Forming the Logical Conlcusion about Sufficiency of Information of Software Requirements Specification for Software Quality Assessment by ISO 25010:2011

Tetiana Hovorushchenko
Computer Engineering & System Programming Department
Khmelnitsky National University
Khmelnitsky, Ukraine
tat_yana@ukr.net

Abstract— **The aim of this study is the development of the method of generating and filling the template of ontology for the quality of the concrete software and method of forming the logical conclusion about the sufficiency of information of software requirements specification (SRS) for software quality assessment (by ISO 25010:2011). The first time proposed method of generating and filling the template of ontology for the quality of the concrete software leaves in the relevant ontology only measures, which are available in the SRS for the concrete software. It provides forming the ontology for the quality of the concrete software. The first time developed method of forming the logical conclusion about the sufficiency of the SRS information for software quality assessment provides the conclusion about the sufficiency or insufficiency of the SRS information for software quality assessment by ISO 25010:2011. If the SRS information is insufficient, this method: provides the conclusion for which characteristics and subcharacteristics the information is insufficient; creates the sorted list of missing measures as the recommended priority of their addition in the SRS; evaluates the veracity of software quality assessment based on the available in the SRS measures.**

Keywords—software requirements specification (SRS), sufficiency of SRS information, software quality, logical conclusion, ontology, template of ontology for quality of the concrete software, evaluation of veracity of software quality assessment

I. INTRODUCTION

Statistics [1, 2] shows, that now the crisis in the field of software quality assurance is ongoing – large projects are behind schedule or have the excess of cost estimates, developed product doesn't have the necessary functionality, its performance is low, software quality is not satisfied the customers. The key factor ensuring the effective use of software and one of the basic needs of customers are high software quality.

Many software bugs arise during requirements formulation – such bugs constitute 10-23% of all software bugs [3]. Moreover the greater the software size, the more bugs are contributed during the requirements formulation [3]. The majority of accidents with software occurred because of the false requirements but not because of coding bugs [4]. Software projects with failed requirements and specifications cannot have the successful implementation. So now the actual and important task is the SRS analysis, possibility to "cut off" the software projects with failed or incomplete (with insufficient information) specification. The *sufficiency of information* is the rational information saturation that eliminates information incompleteness (lack of necessary information).

Currently, the software quality evaluation by standard ISO 25010:2011 [5] is as follows – the software quality is evaluated on the basis of the 8 characteristics, the characteristics are evaluated on the basis of 31 subcharacteristics, and the subcharacteristics are evaluated on the basis of 203 measures, which are specified in ISO 25023:2016 [6]. The basic idea of ISO 25010:2011 [5] is exactly comprehensive quality evaluation, with taking into account all the quality measures, subcharacteristics and characteristics. Today the evaluation of the software quality measures is conducted only for the ready source code [7]. But exactly the software requirements provide the definition of the necessary characteristics of software quality and impact on methods of quantitative evaluation of software quality. So the SRS have all measures, which are needed to the subcharacteristics and characteristics calculation [7], i.e. the information sufficiency (as presence in the SRS all necessary measures) for future software quality assessment can be evaluated on the basis of the SRS.

The research of the methods of the SRS analysis [8, 9] showed that they intended to monitoring the implementation of the requirements and don't evaluate the sufficiency of the SRS information. So the *actual task* is the development of the methodology of evaluating the sufficiency of the SRS information for software quality assessment. Some methods, which constitute such methodology, already are developed by the author – the method of evaluating the sufficiency of the SRS information for software quality assessment (by ISO 25010:2011) based on the ontology [10], the method of evaluating the weights of software quality measures [11], the method of evaluating the sufficiency of the SRS information for software quality assessment (by ISO 25010:2011) based on the weighted ontology [11]. But for completion of such methodology, the method of generating and filling the template of ontology for the quality of the concrete software and method of forming the logical conclusion about the sufficiency of the

SRS information for software quality assessment (by ISO 25010:2011) should be developed, that is the *aim of this study*.

II. Method of Generating and Filling the Template of Ontology for the Quality of The Concrete Software

In [10] the base ontology for subject domain "Software engineering" (part "Software quality") was developed in Protégé 4.2. The developed base ontology has all 138 software quality measures (according to the standard [6], the software quality subcharacteristics and characteristics depend on 203 measures, but only on 138 different measures), which can be contained in the SRS.

The *method of generating and filling the template of ontology for the quality of the concrete software* consists of next steps:

- to open the base ontology for subject domain "Software engineering" (part "Software quality") in Protégé 4.2;
- to remove from the base ontology all measures that aren't found in the SRS for the concrete software (on the basis of the results of analysis of the SRS for the concrete software according to step №1 of method of evaluating the sufficiency of the SRS information for software quality assessment (by the standard ISO 25010:2011) based on the ontology [10]);
- to save these changes, thus creating the ontology for the quality of the concrete software.

III. Method of Forming the Logical Conclusion about the Sufficiency of the SRS Information for Software Quality Assessment (by ISO 25010:2011)

First of all the *production rules of forming the logical conclusion about the sufficiency of the SRS information for software quality assessment by ISO 25010:2011* (the set PR={pr_1,…,pr_{140}}) were developed on the basis of the base [10] and the weighted base [11] ontologies for subject domain "Software engineering" (part "Software quality"):

- if in the SRS the measure «Operation Time» is absent, then: fcr:=fcr+1 (counter of missing measures for Functional Correctness subcharacteritic), fa:=fa+1 (for Functional Appropriateness), ma:=ma+1 (for Maturity), av:=av+1 (for Availability), rvb:=rvb+1 (for Recoverability), tb:=tb+1 (for Time Behaviour), ru:=ru+1 (for Resource Utilization), lb:=lb+1 (for Learnability), ob:=ob+1 (for Operability), md:=md+1 (for Modularity), mfb:=mfb+1 (for Modifability), tst:=tst+1 (for Testability), cf:=cf+1 (for Confidentiality), ig:=ig+1 (for Integrity), cex:=cex+1 (for CoExistence), io:=io+1 (for Interoperability), ab:=ab+1 (for Adaptability) and, respectively, fy:=fy+2 (the information is insufficient for the calculation of 2 subcharacteristics of Functional Suitability characteristic, so counter of missing measures for this characteritic is increased by 2), ry:=ry+3 (counter of missing measures for Reliability characteristic), ey:=ey+2 (for Performance Efficiency), uy:=uy+2 (for Usability), my:=my+3 (for Maintainability), sy:=sy+2 (for Security), cy:=cy+2 (for Compatibility), py:=py+1 (for Portability), i.e. the SRS information is insufficient for the calculation 17 from 31 software quality subcharacteristics and all 8 software quality characteris-tics; `mas[Operation Time]:=17/138` (in the appropriate element of array `mas` the weight of missing measure (from the weighted base ontologies for subject domain "Software engineering" (part "Software quality") [11]) is written);
- if in the SRS the measure «Number Of Faults» is absent, then: ma:=ma+1, ab:=ab+1, ib:=ib+1 (counter of missing measures for Installability) and, respectively, ry:=ry+1, py:=py+2, i.e. the SRS information is insufficient for the calculation 3 from 31 subcharacteristics and 2 from 8 software quality characteristics; `mas[Number Of Faults]:=3/138`;
- if in the SRS the measure «Number Of Failures» is absent, then: ma:=ma+1, ft:=ft+1 (counter of missing measures for Fault Tolerance), ru:=ru+1, md:=md+1, anb:=anb+1 (for Analysability), cex:=cex+1 and, respectively, ry:=ry+2, ey:=ey+1, my:=my+2, cy:=cy+1, i.e. the SRS information is insufficient for the calculation 6 from 31 subcharacteristics and 4 from 8 software quality characteristics; `mas[Number Of Failures]:=6/138`;
- if in the SRS the measure «Number Of Resolved Failures» is absent, then: ma:=ma+1, md:=md+1, mfb:=mfb+1, tst:=tst+1 and, respectively, ry:=ry+1, my:=my+3, i.e. the SRS information is insufficient for the calculation 4 from 31 subcharacteristics and 2 from 8 software quality characteristics; `mas[Number Of Resolved Failures]:=4/138`;
- if in the SRS the measure «Number Of Controllability Requirements» is absent, then: cf:=cf+1, ig:=ig+1 and, respectively, sy:=sy+2, i.e. the SRS information is insufficient for the calculation 2 from 31 subcharacteristics and 1 from 8 software quality characteristic; `mas[Number Of Controllability Requirements]:=2/138`;
- if in the SRS the measure «Data Exchangeability» is absent, then: io:=io+1 and, respectively, cy:=cy+1, i.e. the SRS information is insufficient for the calculation 1 from 31 subcharateristic and 1 from 8 software quality characteristic; `mas[Data Exchangeability]:=1/138`;
- … – were similarly formed the rules for the remaining 132 measures;
- if fc=0 and fcr=0 and fa=0 and … (all counters of missing measures for the remaining 28 software quality subcharacteristics are simultaneously equal to 0), then the SRS information is sufficient for calculation of all software quality subcharacteristics, else: the SRS information is insufficient for the calculation of some software quality subcharacteristics (if 0<fc≤4, then: the SRS information is insufficient for the calculation of Functional Completeness subcharacteristic; if fc=4, then the information for Functional Completeness subcharacteristic is absent in the SRS; … – were similarly formed the rules for the remaining 30 subcharacteristics);
- if fy=0 and ry=0 and ey=0 and uy=0 and my=0 and sy=0 and cy=0 and py=0, then the SRS information is sufficient for calculation of all 8 software quality characteristics by the standard ISO 25010, else: 1) the SRS information is insufficient for calculation of some software quality characteristics (if 0<fy≤15, then: the SRS information is insufficient for calculation of Functional Suitability characteristic; if fy=15, then the

information for Functional Suitability characteristic is absent in the SRS; … – were similarly formed the rules for the remaining 7 characteristics); 2) array `mas` should be sorted in descending the values of elements (weights of missing measures); 3) indices of those elements of the sorted array `mas`, which aren't equal 0, should be displayed – as the recommended priority of addition of the missing measures in the SRS.

The *method of forming the logical conclusion about the sufficiency of the SRS information for software quality assessment (by ISO 25010:2011)* is developed on the basis of the production rules of forming the logical conclusion about the sufficiency of the SRS information for software quality assessment. This method consists of next steps:

- to form the set of missing (in the ontology for the concrete software) measures $\{qms_1,\ldots,qms_{(138\text{-}nm)}\}$ (where nm – quantity of measures, which are available in the specification of the concrete software), according to method of evaluating the sufficiency of the SRS information for software quality assessment (by the standard ISO 25010:2011) based on the ontology [10], taken into account the results of the comparative analysis of the base ontology and ontology for quality of the concrete software;
- by searching in width in the forward direction, to search in the subset of production rules $\{pr_1,\ldots,pr_{138}\}$ the rule for each element of the set $\{qms_1,\ldots,qms_{(138\text{-}nm)}\}$, under which the missing in the SRS measures for the evaluation of the subcharacteristics and characteristics are counted;
- according to rules from subset $\{pr_{139},,pr_{140}\}$, to analyze the SRS information on sufficiency for software quality assessment. If the SRS information is insufficient, then: to form the conclusion for which characteristics and subcharacteristics the information is insufficient; to form the sorted (in descending the values of weights) list of missing measures as the recommended priority of their addition in the SRS;
- to evaluate the veracity of software quality assessment based on the available in the SRS information (the veracity should tend to 1) according to the following equations:

$$q_{chr}=fy/15+ry/30+ey/26+uy/49+my/33+sy/23+cy/9+py/18, \quad (1)$$

where q_{chr} – quantity of software quality characteristics, which cannot be calculated on the basis of available in the SRS measures; numbers in the denominators of fractions indicate the quantity of measures for each software quality characteristic according to the standards [5, 6];

$$D_{chr}=(8-q_{chr})/8, \quad (2)$$

where D_{chr} – veracity of software quality assessment based on the available in the SRS measures.

Given the fact that the methods of evaluating the sufficiency of the SRS information for software quality assessment (by the standard ISO 25010:2011) based on the ontology [10] and based on the weighted ontology [11] are the iterative, after forming the conclusion about the insufficiency of the SRS information, the addition(s) of the necessary measures in the SRS will be held. Such addition(s) would result in increased veracity of software quality assessment.

For evaluation of the gain of the veracity of software quality assessment, should first evaluate the veracity of software quality assessment based on the available after addition(s) information in the SRS, by the following equations:

$$q'_{chr}=fy'/15+ry'/30+ey'/26+uy'/49+$$
$$+my'/33+sy'/23+cy'/9+py'/18, \quad (3)$$

where q'_{chr} – quantity of software quality characteristics, which cannot be calculated on the basis of available after addition(s) measures;

$$D'_{chr}=(8-q'_{chr})/8, \quad (4)$$

where D'_{chr} – veracity of software quality assessment based on the available after addition(s) measures.

Then the gain of the veracity of software quality assessment after addition(s) of the necessary measures in the SRS will be calculated by the following equations:

$$\Delta q_{chr}=q_{chr}-q'_{chr}=(fy/15-fy'/15)+(ry/30-ry'/30)+\ldots+$$
$$(py/18-py'/18), \quad (5)$$

where Δq_{chr} – quantity of software quality characteristics, which can be calculated after addition(s) measures in the SRS;

$$\Delta D_{chr}=D'_{chr}-D_{chr}=\Delta q_{chr}/8, \quad (6)$$

where ΔD_{chr} – gain of the veracity of software quality assessment after addition(s) of the measures in the SRS.

IV. Experiments

For experiment the SRS of information system (IS) for the store and warehouse of spare parts for trucks was analyzed. The measures, which are available in this SRS, were identified. The ontology for the quality of this software was developed on the basis of the method of generating and filling the template of ontology for the quality of the concrete software. This ontology is formed by the components for: Functional Suitability, Compatibility, Performance Efficiency, Portability, Usability, Reliability, Security, and Maintainability of the concrete software. For example, the component of ontology for Functional Suitability of IS for the store and warehouse of spare parts for trucks is represented on Fig.1 (the subcharacteristics, for calculation of which lacks some necessary measures in the SRS, are circled).

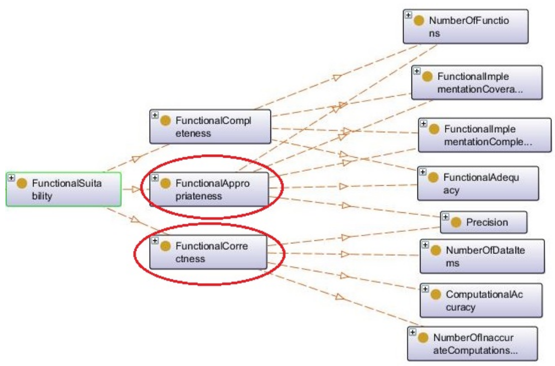

Fig. 1. The component of ontology for Functional Suitability of IS for the store and warehouse of spare parts for trucks

The comparison (Fig.2) of the developed ontology for IS for the store and warehouse of spare parts for trucks with the base ontology for subject domain “Software engineering” (part “Software quality”) provides the conclusion, that in the developed ontology for the quality of the concrete software 6 measures are absent: «Operation Time», «Number Of Faults», «Number Of Failures», «Number Of Resolved Failures», «Number Of Controllability Requirements», «Data Exchangeability». Then the set of missing measures is: $\{qms_1,\ldots,qms_6\}$={Operation Time, Number Of Faults, Number Of Failures, Number Of Resolved Failures, Number Of Controllability Requirements, Data Exchangeability}.

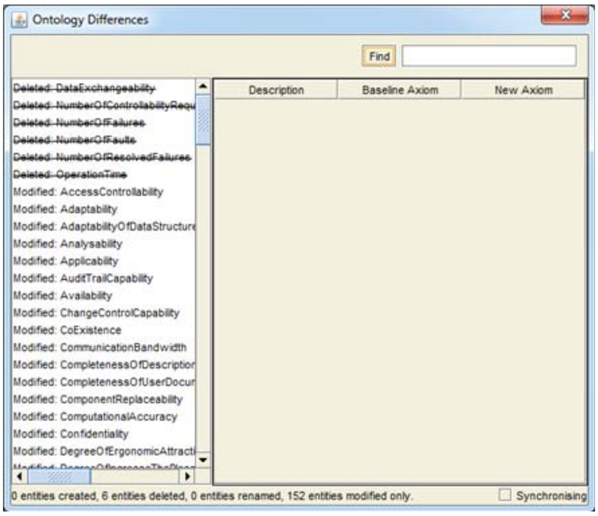

Fig. 2. The comparison of the developed ontology for IS for the store and warehouse of spare parts for trucks with the base ontology for subject domain “Software engineering” (part “Software quality”)

By the method of forming the logical conclusion about the sufficiency of the SRS information for software quality assessment, searching the rule for each element of the set $\{qms_1,\ldots,qms_6\}$ is performed. According to these rules, the missing in the SRS measures for the evaluation of the subcharacteristics and characteristics are counted. The elements of set $\{qms_1,\ldots,qms_6\}$ with the results of the rules for each element of this set are represented in Table I.

According to the penultimate of the above rules, the fact was established, that the available measures in the SRS of IS for the store and warehouse of spare parts for trucks are insufficient for calculation of following subcharacteristics: Functional Correctness, Functional Appropriateness, Maturity, Availability, Fault Tolerance, Recoverability, Time Behaviour, Resource Utilization, Learnability, Operability, Modularity, Analysability, Modifability, Testability, Confidentiality, Integrity, CoExistence, Interoperability, Adaptability, Installability.

According to the 1st part of the last of the above rules, the fact was established, that the available measures in the SRS of IS for the store and warehouse of spare parts for trucks are insufficient for calculation of all 8 software quality characteristics.

Thus, the lack of 6 measures in the SRS led: to the impossibility of calculating the 20 (from 31) subcharactersitics, to the impossibility of calculating all 8 software quality characteristics with high veracity and, respectively, to the impossibility of software quality assessment with high veracity.

TABLE I. LIST OF THE MISSING IN THE SRS MEASURES WITH THE RESULTS OF METHOD OF FORMING THE LOGICAL CONCLUSION ABOUT THE SUFFICIENCY OF THE SRS INFORMATION FOR SOFTWARE QUALITY ASSESSMENT

No	Missing Measures	Results of Method (of Appropriate Production Rules)

1	Operation Time	fcr=1, fa=1, ma=1, av=1, rvb=1, tb=1, ru=1, lb=1, ob=1, md=1, mfb=1, tst=1, cf=1, ig=1, cex=1, io=1, ab=1;
		fy=2, ry=3, ey=2, uy=2, my=3, sy=2, cy=2, py=1;
		`mas[Operation Time]=17/138`

2	Number Of Faults	ma=2, ab=2, ib=1;
		ry=4, py=3;
		`mas[Number Of Faults]=3/138`

3	Number Of Failures	ma=3, ft=1, ru=2, md=2, anb=1, cex=2;
		ry=6, ey=3, my=5, cy=3;
		`mas[Number Of Failures]=6/138`

4	Number Of Resolved Failures	ma=4, md=3, mfb=2, tst=2;
		ry=7, my=8;
		`mas[Number Of Resolved Failures]=4/138`

5	Number Of Controllability Requirements	cf=2, ig=2;
		sy=4;
		`mas[Number Of Controllability Requirements]=2/138`

6	Data Exchangeability	io=2;
		cy=4;
		`mas[Data Exchangeability]=1/138`

After establishing the fact of insufficiency of information of the SRS of IS for the store and warehouse of spare parts for trucks, according to the 2nd and 3rd parts of the last of the above rules: sorting the array `mas` in descending the values of elements (weights of missing measures) was conducted; indices of those elements of the sorted array `mas`, which aren't equal 0, were displayed - Table II.

TABLE II. SORTED LIST OF MISSING IN THE SRS MEASURES IN DESCENDING THE WEIGHTS

No	Missing Measures (Indices of `mas` Elements)	Weights (Values of `mas` Elements)
1	Operation Time	17/138
2	Number Of Failures	6/138
3	Number Of Resolved Failures	4/138
4	Number Of Faults	3/138
5	Number Of Controllability Requirements	2/138
6	Data Exchangeability	1/138

Table II represented the recommended priority of the addition of missing measures in the SRS of IS for the store and warehouse of spare parts for trucks.

Next, the evaluation of the veracity of software quality assessment based on the available in the SRS information is done (according to the equations 1 and 2):

$$q_{chrIS}=2/15+7/30+3/26+2/49+8/33+4/23+4/9+3/18=1.54, \quad (7)$$

$$D_{chrIS}=(8-1.54)/8=0.81 \quad (8)$$

So, for the analyzed SRS the conclusion about insufficient data for software quality assessment was formed, and the software quality can be evaluated with 81% veracity with available in the SRS measures.

Because the proposed methods of evaluating the sufficiency of the SRS information for software quality assessment (by the standard ISO 25010:2011) based on the ontology [10] and based on the weighted ontology [11] are iterative, and there are subcharacteristics and characteristics, for calculation of which the measures of SRS are insufficient, then the addition of the necessary measures in the SRS was held. After addition of the SRS of IS for the store and warehouse of spare parts for trucks, the ontology (version 2) for the quality of this software was re-developed. The comparison of the re-developed ontology (version 2) for IS for the store and warehouse of spare parts for trucks with the base ontology for subject domain "Software engineering" (part "Software quality") provides the conclusion, that 3 measures were added in the SRS: «Operation Time» (1st in the sorted list (Table II)), «Number Of Faults» (4th in the sorted list), «Data Exchangeability» (6th in the sorted list). Then the set of missing measures is: $\{qms'_1,\ldots,qms'_3\}$={ Number Of Failures, Number Of Resolved Failures, Number Of Controllability Requirements}.

By the method of forming the logical conclusion about the sufficiency of the SRS information for software quality assessment, searching the rule for each element of the set $\{qms'_1,\ldots,qms'_3\}$ is performed. The elements of set $\{qms'_1,\ldots,qms'_3\}$ with the results of the rules for each element of this set are represented in Table III.

TABLE III. LIST OF THE MISSING (AFTER ADDITION) IN THE SRS MEASURES WITH THE RESULTS OF METHOD OF FORMING THE LOGICAL CONCLUSION ABOUT THE SUFFICIENCY OF THE SRS INFORMATION FOR SOFTWARE QUALITY ASSESSMENT

No	Missing Measures	Results of Method (of Appropriate Production Rules)
1	Number Of Failures	ma′=1, ft′=1, ru′=1, md′=1, anb′=1, cex′=1; ry′=2, ey′=1, my′=2, cy′=1; `mas[Number Of Failures]=6/138`
4	Number Of Resolved Failures	ma′=2, md′=2, mfb′=1, tst′=1; ry′=3, my′=5; `mas[Number Of Resolved Failures]=4/138`
5	Number Of Controllability Requirements	cf′=1, ig′=1; sy′=2; `mas[Number Of Controllability Requirements]=2/138`

According to the penultimate of the above rules, the fact was established, that the available measures in the SRS of IS for the store and warehouse of spare parts for trucks are still insufficient for calculation of 10 subcharacteristics (with indicating these subcharacteristics), but addition 3 measures in the SRS made possible the calculation of: Functional correctness, Functional appropriateness, Availability, Recoverability, Time behaviour, Learnability, Operability, Interoperability, Adaptability, Installability of this software.

According to the 1st part of the last of the above rules, the fact was established, that the available measures in the SRS of IS for the store and warehouse of spare parts for trucks are still insufficient for calculation of 5 (from 8) software quality characteristics.

After establishing the fact of insufficiency of information of the SRS of IS for the store and warehouse of spare parts for trucks, according to the 2nd and 3rd parts of the last of the above rules: sorting the array `mas` in descending the values of elements (weights of missing measures) was conducted; indices of those elements of the sorted array `mas`, which aren't equal 0, were displayed - Table IV.

TABLE IV. SORTED LIST OF MISSING (AFTER ADDITION) IN THE SRS MEASURES IN DESCENDING THE WEIGHTS

No	Missing Measures (Indices of `mas` Elements)	Weights (Values of `mas` Elements)
1	Number Of Failures	6/138
2	Number Of Resolved Failures	4/138
3	Number Of Controllability Requirements	2/138

Next, the evaluation of the veracity of software quality assessment based on the available in the SRS information is done (according to the equations 3 and 4):

$$q'_{chr}=0/15+3/30+1/26+0/49+5/33+2/23+1/9+0/18=0.49, \quad (9)$$

$$D'_{chr}=(8-0.49)/8=0.94 \quad (10)$$

The process of addition the necessary measures in the SRS is iterative. It can be continued until all quality characteristics and subcharacteristics will be possible to calculate or until the conclusion will be formed, that data for software quality assessment are insufficient. The customer of developed IS for the store and warehouse of spare parts for trucks has decided that further complement of the specification is economically inexpedient therefore the conclusion about insufficient data for determination of the software quality was formed. The software quality can be evaluated with 94% veracity with available (after addition) in the SRS measures.

The gain of the veracity of software quality assessment after addition of the necessary measures in the SRS was calculated by the equations 5 and 6:

$$\Delta q_{chr}=q_{chr}-q'_{chr}=1.54-0.49=1.05 \quad (11)$$

$$\Delta D_{chr}=D'_{chr}-D_{chr}0.94-0.81=0.13, \quad (12)$$

$$\Delta D_{chr}= \Delta q_{chr}/8=1.05/8=0.13 \quad (13)$$

The gain of the veracity of software quality assessment is illustrated by Table V.

TABLE V. GAIN OF THE VERACITY OF SOFTWARE QUALITY ASSESSMENT

Evaluation of the Veracity of Software Quality Assessment Before Addition of the SRS	Evaluation of the Veracity of Software Quality Assessment After Addition of the SRS	Gain of the Veracity
81%	94%	13%

Conclusions

This paper proves: the need of deepening of the analysis of the software requirements specification, the dependence of quality and success of implementation of software project on the SRS, the actuality and importance of the evaluation of the sufficiency of the SRS information.

In this paper the actuality and necessary of the development of methodology of evaluating the sufficiency of the SRS information for software quality assessment were also proved. For completion of such methodology, the method of generating and filling the template of ontology for the quality of the concrete software, and method of forming the logical conclusion about the sufficiency of the SRS information for software quality assessment (by ISO 25010:2011) were developed.

The first time proposed method of generating and filling the template of ontology for the quality of the concrete software leaves in the relevant ontology only measures, which are available in the SRS for the concrete software. It provides forming the ontology for the assessment of quality of the concrete software by ISO 25010:2011.

The method of forming the logical conclusion about the sufficiency of the SRS information for software quality assessment by ISO 25010:2011 was first time developed on the basis of the proposed production rules of forming the logical conclusion about the sufficiency of the SRS information for software quality assessment. This method (considering the fact that the developed ontological methods are iterative):

- provides the conclusion about the sufficiency or insufficiency of the SRS information for software quality assessment by ISO 25010:2011;
- if the SRS information is insufficient: provides the conclusion for which characteristics and subcharacteristics the information is insufficient; creates the sorted list of missing measures in descending its weights – as the recommended priority of their addition in the SRS; evaluates the veracity of software quality assessment based on the available in the SRS measures.

Use of the methodology of evaluating the sufficiency of the SRS information for software quality assessment provides to increase the veracity of software quality assessment by 13% for the information system for the store and warehouse of spare parts for trucks due to iterative nature of methods and possibility of the addition of the necessary measures in the software requirements specification.

The author has following perspective for future research: development of information technology of evaluating the sufficiency of the SRS information for software quality assessment.

References

[1] CHAOS Manifesto: Think big, act small. The Standish Group International: CHAOS Knowledge Center, 2013.

[2] S. Hastie, and S. Wojewoda, “Standish Group 2015 Chaos Report - Q&A with Jennifer Lynch” [Electronic resource] – Access mode: http://www.infoq.com/articles/standish-chaos-2015.

[3] S.McConnell, Code complete. Microsoft Press, 2013.

[4] T. Ishimatsu, N. Levenson, J. Thomas, C. Fleming, M. Katahira, Y. Miyamoto, and R. Ujiie, “Hazard analysis of complex spacecraft using systems-theoretic process analysis”, Journal of Spacecraft and Rockets, vol. 51, no.2, 2014, pp. 509-522

[5] ISO/IEC 25010:2011. Systems and software engineering. Systems and software Quality Requirements and Evaluation (SQuaRE). System and software quality models, 2011.

[6] ISO 25023:2016. Systems and software engineering. Systems and software Quality Requirements and Evaluation (SQuaRE). Measurement of system and software product quality, 2016.

[7] D. Maevskiy, and Y. Kozina, “Where and When Is Formed of Software Quality?”, Electrical and Computer Systems, vol. 18, 2016, pp. 55-59 (in Russian).

[8] A. Fatwanto, “Software requirements specification analysis using natural language processing technique”, in: International Conference on Quality in Research, 2013, pp.105-110.

[9] T. Rehman, M. N. A. Khan, and N. Riaz, “Analysis of requirement engineering processes, tools/techniques and methodologies”, I.J. Information Technology and Computer Science, 2013, pp. 40-48.

[10] T. Hovorushchenko, and O. Pomorova, “Ontological Approach to the Assessment of Information Sufficiency for Software Quality Determination”, CEUR-WS, vol.1614, 2016, pp. 332-348.

[11] T. Hovorushchenko, O. Ivanov, and O. Pavlova, “Method of Evaluation of Sufficiency of Information for the Software Quality Assessment Based on Weighted Ontology”, Transactions of Khmelnitsky National University, vol. 6, 2016, pp. 191-196 (in Ukrainian)