THE INTELLIGENCE SYSTEM OF SOFTWARE COMPLEXITY AND QUALITY EVALUATION AND PREDICTION

Oksana Pomorova, Tetyana Hovorushchenko

Abstract: Considering the software evaluation methods analysis results conclusion was drawn, that the perspective research direction is development of intelligent systems, which will be analyzed and processed the design stage metrics analysis results and will be provided the project evaluation and the designed software characteristics prediction. Intelligence System of Software Complexity and Quality Evaluation and Prediction (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. Today the main parameters in the selection of software project version are the design cost and time and designing company reputation, but a 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. The proposed intelligence system of software complexity and quality evaluation and prediction provides 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.

Keywords: software, software metric, software metric of design stage, software quality, software complexity, Safety Case methodology, artificial neural network.

Introduction

Safety Case methodology (Safety computer-aided software engineering) is developed over 20 years [1, 2]. 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 [3, 4] represents the generic Safety Case model.

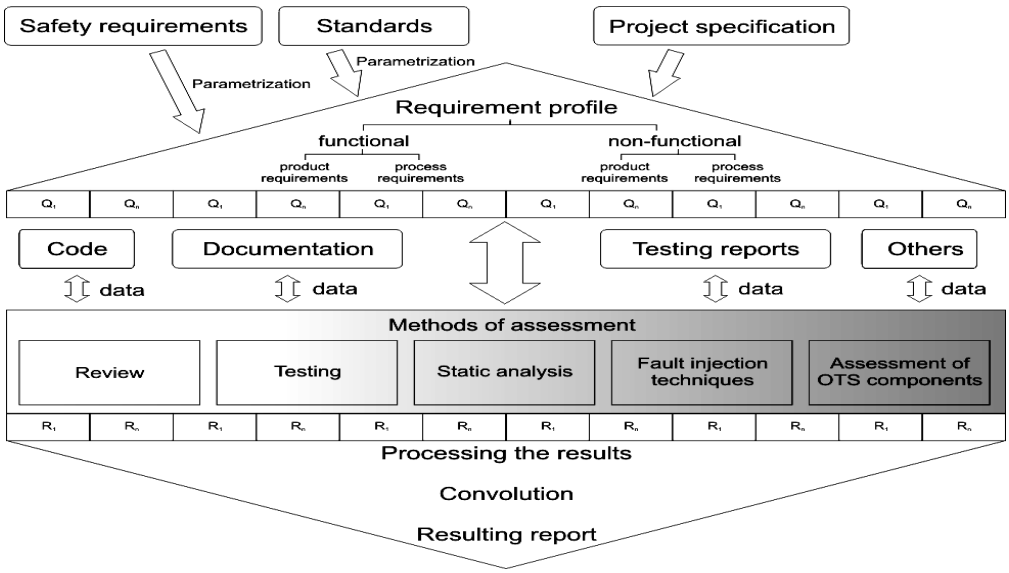

Figure 1: The Generic Safety Case Model

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 (results profile) accordance to its requirements (requirements profile).

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.

In spite of numerous researches of software metrics many unsolved tasks remain in this domain. One of these tasks is the lack of unified standards for metrics (over a thousand metrics were created), therefore each provider of "measurement" system offers own software quality evaluation measures and proper metrics. The task of the metrics values interpretation is also difficult - for most users the metrics and its value are not informative. 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. According to statistical data [5], 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. Quality measurement technology has not yet reached maturity, since only 0.5% software companies are on optimizing, mature level of Capability Maturity Model (CMM).

Considering the software evaluation methods analysis results conclusion was drawn, that the perspective research direction is development of intelligent systems, which will be analyzed and processed the design stage metrics analysis results and will be provided the project evaluation and the designed software characteristics prediction.

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 (Table 1) were chosen as the basic metrics for processing by intelligence system of software complexity and quality evaluation and prediction (ISCQEP).

Table 1. Metrics of Design Stage with the Exact and Predicted Values

№	Design Stage Metrics with the Exact Values		Design Stage Metrics with the Predicted Values	
	Complexity Metrics	Quality Metrics	Complexity Metrics	Quality Metrics
(1)	(2)	(3)	(4)	(5)
1	Chepin's metric	Cohesion metric	Expected Lines Of Code (LOC)	Software design total time
2	Jilb's metric (absolute modular complexity)	Coupling metric	Halstead's metric	Design stage time
3	McClure's metric	Metric of the global variables calling	McCabe's metric	Software design expected cost (USD)
4	Kafur's metric	Time of models modification	Jilb's metric (relative logical complexity)	Software quality audit expected cost (USD)

(1)	(2)	(3)	(4)	(5)
5		Quantity of found bugs during the models and prototype inspection	Expected quantity of program statements	Software realization productivity (minutes for 1 line of code)
6			Expected estimate of interfaces complexity	Program code realization expected cost (USD)
7				Expected functional points (FP)
8				Effort applied by Boehm's model (man-months)
9				Expected development time by Boehm's model (months)

The Structure of Intelligence System of Software Complexity and Quality Evaluation and Prediction (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 are 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 is represented on Fig. 2.

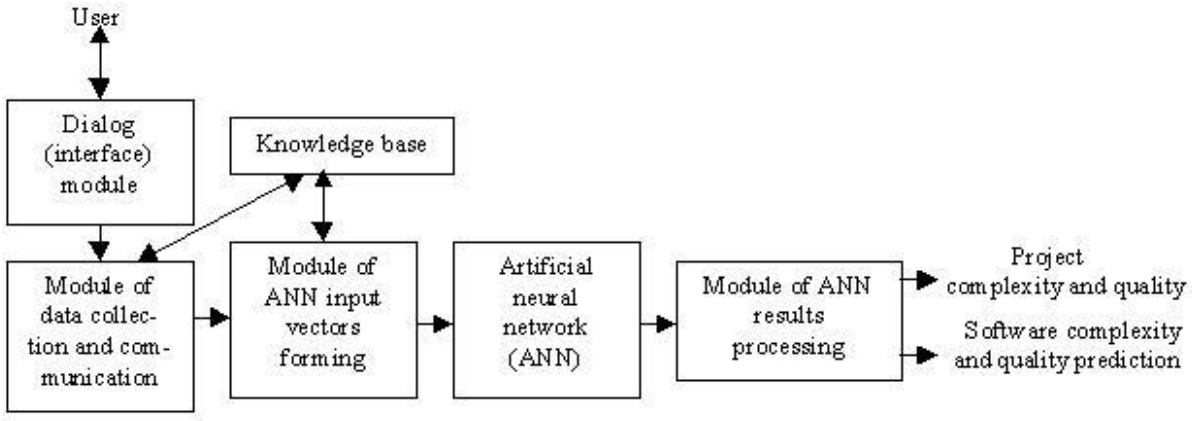

Figure 2: ISCQEP Structure

ISCQEP consists of next modules:

- dialog (interface) module;
- module of data collection and communication;

- knowledge base;
- module of ANN input vectors forming;
- artificial neural network;
- 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 characteristics. Input data for ANN are the set of the design stage metrics with the exact values $TMP = \{ tmp_a \mid a = 1..9 \}$ and the set of the design stage metrics with the predicted values $PMP = \{ pmp_b \mid b = 1..15 \}$. If a certain metric was not determined, the proper element of set will be equal -1.

The *module of ANN input vectors forming* prepares the metrics values of the knowledge base for the ANN inputs. ANN has 9 inputs x' and 15 inputs x. The quantitative values of design stage exact metrics are given on inputs x', and the quantitative values of design stage predicted metrics are given on inputs x. The value of i-th TMP element is the value of input x'_i ($i = 1..9$), the value of j-th PMP element is the value of input x_j ($j = 1..15$).

Multilayer perceptron is ANN for solving of task of the metrics analysis and software quality characteristics prediction. This 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 Fig.3.

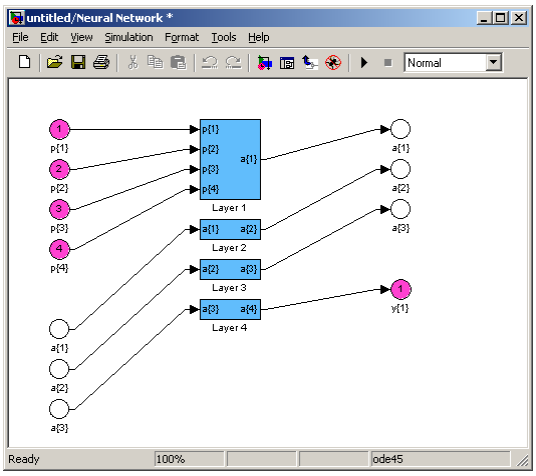

Figure 3: ANN Layers Structural Scheme in Simulink

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 $\xi = 0{,}102197$.

The ANN handles the set of input vectors and gives 4 output values from the range [0, 1]: *OSP* - project complexity estimate; *OQP* - project quality evaluation; *PSPZ* - designed software complexity prediction; *PQPZ* - designed software quality prediction. The complexity characteristics include not only the simplicity or complexity of designed software from the viewpoint of its size, cost and design time, but also the maintenance difficulty or simplicity, usability and the effectiveness of the methods chosen to solve the task.

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 the following rules:

- if $OSP = 0$, then design stage complexity metrics with the exact values were not determined;
- if $OSP \to 0$, then the project is complicated to realization;
- if $OSP \to 1$, then the project is simple to realization;
- if $OQP = 0$, then design stage quality metrics with the exact values were not determined;
- if $OQP \to 0$, then project is a low quality;

- if $OQP \to 1$, then the project satisfies the customer requirements in quality;
- if $PSPZ = 0$, then design stage complexity metrics with the predicted values were not determined;
- if $PSPZ \to 0$, then designed software will have significant complexity;
- if $PSPZ \to 1$, then designed software is expected simple;
- if $PQPZ = 0$, then design stage quality metrics with the predicted values were not determined;
- if $PQPZ \to 0$, then designed software is low quality;
- if $PQPZ \to 1$, then high quality software is expected.

Example of Forming of Conclusions About Project and Software Complexity and Quality by ISCQEP

The metric analysis results of 3 projects (Table 2) of software company "STU-Electronics" (Khmelnitsky, Ukraine) will be used as the ISCQEP input data.

Table 2. The Metric Analysis Results of Software Company "STU-Electronics" Projects

№ pr.	Stage Design Metrics with the Exact Values		Stage Design Metrics with the Predicted Values	
	Complexity Metrics	Quality Metrics	Complexity Metrics	Quality Metrics
(1)	(2)	(3)	(4)	(5)
1	Chepin's metric - 22750 Jilb's metric (absolute modular complexity) - 1730 McClure's metric - 84050 Kafur's metric - was not determined	Cohesion metric - 3 Coupling metric - 7 Metric of the global variables calling - was not determined Time of models modification - 33 Quantity of found bugs during the models and prototype inspection - was not determined	Expected LOC - 35300 Halstead's metric - was not determined McCabe's metric - 1680 Jilb's metric (relative logical complexity) - 0,7 Expected quantity of program statements - 35000 Expected estimate of interfaces complexity - was not determined	Software design total time - 364 Design stage time - 127 Software design expected cost - 17500 Software quality audit expected cost - was not determined Software realization productivity - was not determined Program code realization expected cost - 6125 Expected FP - 2100 Effort applied (Boehm) - 283 Expected development time (Boehm) - was not determined

(1)	(2)	(3)	(4)	(5)
2	Chepin's metric - 16250 Jilb's metric (absolute modular complexity) - 1250 McClure's metric - 60050 Kafur's metric - 257000	Cohesion metric - 5 Coupling metric - 4 Metric of the global variables calling - 0,5 Time of models modification - 24 Quantity of found bugs during the models and prototype inspection - 2500	Were not determined	Were not determined
3	Were not determined	Were not determined	Expected LOC - 10800 Halstead's metric - 312501 McCabe's metric - 480 Jilb's metric (relative logical complexity) - 0,2 Expected quantity of program statements - 10000 Expected estimate of interfaces complexity - 0,2	Software design total time - 104 Design stage time - 37 Software design expected cost - 5000 Software quality audit expected cost - 500 Software realization productivity - 1,1 Program code realization cost - 1750 Expected FP - 560 Effort applied (Boehm) - 80 Expected development time (Boehm) - 5

After metric analysis data receiving the module of data collection and communication transmits this information to the module of ANN input vectors forming and saves in the knowledge base data section. Module of ANN input vectors forming creates the following vectors for ANN inputs (Table 3).

Table 3. Example of ANN Inputs Vectors

№ pr.	Input1 (x_1')	Input2 (x_2')	Input3 (x_1)	Input4 (x_2)
1	{[22750;1730;84050;0];	[3;7;0;33;0];	[35300;0;1680;0.7;35000; 0];	[364;127;17500;0; 0;6125;2100;283;0]}
2	{[16250;1250;60050;257000];	[5;4;0.5;24;2500];	[0;0;0;0;0;0];	[0;0;0;0;0;0;0;0;0]}
3	{[0;0;0;0];	[0;0;0;0;0];	[10800;312501;480;0.2; 10000;0.2];	[104;37;5000;500;1.1; 1750;560;80;5]}

ANN handles the input vectors and gives the results, on the basis of which the module of ANN results processing makes certain conclusions (Table 4) according to the rules of knowledge base rules section.

Table 4. ANN Results and Conclusions about Analyzed Projects

№ pr.	Value OSP and ISCQEP conclusion	Value OQP and ISCQEP conclusion	Value PSPZ and ISCQEP conclusion	Value PQPZ and ISCQEP conclusion
1	0.32	0.29	0.33	0.31
	The project is sufficiently complicated to realization	The project has low quality	The designed software will has significant complexity	The designed software will has low quality
2	0.50	0.51	0	0
	The project has medium complexity	The project has medium quality	The design stage complexity metrics with the predicted values were not determined	The design stage quality metrics with the predicted values were not determined
3	0	0	0.83	0.80
	The design stage complexity metrics with the exact values were not determined	The design stage quality metrics with the exact values were not determined	The designed software will has significant complexity	High quality software is expected

Data of Table 4 shows, that for the project №2 the design stage complexity and quality metrics with the predicted values were not determined, therefore ISCQEP can not make the conclusion about designed software complexity and quality. For the project №3 the design stage complexity and quality metrics with the exact values were not determined, therefore ISCQEP can not evaluate the project complexity and quality. The research of projects with determined all groups of metrics [5] shows that the project quality and complexity evaluations and the designed software complexity and quality evaluations are approximately equal. According to these results the conclusion can be made, that the simplest and most high-quality project is the project №3. The software designed for project number 3 will be a simple and high quality also.

ISCQEP Functioning Results Using

Today the main parameters in the selection of software project version are the design cost and time and designing company reputation, but a 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 5.

Table 5. The Criterions of Project Choice

№ project	Values Y_1, Y_2	Values Y_3, Y_4	Design Cost	Design Time
1	Y_1=0,85; Y_2=0,86	Y_3=0,81; Y_4=0,87	12500 USD	250 working days
2	Y_1=0,22; Y_2=0,25	Y_3=0,28; Y_4=0,27	13125 USD	260 working days

The project characteristics from Table 5 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 in favor of project №1, which in this case has the best complexity and quality characteristics of two proposed projects.

Acknowledgment

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 intelligence system of software complexity and quality evaluation and prediction provides 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.

Notes

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

2. T. Kelly, "Arguing Safety – A Systematic Approach to Managing Safety Cases", <http://www-users.cs.york.ac.uk/~tpk/tpkthesis.pdf> (Sept. 1998).

3. 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), 1-8.

4. K. Netkachova, "Safety Case Methodology: Architecting Principles", *Radioelectronic and computer systems,* 3 (2010): 109-112

5. V.Lipayev, *Selection and evaluation of software quality characteristics: methods and standards* (Moscow: Sinteg, 2001), 224.

Authors:

OKSANA POMOROVA, Doctor of technical sciences, Professor; IEEE Member; Head of System Programming Department of Khmelnitsky National University, Ukraine. In 1992 she graduated from TARAS SHEVCHENKO NATIONAL UNIVERSITY OF KYIV at Applied Mathematics specialty. In 2001 received she gained a PhD degree at the specialty 05.13.06 - "Automated systems and advanced information technologies". In 2007 she defended doctoral thesis "Theoretical foundations, methods and means for diagnosing computer systems" at the specialty 05.13.13 - "Computers, systems and networks". Research interests: Critical software verification and validation; Methodology and means of intelligent diagnosis of computer systems; Modeling of knowledge bases and specialized computer systems. She is initiator and manager of several Ukrainian research projects and international TEMPUS projects which are hold by the Department of Systems Programming.. She has more than 100 publications and is an author of 8 manuals and monographs.

TETYANA HOVORUSHCHENKO, PhD, Senior Researcher, Associate Professor at the System Programming Department of Khmelnitsky National University, Ukraine. In 2002 she graduated with honors from Technological University of Podillya, has a degree "Master of Computer Engineering". In 2007 she defended her PhD thesis "Software testing process reliability increasing on basis of neuronet information technologies" on specialty 05.13.06 - "Automated systems and advanced information technologies". Research interests: Intelligent diagnosis of software; Intelligent Evaluation and Prediction of Software Complexity and Quality Characteristics. Autor of more than 50 publications.