neurons of the adjusting layer and 4 neurons of the output layer. The ANN layers structural scheme in Simulink is shown on Figure 3.

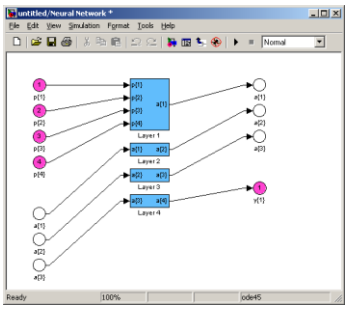

Fig.3 ANN layers structural cheme in Simulink

Realized neural network was trained with training sample of 1935 vectors and tested with testing sample of 324 vectors by one step secant backpropagation method (OSS). The training performance is $\xi = 0{,}102197$. ANN training and testing by OSS are represented on Figure 4, where the bottom line shows the ANN training and the top line - ANN testing.

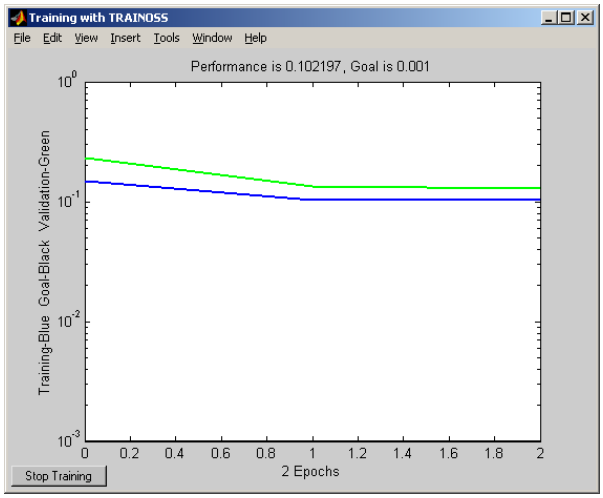

Fig.4 ANN training and testing by OSS method

IV.Intelligence Decision Support System Of Process Of Evaluation And Prediction Of Software Complexity And Quality (ISCQEP)

ISCQEP is designed to the evaluation of design stage results and prediction of software complexity and quality characteristics on the basis of processing of design stage metrics with exact and predicted values. Quantitative exact and predicted values of design stage metrics is given to the ISCQEP input, and conclusions about the project and designed software complexity and quality are the results of the system functioning. Structure of ISCQEP represented on Figure 5.

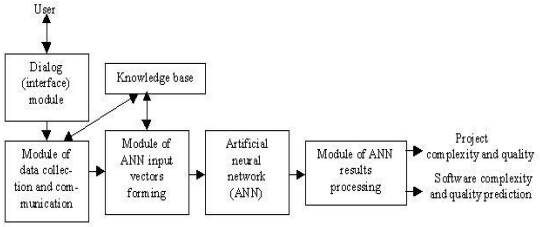

Fig.5 ISCQEP Structure

ISCQEP consists of next modules: 1) dialog (interface) module; 2) module of data collection and communication; 3) knowledge base; 4) module of ANN input vectors forming; 5) artificial neural network; 6) module of ANN results processing.

The *dialog (interface) module* visualizes the functioning of module of data collection and communication, displays the system functioning and produces the messages to user in an understandable form for him.

The *module of data collection and communication* reads the user information about the quantitative values of exact and predicted metrics of software design stage, saves the obtained information in the knowledge base and transmits its to the module of ANN input vectors forming.

Knowledge base contains the quantitative values of exact and predicted metrics of software design stage, the ANN input vectors and the rules of ANN results processing.

The *artificial neural network (ANN)* provides the approximation of software design stage metrics and gives the quantitative evaluation of project complexity and quality and prediction of designed software complexity and quality by NMEP.

The *module of ANN input vectors forming* prepares the metrics values of the knowledge base for the ANN inputs.

The *module of ANN results processing* makes the conclusions about the project quality and complexity and the expected quality and complexity of designed software on the basis of ANN results and the following rules:

- if $PCE = 0$, then design stage complexity metrics with the exact values were not determined;
- if $PCE \to 0$, then the project is complicated to realization;
- if $PCE \to 1$, then the project is simple to realization;
- if $PQE = 0$, then design stage quality metrics with the exact values were not determined;
- if $PQE \to 0$, then project is a low quality;
- if $PQE \to 1$, then the project satisfies the customer requirements in quality;
- if $SCP = 0$, then design stage complexity metrics with the predicted values were not determined;
- if $SCP \to 0$, then designed software will have significant complexity;

- if $SCP \to 1$, then designed software is expected simple;
- if $SQP = 0$, then design stage quality metrics with the predicted values were not determined;
- if $SQP \to 0$, then designed software is low quality;
- if $SQP \to 1$, then high quality software is expected.

Using these rules, ISCQEP gives the evaluation of project complexity and quality and prediction of the developed software complexity and quality, which help the customer to make the right decisions about project selection.

V.ISCQEP Functioning Results Using

ANN results analysis was performed on the basis of projects developed by software company "STU-Electronics", Khmelnitsky (Ukraine). Examples of results for 2 projects are shown in Table 2.

TABLE 2

PROCESSING OF DESIGN STAGE METRICS USING ANN

№	Stage Design Metrics	ANN Result
1	McClure's metric - 90000 Kafur's metric - 376900 Chepin's metric - 24530 Cohesion metric - 3 Coupling metric - 7 Expected Lines Of Code (LOC) - 40135 Halstead's metric - 124928 McCabe's metric - 1903 Expected functional points (FP) - 2220	Y_1=0,24 Y_2=0,31 Y_3=0,21 Y_4=0,28
2	McClure's metric - 12000 Kafur's metric - 64238 Chepin's metric - 3241 Cohesion metric - 9 Coupling metric - 3 Metric of the global variables calling - 0,089 Expected Lines Of Code (LOC) - 6240 Halstead's metric - 162251 Jilb's metric - 0,12 McCabe's metric - 298 Effort applied by Boehm- 60 man-months	Y_1=0,91 Y_2=0,89 Y_3=0,90 Y_4=0,88

According to the results: 1-st project is complicated and low quality; the designed software will have the same characteristics; 2-nd project is simple and has high quality, the designed software is also expected a simple and high quality.

ISCQEP conclusions allow to compare the different project versions, when the cost and time is approximately equal, but the decisions on the basis of these parameters are not always guarantee the proper software quality. ISCQEP conclusions allow to compare the different project versions, when the cost and time is approximately equal. For example, data about 2 projects of software company "STU-Electronics" (Khmelnitsky, Ukraine) are considered in Table 3.

TABLE 3

THE CRITERIONS OF PROJECT CHOICE

№	Values Y_1, Y_2	Values Y_3, Y_4	Design cost	Design time
1	Y_1=0,81 Y_2=0,84	Y_3=0,78 Y_4=0,77	13000 usd	200 working days
2	Y_1=0,21 Y_2=0,24	Y_3=0,30 Y_4=0,29	13250 usd	220 working days

The project characteristics from Table 3 evidence that both versions have approximately the same design cost and time, but significantly different estimates of project complexity and quality and prediction of designed software complexity and quality. On the basis of only cost and time software company can make a false conclusion about project selection. ISCQEP conclusions help to make the right selection

Acknowledgments

The necessity and actuality of scientific research in software quality evaluation and prediction comes from the results of the software metric evaluation methods analysis.

The proposed artificial neural network's method and intelligence system of evaluation and prediction of software complexity and quality provide the motivated and grounded decision about selection of project on the basis not only cost and time, but also considering project and designed software complexity and quality.

Unsolved problems are:

- metric information lack to increasing of the training and testing samples size;
- need the such diverse utilities to comparing of metric information processing results of this project;
- need the development of designed software complexity evaluation metrics from the viewpoint of the maintenance difficulty or simplicity, usability and the effectiveness of the methods chosen to solve the task.

References

[1] P. Bishop and R. Bloomfield, A "Methodology for Safety Case Development", <http://www.adelard.com/papers/sss98web.pdf> (Feb. 1998).

[2] A. Gordeyev and V. Kharchenko, eds. "Case-Based Software Reliability Assessment by Fault Injection Unified Procedures" (*paper presented at the 2008 International Workshop on Software Engineering in East and South Europe, Leipzig, May 2008*), pp.1-8

[3] V.Lipayev, Selection and evaluation of software quality characteristics: methods and standards (Moscow: Sinteg, 2001), 224 p.