DOI: https://doi.org/10.31341/jios.42.1.4 UDC 004.415.5
Open Access Original Scientific Paper

Methodology of Evaluating the Sufficiency of Information for Software Quality Assessment According to ISO 25010

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

Abstract

The research is devoted to the development of the formalized and ontological models of the software quality according to ISO 25010. These models provide the possibility of the formalization of the software quality assessment according to ISO 25010. This standard would benefit from a formal description. The paper proves that information sufficiency is a critical aspect of software quality assessment. The methods and system of evaluating the sufficiency of the information for software quality assessment according to ISO 25010 are developed. The developed methods and tools provide the increasing the veracity of software quality assessment. The conducted experiments confirm that the developed methodology of evaluating the sufficiency of information for software quality assessment increased the veracity of the software quality assessment in 12% for the automated system for large-format photo print.
Keywords: software, software quality, software requirements specification (SRS), sufficiency of information, ISO 25010.

1. Introduction

Today almost all areas of human activity are associated with computer and applied information systems, which are based on the software. The key factor of the effectively using of software and one of the main requirements of user and stakeholders to modern software is high quality. Software quality is the degree of compliance of software to customers' needs [1], [2].

Analytical studies and reviews on software [3], [4] show that now the crisis in the field of assessment and assurance of the software quality is ongoing.

Exactly the software requirements determine the necessary characteristics of software quality and impact on methods of quantitative evaluation of software quality. Software projects with incomplete requirements and specifications cannot have the successful implementation.

Many software bugs arise at the requirements formulation stage (10-23% of all software bugs) [5]. The sooner the defect will be discovered, the cheaper it is to fix. As Figure 1 shows, defects discovered after the product is released can cost nearly 100 times more to fix than defects found during the requirements process [6].

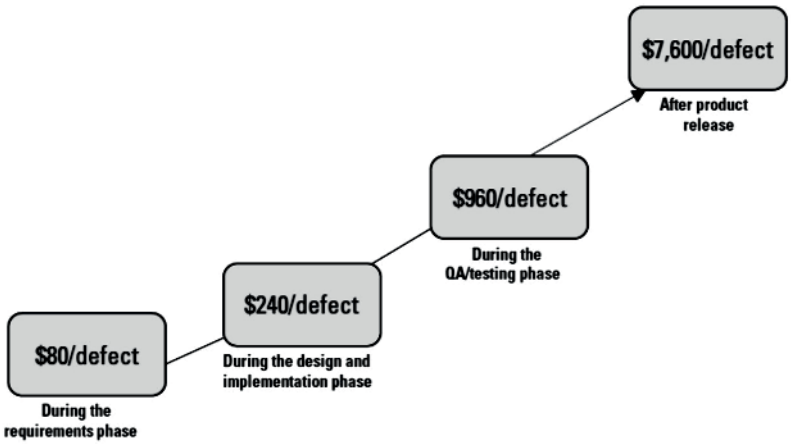

Figure 1. The cost of correcting defects increases dramatically through-out the development process [6]

During the requirements formulation the information loss are occurring through incomplete and differences of the understanding of the needs and context of specification information These losses are particularly noticeable for software projects at the intersection of subject domains (e.g. software for medicine) when you need to consider the software development standards and standards of the subject domain for which software is developed. This quantity of standards is difficult to implement, and the degree of consideration of recommendations of these standards is even more difficult to verify.

So now the actual and important task is the software requirements specification (SRS) analysis, possibility to "cut off" the software projects with the incomplete (with insufficient information) specification. *The sufficiency of information* is the rational information saturation that eliminates information incompleteness (lack of necessary information).

Today the evaluation of the information for software quality assessment is conducted only at the late stages of the lifecycle (at the implementation stage or QA/testing stage) [7]. The research of the modern methods and tools of the SRS analysis [8], [9], [10] showed that they intended to monitoring the implementation of the requirements and don't evaluate the sufficiency of the SRS information for future software quality assessment. But the SRS have all necessary information, i.e. the information sufficiency for future software quality assessment can be evaluated on the basis of the SRS and, in the absence of important information, the necessary information can be added in the SRS.

For improving the veracity of software quality assessment the knowledge of experts (who already have experience of quality assessment for different types of software) are significant. For example, the knowledge about the mutual influence and correlation of software quality subcharacteristics (by measures) are valuable. If the measures (for which there is a correlation) will be missed, then the accuracy and veracity of the software quality assessments will be fall. The information about the

software quality characteristics, subcharacteristics and measures (e.g. about the correlation of the subcharacteristics by the measures) is conveniently presented as ontologies, which provides the reflection of the causal relationships between concepts. The advantages of using the ontologies are: the systematic approach to the study of the subject domain; the possibility of the holistic filing the known information of subject domain; the identification of the overlaps and gaps in knowledge on the basis of visualization of the missing logical connections; possibility of access, understanding and analyzing the information by intelligent and non-intelligent agents (which is very important in the present era of transition to semantic Web where resources should be clear not only for humans but also for agents).

The idea of using ontologies in the field of software engineering is not new. Thus, the authors of [11] investigate the creation of an ontological infrastructure that aims to be a single coherent underpinning for all ISO/IEC JTC1's SC7 standards. The authors of [12] proposed the using an Ontology Pattern Language as the main component of ontological infrastructure to establish a common conceptualization for underpinning all ISO/IEC JTC1/SC7 standards. The purpose of [13] is to create domain ontology for ISO/IEC 24744, which will serve as semantic reference in order to assist for a better interoperability between the different users of the standard (human, software or machine). The paper [14] is devoted to using the domain ontology in the software analysis and reengineering tools. So, ontologies in the software engineering domain used for the software development process, but not for the software quality assessment process.

Consequently, the lack of methodology of evaluating the sufficiency of information for software quality assessment creates the *scientific problem.* One of the ways of this problem solution is the development of models, methods, and tools for evaluating the sufficiency of the information in the SRS for further software quality assessment. The *aim of the research* is the solving problem of the development of the methodology of evaluating the sufficiency of the SRS information for software quality assessment (according to ISO 25010) by using ontological models, which reduces the complexity of their development and adaptation to the features of the subject domain.

The rest of the paper is structured as follows: In section 2, formalized and ontological models of the software quality (by ISO 25010) are presented. Next, the methods (section 3) and system (section 4) of evaluating the sufficiency of the SRS information for software quality assessment (by ISO 25010) are provided. The experimental results and discussion are given in section 5. Finally, in the section 6, the conclusions are presented.

2. Formalized and Ontological Models of the Software Quality According to ISO 25010:2011

Currently, the software quality evaluation by standard ISO 25010:2011 [1] is as follows – the software quality is evaluated on the basis of the characteristics, the characteristics are evaluated on the basis of subcharacteristics, and the

subcharacteristics are evaluated on the basis of measures, which are specified in ISO 25023:2016 [15]. The basic idea of SQuaRE model (ISO 25010:2011) [1] is the exactly comprehensive quality evaluation, with taking into account all the quality measures, subcharacteristics and characteristics.

Today, most standards for software quality assessment (including the ISO 25010 [1] and ISO 25023 [15]) are written in text form (without formalization). Respectively the mechanism of the verification of the degree of the standard requirements implementation during the software development is absent.

Therefore, for the evaluation of the mutual influences of the software quality characteristics, subcharacteristics and measures it's necessary to develop the formalized model of the software quality, which will consider the mutual influences of the characteristics, subcharacteristics and measures.

The software quality (Q), according to ISO 25010 [1], is the function of 8 software quality characteristics, i.e. $Q = f(qch_1,...,qch_8)$. The set of the software quality characteristics is $QCH = \{qch_1,...,qch_8\} = \{Fs, Pe, Ub, Rb, Cb, Scr, Mb, Pb\}$, where: Fs – Functional Suitability, Pe – Performance Efficiency, Ub – Usability, Rb – Reliability, Cb – Compatibility, Scr – Security, Mb – Maintainability, Pb – Portability, which are the values from the certain range. Then the software quality has the form:

$$Q = f(Fs, Pe, Ub, Rb, Cb, Scr, Mb, Pb). \quad (1)$$

Each software quality characteristic is the function of several subcharacteristics:

$$Fs = f_1(qsch_1, qsch_2, qsch_3) = f_1(FCom, FCor, FAppr), \quad (2)$$

$$Pe = f_2(qsch_4, qsch_5, qsch_6) = f_2(Tb, Ru, Cc), \quad (3)$$

$$Ub = f_3(qsch_7,...,qsch_{12}) = f_3(Ar, Lb, Ob, Uep, Uia, Ab), \quad (4)$$

$$Rb = f_4(qsch_{13},...,qsch_{16}) = f_4(Mat, Avb, Ft, Rcv), \quad (5)$$

$$Cb = f_5(qsch_{17}, qsch_{18}) = f_5(Ce, Ib), \quad (6)$$

$$Scr = f_6(qsch_{19},...,qsch_{23}) = f_6(Conf, Int, Nr, Acb, Auth), \quad (7)$$

$$Mb = f_7(qsch_{24},...,qsch_{28}) = f_7(Mod, Rub, Anb, Mdfb, Tsb), \quad (8)$$

$$Pb = f_8(qsch_{29}, qsch_{30}, qsch_{31}) = f_8(Adb, Inb, Rpb), \quad (9)$$

where $FCom$ – Functional Completeness, $FCor$ – Functional Correctness, $FAppr$ – Functional Appropriateness; Tb – Time Behavior, Ru – Resource Utilization, Cc – Capacity; Ar – Appropriateness Recognizability, Lb – Learnability, Ob – Operability, Uep – User Error Protection, Uia – User Interface Aesthetics, Ab – Accessibility; Mat – Maturity, Avb – Availability, Ft – Fault Tolerance, Rcv – Recoverability; Ce – Co-existence, Ib – Interoperability; $Conf$ – Confidentiality,

Int – Integrity, Nr – Non-repudiation, Acb – Accountability, $Auth$ – Authenticity; Mod – Modularity, Rub – Reusability, Anb – Analyzability, $Mdfb$ – Modifiability, Tsb – Testability; Adb – Adaptability, Inb – Installability, Rpb – Replaceability.

Thus, the set of the software quality subcharacteristics has the form: $QSCH = \{qsch_1,...,qsch_{31}\} = \{FCom, FCor, FAppr, Tb, Ru, Cc, Ar, Lb, Ob, Uep, Uia, Ab, Mat, Avb, Ft, Rcv, Ce, Ib, Conf, Int, Nr, Acb, Auth, Mod, Rub, Anb, Mdfb, Tsb, Adb, Inb, Rpb\}$

So, *formalized model of software quality according to ISO 25010* has the form:

$$Q = f(Fs, Pe, Ub, Rb, Cb, Scr, Mb, Pb) = f\begin{pmatrix} f_1(FCom, FCor, FAppr), \\ f_2(Tb, Ru, Cc), \\ f_3(Ar, Lb, Ob, Uep, Uia, Ab), \\ f_4(Mat, Avb, Ft, Rcv), \\ f_5(Ce, Ib), \\ f_6(Conf, Int, Nr, Acb, Auth), \\ f_7(Mod, Rub, Anb, Mdfb, Tsb), \\ f_8(Adb, Inb, Rpb) \end{pmatrix}. \quad (10)$$

In turn, each software quality subcharacteristic is the function of several measures, which are specified in ISO 25023 [15]. The set of software quality measures, according to [15], has the form $QMS = \{qms_1,...,qms_{138}\}$, because the software quality subcharacteristics and characteristics depend on 203 measures, but only on 138 different measures (there are measures, which affect to more than one subcharacteristic and characteristic).

There are the examples of the models for Functional Completeness, Functional Correctness, Functional Appropriateness:

$$FCom = \varphi_1(qms_1,...,qms_4) = \varphi_1(Nof, Ficn, Faq, Fic), \quad (11)$$

$$FCor = \varphi_2(qms_5,...,qms_9) = \varphi_2(Ot, Nic, Ndi, Ca, Pc), \quad (12)$$

$$FAppr = \varphi_3(qms_5, qms_1, qms_2, qms_3, qms_4) = \varphi_3(Ot, Nof, Ficn, Faq, Fic, Pc), \quad (13)$$

where Nof – Number Of Functions, $Ficn$ – Functional Implementation Completeness, Faq – Functional Adequacy, Fic – Functional Implementation Coverage; Ot – Operation Time, Nic – Number Of Inaccurate Computations Encountered By Users, Ndi – Number Of Data Items, Ca – Computational Accuracy, Pc – Precision.

Similarly, the models for remaining 28 software quality subcharacteristics were developed.

The developed models show, that there is the correlation of subcharacteristics and characteristics by measures. These correlations influence to significance and weights of the software quality measures. If the measures, which influences to few

subcharacteristics, are inaccurate or absent, then the simultaneous use of these subcharacteristics significantly affects to the veracity of software quality assessments (because the possibility of calculation of just a few characteristics will disappear). Therefore, it's necessary to identify the joint measures for software quality subcharacteristics and characteristics and to determine the significance (probability) of the measures for improving the veracity of the software quality assessment. The knowledge of experienced professionals about the mutual influences and correlation of subcharacteristics and characteristics by measures are valuable in identifying the joint measures, so they should be stored and used. The ontologies were selected for this knowledge reflection and accumulation.

The ontological model of software quality according to ISO 25010:2011 has the form: $O_Q =< X_Q, RX_Q, F_Q >$, where X_Q – finite set of the software quality characteristics, subcharacteristics and measures, RX_Q – finite set of relationships between concepts, F_Q – finite set of interpretation functions for the software quality characteristics, subcharacteristics and measures.

Considering the formalized model of software quality according to ISO 25010, the set of the software quality characteristics, subcharacteristics and measures is:

$$X_Q = \{QCH, QSCH, QMS\} = \{x_{Q_1}, ..., x_{Q_{177}}\}, \quad (14)$$

where: $\{x_{Q_1}, ..., x_{Q_8}\} \in QCH$, $\{x_{Q_9}, ..., x_{Q_{39}}\} \in QSCH$, $\{x_{Q_{40}}, ..., x_{Q_{177}}\} \in QMS$.

The set of relationships between concepts RX_Q consists of relationship «depends on», i.e. $RX_Q = \{"depends\ \ on"\}$. The set F_Q of interpretation functions for the software quality characteristics, subcharacteristics and measures, consists of function for quality depending on the characteristics, functions for characteristics depending on the subcharacteristics and functions for subcharacteristics depending on the measures, i.e. $F_Q = \{f_{Q_1}, ... f_{Q_{40}}\} = \{f(), f_1(), ..., f_8(), \varphi_1(), ... \varphi_{31}()\}$.

Thus *the base (universal) ontological model for the subject domain "Software engineering" (part "Software quality" according to ISO 25010)* has the form:

$$\begin{aligned} O_Q = \{&qch_1, ..., qch_8, qsch_1, ... qsch_{31}, qms_1, ... qms_{138}, \\ &"depends\ \ on", f(), f_1(), ..., f_8(), \varphi_1(), ... \varphi_{31}()\} \end{aligned}. \quad (15)$$

And *the ontological model of the concrete software quality according to ISO 25010* has the form:

$$\begin{aligned} O_{Q_{real}} = \{&qch_1, ..., qch_{nch}, qsch_1, ... qsch_{nsch}, qms_1, ... qms_{nm}, \\ &"depends\ \ on", f(), f_1(), ..., f_8(), \varphi_1(), ... \varphi_{31}()\} \end{aligned}, \quad (16)$$

where nch ($nch \le 8$) – quantity of software quality characteristics, which can be calculated on the basis of the available measures in the SRS of concrete software,

$nsch$ ($nsch \le 31$) – quantity of software quality subcharacteristics, which can be calculated on the basis of the available measures in the SRS of concrete software, nm ($nm \le 138$) – quantity of quality measures, which are available in the SRS of concrete software.

Considering the SRS structure according to ISO 29148 [16], the SRS can be represented in the following formalized form (in terms of the availability of software quality measures in it):

$$SRS =< R1, R2, R3, R4, R5 >, \tag{17}$$

where $R1$ – set of quality measures of section 1 of SRS, $R2$ – set of measures of section 2 of SRS, $R3$ – set of measures of section 3 of SRS, $R4$ – set of measures of section 4 of SRS, $R5$ – set of measures of section 5 of SRS.

According to ISO 29148, the section 3 "Specific requirements" of the SRS has the subsection "Software system measures" that can contain the values of all 138 software quality measures. So $QMS \in R3$.

Thus *the formalized model of the SRS (in terms of the availability of software quality measures)* has the form:

$$SRS_m = \langle \varnothing, \varnothing, \{qms_1, ...qms_{138}\}, \varnothing, \varnothing \rangle. \tag{18}$$

The ontological model of the SRS in terms of the availability of software quality measures has the form: $O_{SRS} =< X_{SRS}, RX_{SRS} >$, where X_{SRS} – finite set of software quality measures in the SRS, RX_{SRS} – finite set of relationships between concepts.

Considering the formalized model of the SRS (in terms of the availability of software quality measures), the set of software quality measures $X_{SRS} = \{SRS, QMS\} = \{x_{SRS_1}, ..., x_{SRS_{143}}\}$, where $\{x_{SRS_1}, ..., x_{SRS_5}\} = \{R_1, ...R_5\}$ $\{x_{SRS_6}, ..., x_{SRS_{143}}\} = \{qms_1, ...qms_{138}\}$. The set of relationships between concepts consists of relationship «contained in», i.e. $RX_{SRS} = \{"contained\ \ in"\}$.

Thus *the base (universal) ontological model of the SRS (in terms of the availability of software quality measures)* has the form:

$$O_{SRS} = \{R1, ..., R5, qms_1, ...qms_{138}, "contained\ \ in"\}. \tag{19}$$

And *the ontological model of the SRS for concrete software (in terms of the availability of software quality measures)* has the form:

$$O_{SRS_{real}} = \{R1, ..., R5, qms_1, ...qms_{nm}, "contained\ \ in"\}, \tag{20}$$

where nm ($nm \le 138$) – quantity of quality measures, which are available in the SRS of concrete software.

The formalized and ontological models of software quality (according to ISO 25010) are based on the ontologies. They take into account the basic idea of standard – comprehensive quality assessment with considering all characteristics,

subcharacteristics and measures. These models provide the formalization of the software quality assessment (by ISO 25010). The formalized and ontological models of the SRS examine the SRS in terms of availability of the software quality measures. They provide the formalization of the SRS in the terms of software quality assessment.

The developed models are the basis for the development of the methods of evaluating the sufficiency of the SRS information for software quality assessment according to ISO 25010.

3. Methods of Evaluating the Sufficiency of the SRS Information for Software Quality Assessment According to ISO 25010:2011

The base ontology for the subject domain «Software engineering» is the realization of the base (universal) ontological model for the subject domain "Software engineering" (part "Software quality" according to ISO 25010), which is represented by equation (15) of section 2. This ontology was developed in [17]. In this ontology, there is information about software quality characteristics, subcharacteristics and measures.

For the concrete software the development of other ontology is supposed. This ontology has only measures, which are available in the concrete SRS. So this ontology is the realization of the ontological model of the concrete software quality according to ISO 25010, which is represented by equation (16) of section 2.

The comparative analysis of the ontology for the concrete software and the base (universal) ontology provides the identification of existence and quantity of the measures, which are missing in the concrete SRS. Therefore for the evaluating the sufficiency of the SRS information for software quality assessment the methods on the basis of the comparative analysis of ontologies were developed.

The method of evaluating the sufficiency of the SRS information for software quality assessment (by the standard ISO 25010:2011) based on the ontology consists of the next stages:

1) analysis of the SRS of the concrete software project for the presence of measures, which are necessary for the software quality characteristics and subharacteristics evaluation, i.e. for the presence of the elements of set $X_{SRS} = \{SRS, QMS\} = \{x_{SRS_1}, ..., x_{SRS_{143}}\}$;

2) generating and filling the template of ontology for the quality of the concrete software, i.e. generating and filling the template of ontology $O_{SRS_{real}} = \{R1, ..., R5, qms_1, ...qms_{nm}, "contained\ \ in"\}$; this ontology is the realization of the ontological model of the concrete software quality according to ISO 25010, which is represented by equation (16) of section 2.

3) comparison of the developed ontology with the base ontology for the subject domain "Software Engineering" (part "Software quality"), i.e. comparison the set of measures $\{qms_1, ..., qms_{nm}\}$ from the ontology $O_{SRS_{real}}$ of the SRS for

concrete software with the appropriate set $\{qms_1,...,qms_{138}\}$ from the base ontology O_Q for the subject domain "Software Engineering" (part "Software quality") ;

4) identification of measures, which are absent in the ontology for the concrete software, i.e. forming the set $\{qms_1,...,qms_{(138-nm)}\} = \{qms_1,...,qms_{138}\} \setminus \{qms_1,...,qms_{nm}\}$, where $\{qms_1,...,qms_{138}\} \in O_Q$, $\{qms_1,...,qms_{nm}\} \in O_{SRS_{real}}$ (if the set is not empty, then information of the SRS is insufficient for calculation of the software quality characteristics and subcharacteristics – the more elements are in this set, the smaller degree of sufficiency of the SRS information is);

5) identification (on the basis of the comparative analysis of the developed ontologies) of quality characteristics and subcharacteristics, that cannot be calculated on the basis of the existing measures;

6) the presence of subcharacteristics and characteristics, which cannot be calculated on the basis of the available in the SRS measures, indicates the insufficiency of the SRS information for veracity assessment of the software quality, i.e. indicates the need to complement of this SRS by the necessary measures;

7) repeating the steps 1-6 until all quality characteristics and subcharacteristics will be possible to calculate or until the conclusion will be formed, that the SRS information is insufficient for software quality assessment.

The final conclusion about insufficient information for the software quality assessment will be taken in the event that the costs of the SRS completion are greater than expected effect on quality assessment.

One of the basic properties of the base ontology for the subject domain "Software Engineering" (part "Software quality") is an illustration of the presence of cross-correlation of characteristics and subcharacteristics by the measures. Because the condition of the mitigation of influence of cross-correlation of characteristics and subcharacteristics by the measures is important, then should focus on those measures, which simultaneously affect to several characteristics and subcharacteristics.

The method of evaluating the weights of software quality measures consists of the next stages:

1) identifying the joint measures for software quality characteristics on the basis of the developed base ontology for subject domain “Software engineering” (part “Software quality”);

2) identifying the joint measures for software quality subcharacteristics on the basis of the developed base ontology for subject domain “Software engineering” (part “Software quality”);

3) calculating the weights of the software quality measures on the basis of the subcharacteristics quantity, which depend on these measures:

3.1) calculating quantity of subcharacteristics $k_{sch_{m_h}}$, which depend on h-th joint measure;

3.2) calculating the weights by the following equation:

$$\omega_{m_h} = \frac{k_{sch_{m_h}}}{k_m}, \quad (21)$$

where k_m – total quantity of measures (now it is 138 different measures, i.e. $k_m = 138$). Obviously, the numerator of the weight of each measure indicates the number of the software quality subcharacteristics that cannot be calculated without this measure.

The weights of software quality measures were calculated according to the proposed method and were represented in [18].

In evaluating the software quality according to ISO 25010:2011 it's important to satisfy the availability in the SRS of those measures, which have larger weights, with the purpose of providing the appropriate level of assessment veracity.

The weighted ontology of the subject domain "Software Engineering" (part "Software quality") will be called the ontology, in which the quality measures have the weights with the purpose of recommendations about the further satisfaction of these measures in the SRS.

The method of evaluating the sufficiency of the SRS information for software quality assessment (according to ISO 25010) based on the weighted ontology consists of the next stages:

1) development of the weighted base ontology for the subject domain "Software Engineering" (part "Software quality") – the part of the weighted base ontology for Functional Suitability is represented on Figure 2, the parts of the weighted base ontology for other quality characteristics are similarly developed;

2) execution of the stages 1-2 of the above-developed method of evaluating the sufficiency of the SRS information for software quality assessment (by the standard ISO 25010:2011) based on the ontology;

3) comparing the developed ontology for concrete software with the weighted base ontology of the subject domain "Software Engineering" (part "Software quality "), i.e. comparing the set of measures $\{qms_1,...,qms_{nm}\}$ from the ontology of the SRS for concrete software with the appropriate set $\{qms_1,...,qms_{138}\}$ from the weighted base ontology for the subject domain "Software Engineering" (part "Software quality");

4) identifying the measures, which are absent in the ontology for concrete software, i.e. forming the set $\{qms_1,...,qms_{(138-nm)}\} = \{qms_1,...,qms_{138}\} \setminus \{qms_1,...,qms_{nm}\}$; sorting of the missing measures in descending the values of weights; herewith the numerator of the weight of each missing measure indicates the number of software subcharacteristics that cannot be calculated without this measure;

5) identifying the characteristics and subcharacteristics, which cannot be calculated on the basis of the available (in the SRS) measures;

6) making the decision about the need the addition of measures in the SRS, if there are characteristics and subcharacteristics, whose values cannot be calculated

based on the available measures; herewith the measures with larger weights (the first in the sorted list of missing measures) should be added in the SRS first of all;

7) repeating the steps 2-6 until all quality characteristics and subcharacteristics will be possible to calculate or until the conclusion will be formed, that the SRS information is insufficient for software quality assessment.

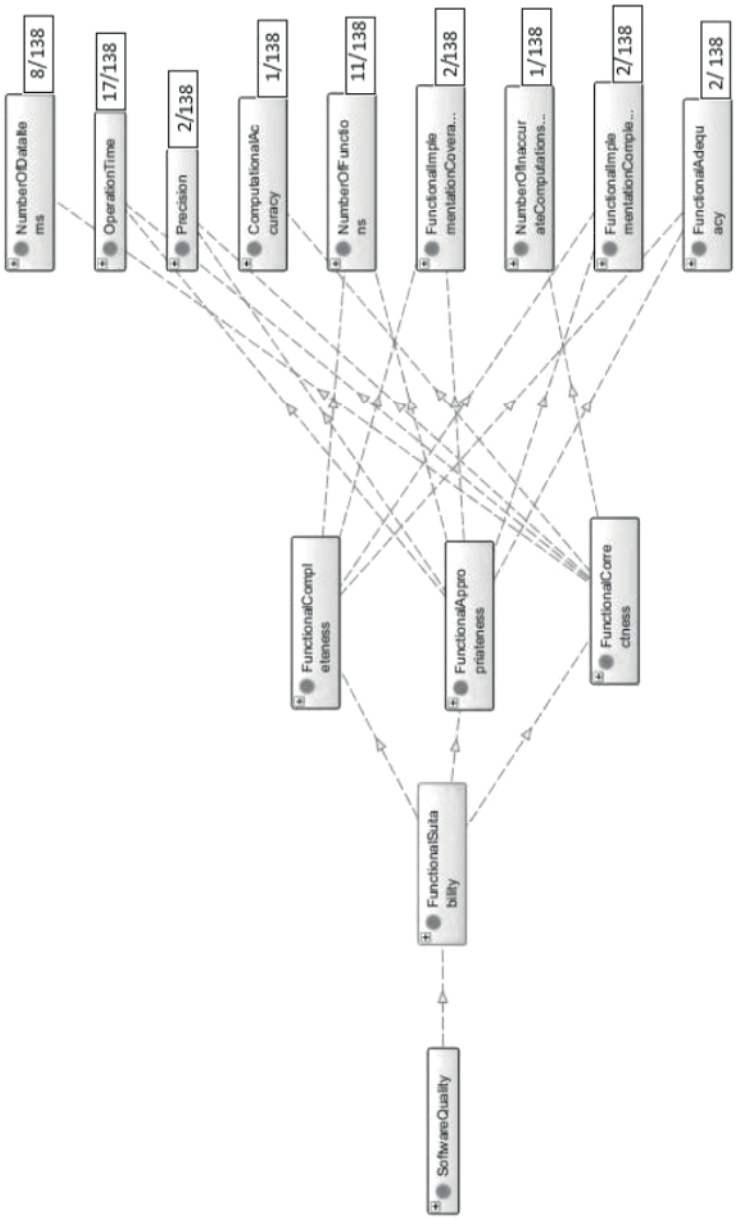

Figure 2. The part of the weighted base ontology of the subject domain "Software Engineering" (part "Software quality") for Functional Suitability

One of the stages of the methods of evaluating the sufficiency of the SRS information for software quality assessment (according to ISO 25010) based on the ontology and the weighted ontology is the generating and filling the template of ontology for the quality of the concrete software. Consequently it is necessary the method, which provides the ontology for the quality of the concrete software that has only measures, which are available in the concrete SRS.

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

1) to open the base ontology for subject domain "Software engineering" (part "Software quality") in Protégé 4.2;

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);

3) to save these changes, thus creating the ontology for the quality of the concrete software.

For the development of the 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 $PR = \{pr_1,..,pr_{140}\}$) should be developed on the basis of the base and the weighted base ontologies for subject domain "Software engineering" (part "Software quality"):

1) 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 Modifiability), $tst := tst + 1$ (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 characteristic 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 characteristics; `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")) is written);

2)-138) – were similarly formed the rules for the remaining 137 measures;

139) 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);

140) if fy=0 and ry=0 and ey=0 and uy=0 and my=0 and sy=0 and cy=0 and py=0 (all counters of missing measures for all 8 software quality characteristics are simultaneously equal to 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 the next stages:

1) to form the set of missing (in the ontology for the concrete software) measures $\{qms_1,...,qms_{(138-nm)}\}$, 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, considering the results of the comparative analysis of the base ontology and ontology for quality of the concrete software;

2) by searching in width in the forward direction, to search in the subset of production rules $\{pr_1,...,pr_{138}\}$ the rule for each element of the set $\{qms_1,...,qms_{(138-nm)}\}$, under which the missing in the SRS measures for the evaluation of the subcharacteristics and characteristics are counted;

3) according to rules from the 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;

4) 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:

$$q_{chr} = \frac{fy}{15} + \frac{ry}{30} + \frac{ey}{26} + \frac{uy}{49} + \frac{my}{33} + \frac{sy}{23} + \frac{cy}{9} + \frac{py}{18}, \quad (22)$$

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 above-developed models;

$$D_{chr} = \frac{8 - q_{chr}}{8}, \quad (23)$$

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

5) to evaluate the veracity of software quality assessment based on the available after addition(s) information in the SRS, by the following equations:

$$q'_{chr} = \frac{fy'}{15} + \frac{ry'}{30} + \frac{ey'}{26} + \frac{uy'}{49} + \frac{my'}{33} + \frac{sy'}{23} + \frac{cy'}{9} + \frac{py'}{18}, \quad (24)$$

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

$$D'_{chr} = \frac{8 - q'_{chr}}{8}, \quad (25)$$

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

6) to evaluate the gain of the veracity of software quality assessment after addition(s) of the necessary measures in the SRS, by the following equations:

$$\Delta q_{chr} = q_{chr} - q'_{chr} = (\frac{fy}{15} - \frac{fy'}{15}) + (\frac{ry}{30} - \frac{ry'}{30}) + ... + (\frac{py}{18} - \frac{py'}{18}), \quad (26)$$

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

$$\Delta D_{chr} = D'_{chr} - D_{chr} = \frac{\Delta q_{chr}}{8}, \quad (27)$$

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

The developed methods of evaluating the sufficiency of the SRS information for software quality assessment (by ISO 25010) based on the ontology and the weighted ontology, method of forming the logical conclusion about the sufficiency of the SRS information for software quality assessment (by ISO 25010) provide (on the basis of the comparative analysis of ontologies): 1) the conclusion about the sufficiency or insufficiency of the SRS information (sufficiency or insufficiency of the available measures) for the calculation of the software quality characteristics and subcharacteristics; 2) the conclusion about the need of the addition of measures in the SRS; 3) (if the SRS information is insufficient) the conclusions about for calculation of which characteristics and subcharacteristics the SRS information is insufficient; 4) the sorted (in descending values of weights) list of missing measures as recommended priority of the addition of the measures in the SRS; 5) evaluation of the veracity of the software quality assessment on the basis of the available in the SRS measures; 6) the increasing the veracity of the assessments of software quality at the early stages of the life cycle.

4. System of Evaluating the Sufficiency of the SRS Information for Software Quality Assessment According to ISO 25010:2011

On the basis of the developed in section 3 methods, the system of evaluating the sufficiency of the SRS information for software quality assessment according to ISO 25010 is developed. The structure of this system is represented on Figure 3.

The input of the system of evaluating the sufficiency of the SRS information for software quality assessment according to ISO 25010:2011 is the set $\{qms_1,...qms_{nm}\}$ ($nm \le 138$) of the available in the SRS software quality measures, which are necessary for the subcharacteristcs and characteristics calculation.

For support the user, the measures, which are necessary for software quality assessment according to ISO 25010, is presented in the form of the base ontology for subject domain "Software engineering" (part "Software requirements specification (software quality measures)") considering the distribution by the sections of the SRS – Figure 4. This ontology is the realization of the base (universal) ontological model of the SRS (in terms of the availability of software quality measures), which is represented by equation (19) of section 2, and is the template of the SRS in terms of the availability of software quality measures.

The results of this system are (according to above-developed methods of evaluating the sufficiency of the SRS information for software quality assessment (by ISO 25010) based on the ontology and the weighted ontology, method of forming the logical conclusion about the sufficiency of the SRS information for software quality assessment): 1) conclusion about the sufficiency of the SRS information for software quality assessment according to ISO 25010:2011; 2) recommendations about necessary and priority of the addition of the measures in the SRS for software quality assessment according to ISO 25010:2011; 3) evaluation of the veracity of the software quality assessment according to ISO 25010 on the basis of the available in the SRS measures.

The developed system is based on the comparative analysis of ontologies and is designed for the early stages of the software lifecycle.

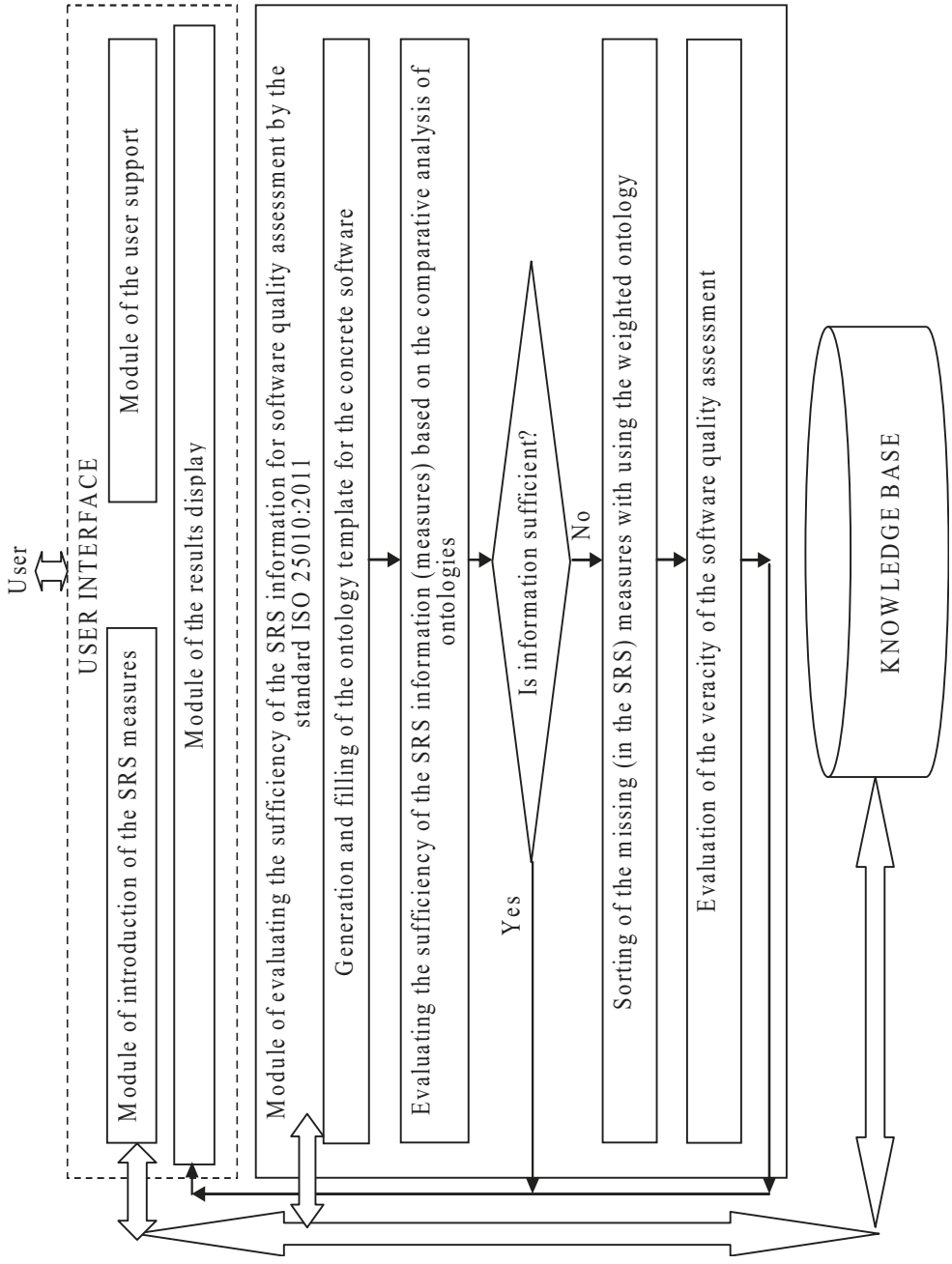

Figure 3. The structure of the system of evaluating the sufficiency of the SRS information for software quality assessment according to ISO 25010:2011

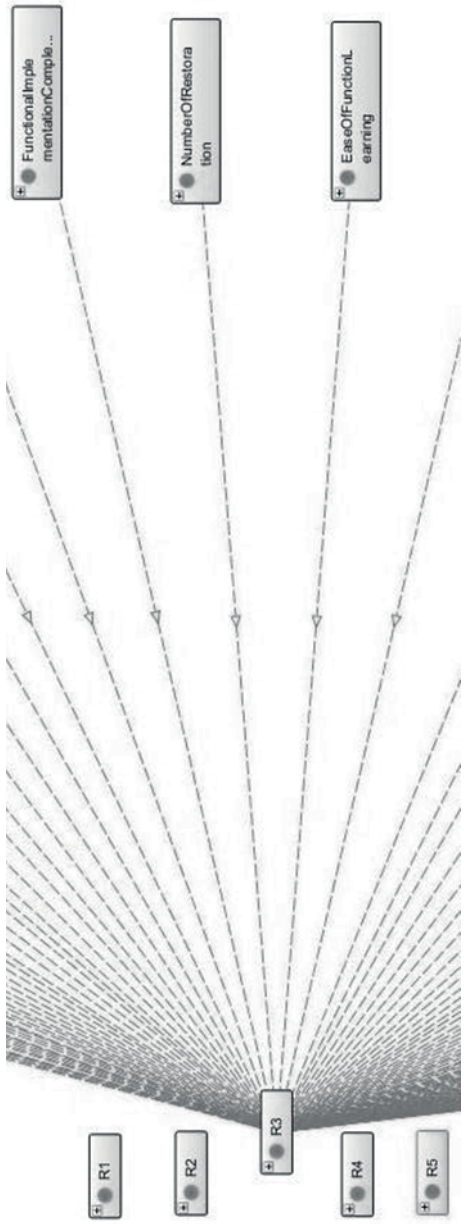

Figure 4. Fragment of the base ontology for subject domain "Software engineering" (part "Software requirements specification (software quality measures)")

5. Experiments: Evaluating the Sufficiency of SRS Information for Software Quality Assessment According to ISO 25010:2011

For experiment, the SRS of automated system (AS) for large-format photo print 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. For example, the part of ontology for Functional Suitability of AS for large-format photo print is represented on Figure 5 (the remaining 7 parts of this ontology have the similar form).

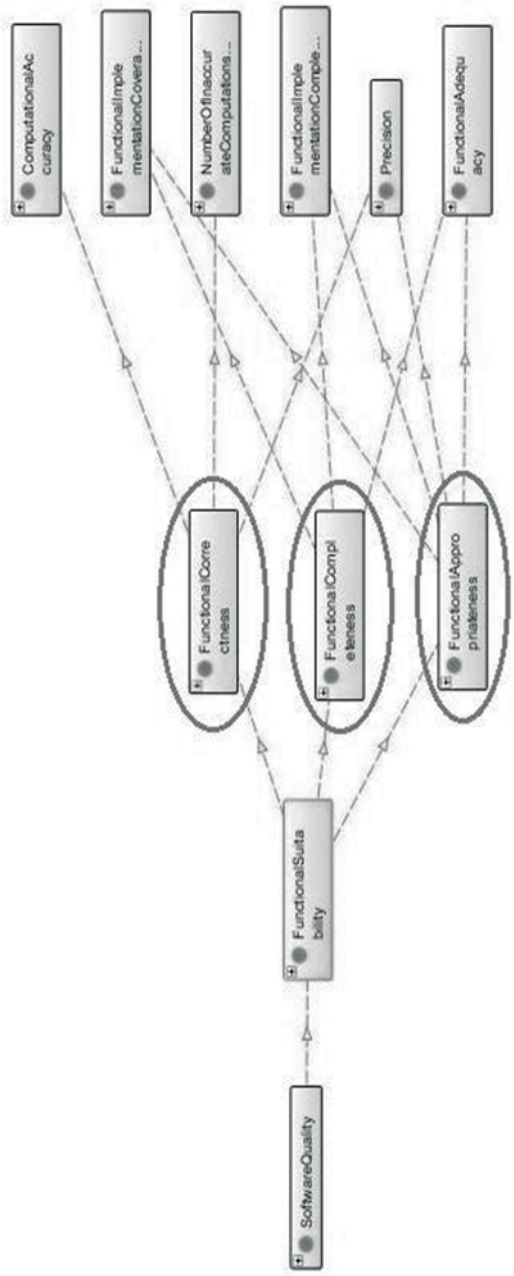

Figure 5. The part of ontology for Functional Suitability of AS for large-format photo print

The comparison (in Protégé 4.2) of the developed ontology for AS for large-format photo print 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 4 measures are absent: «Number Of Functions», «Operation Time», «Number Of Data Items», «Number Of Test Cases». Then the set of missing measures is: $\{qms1_{AS},...,qms4_{AS}\} = \{"NumberOfFunctions", "OperationTime", "NumberOfDataItems", "NumberOfTestCases"\}$.

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 $\{qms1_{AS},...,qms4_{AS}\}$ is performed. According to these rules, the missing in the SRS measures for the evaluation of the subcharacteristics and characteristics are counted. According to the rule No.139, the fact was established, that the available measures in the SRS of AS for large-format photo print are insufficient for calculation of following subcharacteristics: Functional Completeness, Functional Correctness, Functional Appropriateness, Maturity, Availability, Fault Tolerance, Recoverability, Time Behaviour, Resource Utilization, Capacity, Appropriateness Recognisability, Learnability, Operability, Modularity, Analysability, Modifiability, Testability, Confidentiality, Integrity, CoExistence, Interoperability, Adaptability, Replaceability. The subcharacteristics of Functional Suitability, for calculation of which some necessary measures in the SRS lacks, are circled on Figure 5.

According to the 1-st part of the rule No.140, the fact was established, that the available measures in the SRS of AS for large-format photo print are insufficient for calculation of all 8 software quality characteristics. Thus, the lack of 4 measures in the SRS led: to the impossibility of calculating the 23 (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. So the system of evaluating the sufficiency of the SRS information for software quality assessment according to ISO 25010:2011 gives the conclusion: "The information of the SRS is insufficient for software quality assessment according to ISO 25010:2011".

After establishing the fact of insufficiency of information of the SRS of AS for large-format photo print, according to the 2-nd and 3-rd parts of the rule No.140: 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. So the system of evaluating the sufficiency of the SRS information for software quality assessment according to ISO 25010:2011 gives the conclusion: "For increasing the veracity of software quality assessment the next measures should be added in the SRS in this consistency: 1) Operation Time; 2) Number Of Functions; 3) Number Of Data Items; 4) Number Of Test Cases".

Next, the evaluation of the veracity of software quality assessment based on the available in the SRS information is done (according to the equations 22, 23):

$$q_{chr_{AS}} = \frac{5}{15} + \frac{6}{30} + \frac{3}{26} + \frac{5}{49} + \frac{7}{33} + \frac{6}{23} + \frac{4}{9} + \frac{5}{18} = 1{,}95\,, \tag{28}$$

$$D_{chr_{AS}} = \frac{8 - 1{,}95}{8} = 0{,}76\,. \tag{29}$$

So the system of evaluating the sufficiency of the SRS information for software quality assessment according to ISO 25010:2011 gives the conclusion: "The veracity of software quality assessment based on the available in the SRS measures are 79% by subcharacteristics and 76% by characteristics".

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 and the weighted ontology 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 AS for the large-format photo print, the ontology (version 2) for the quality of this software was re-developed. The comparison of the re-developed ontology for AS for large-format photo print with the base ontology for subject domain "Software engineering" (part "Software quality") provides the conclusion, that 2 measures were added in the SRS: «Number Of Functions» (2-nd in the sorted list), «Number Of Data Items» (3-rd in the sorted list). The set of missing measures is $\{qms'_{1_{AS}}, qms'_{2_{AS}}\} = \{"OperationTime", "NumberOfTestCases"\}$.

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 $\{qms'_{1_{AS}}, qms'_{2_{AS}}\}$ is performed. According to the rule No.139, the fact was established, that the available measures in the SRS of AS for large-format photo print are still insufficient for calculation of some subcharacteristics (with indicating these subcharacteristics), but addition 2 measures in the SRS made possible the calculation of Functional Completeness, Capacity, Appropriateness Recognisability, Analyzability, Replaceability of this software.

According to the 1-st part of the rule No.140, the fact was established, that the available measures in the SRS of AS for large-format photo print are still insufficient for calculation of all 8 software quality characteristics. Thus, the lack of 2 measures in the SRS still led 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. So the system of evaluating the sufficiency of the SRS information for software quality assessment according to ISO 25010:2011 again gives the conclusion: "The information of the SRS is insufficient for software quality assessment according to ISO 25010:2011".

After establishing the fact of insufficiency of information of the SRS of AS for large-format photo print, according to the 2-nd and 3-rd parts of the rule No.140: 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. So the system of evaluating the sufficiency of the

SRS information for software quality assessment according to ISO 25010:2011 gives the conclusion: "For increasing the veracity of software quality assessment the next measures should be added in the SRS in this consistency: 1) Operation Time; 2) Number Of Test Cases".

Next, the evaluation of the veracity of software quality assessment based on the available in the SRS information is done (according to the equations 24, 25):

$$q'_{chr_{AS}} = \frac{2}{15} + \frac{5}{30} + \frac{2}{26} + \frac{2}{49} + \frac{4}{33} + \frac{4}{23} + \frac{2}{9} + \frac{1}{18} = 0{,}99\,, \tag{30}$$

$$D'_{chr_{AS}} = \frac{8 - 0{,}99}{8} = 0{,}88\,. \tag{31}$$

So the system of evaluating the sufficiency of the SRS information for software quality assessment according to ISO 25010:2011 gives the conclusion: "The veracity of software quality assessment based on the available in the SRS measures are 90% by subcharacteristics and 88% by characteristics".

The gain of the veracity of software quality assessment after addition of the 2 necessary measures in the SRS is calculated (according to the equations 26, 27):

$$\Delta q_{chr_{AS}} = q_{chr_{AS}} - q'_{chr_{AS}} = 1{,}95 - 0{,}99 = 0.96\,, \tag{32}$$

$$\Delta D_{chr_{AS}} = D'_{chr_{AS}} - D_{chr_{AS}} = 0{,}88 - 0{,}76 = 0{,}12\,, \tag{33}$$

The customer of the developed AS for large-format photo print has decided that further complement of the SRS is economically inexpedient. So it was formed the conclusion about the insufficiency of information for the software quality assessment: the veracity of software quality assessment based on the available in the SRS measures are 90% by subcharacteristics and 88% by characteristics. Experiments confirmed that the system of evaluating the sufficiency of the SRS information for software quality assessment according to ISO 25010 provides increasing the veracity of the software quality assessment in 12% for AS for large-format photo print.

6. Conclusions

Scientific novelty of research results is the development of the methodology of evaluating the sufficiency of the information of the SRS for software quality assessment according to ISO 25010:2011. This methodology is based on the use of ontological models, which reduces the complexity of their development and adaptation to the features of the subject domain, and provides the access, analysis and understanding of the information not only for human, but also for agents.

The first time developed methodology (models, methods, and tools) provides:

1) the formalization of the standards ISO 25010:2011, ISO 25023:2016, ISO/IEC/IEEE 29148:2011;

2) the verification of the degree of implementation of these standards in the software development (especially useful for software projects at the intersection of subject domains);

3) the forming of the conclusion about the SRS information sufficiency or insufficiency for software quality assessment;

4) the forming of the conclusion about necessary of the addition of measures in the SRS;

5) the prioritization of measures additions in SRS;

6) the evaluation of the veracity of software quality assessment according to ISO 25010 based on the available in the SRS measures;

7) the increasing of the veracity of the software quality assessment;

8) the increasing of the software quality at the early stages of the life cycle.

The evolution of the methodology of evaluating the sufficiency of information for software quality assessment is possible through the use of descriptive logic and reasoners for the verification of the developed ontologies and forming the new knowledge. The implementation of methodology (after evolution) in software companies can automate the process of the SRS developing, can allow the use of software agents for supplement the SRS or receiving the new knowledge on its basis. Therefore, the perspective direction for further research is the development of intelligent agents based on ontological approach. These agents will execute automatic analysis of the developed SRS, verification of the sufficiency of the SRS information (considering the recommendations of the standards for software development and standards for subject domain, for which software is developed), addition of the SRS based on the ontology, and will evaluate early stages of lifecycle with the purpose of the minimization of the information and financial losses.

References

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

[2] ISO/IEC TR 19759:2015. “Software Engineering. Guide to the software engineering body of knowledge (SWEBOK)”. ISO/IEC, Geneva, 2015.

[3] The Standish Group International: CHAOS Knowledge Center. (2013). “CHAOS Manifesto – Think big, act small”. [Online]. Available: http://www.versionone.com/assets/img/files/CHAOSManifesto2013.pdf

[4] Sh. Hastie, S. Wojewoda, (2016). “Standish Group 2015 Chaos Report – Q&A with Jennifer Lynch”. [Online]. Available: http://www.infoq.com/articles/standish-chaos-2015

[5] S. McConnell, *Code complete,* Microsoft Press, Redmond, 2013.

[6] C. Shamieh, *Systems engineering for dummies*, Wiley Publishing, New Jersey, 2011.

[7] D. Maevskiy, Y. Kozina, "Where and When Is Formed of Software Quality?" in *Electrical and Computer Systems*, no 18, pp. 55-59, 2016.

[8] A. Chen, J. Beatty, *Visual models for software requirements,* Microsoft Press, Redmond, 2012.

[9] A. Fatwanto, "Software requirements specification analysis using natural language processing technique" in *International Conference on Quality in Research*, Yogyakarta (Indonesia), 2013, pp.105-110.

[10] T. Rehman, M. N. A. Khan, N. Riaz, "Analysis of requirement engineering processes, tools/techniques and methodologies", *Information Technology and Computer Science*, vol. 5, no 3, pp.40-48, 2013.

[11] B. Henderson-Sellers, C. Gonzalez-Perez, T. McBride, G. Low, "An ontology for ISO software engineering standards: 1) Creating the infrastructure", *Computer Standards & Interfaces*, vol. 36, issue 3, pp. 563-576, 2014.

[12] F. B. Ruy, R. A. Falbo, M. P. Barcellos, G. Guizzardi, G. K. S. Quirino, "An ISO-based software process ontology pattern language and its application for harmonizing standards", *ACM SIGAPP Applied Computing Review*, vol. 15, issue 2, pp. 27-40, 2015.

[13] M. M. Hamri, S. M. Benslimane, "Building an Ontology for the Metamodel ISO/IEC24744 using MDA Process", *International Journal on Modern Education and Computer Science,* vol. 8, pp. 48-60, 2015.

[14] D. Jin, J. R. Cordy, "Ontology-based software analysis and reengineering tool integration: The OASIS service-sharing methodology" in *International Conference on Software Maintenance*, Budapest (Hungary), 2005, pp. 613-616.

[15] ISO 25023:2016. "Systems and software engineering. Systems and software Quality Requirements and Evaluation (SQuaRE). Measurement of system and software product quality". ISO/IEC, Geneva, 2016.

[16] ISO/IEC/IEEE 29148-2011. "Systems and software engineering. Life cycle processes. Requirements engineering". ISO/IEC/IEEE, Geneva, 2011.

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

[18] T. Hovorushchenko, "Method of Evaluating the Weights of Software Quality Measures and Indicators", *Application and Theory of Computer Technology*, vol. 2, no. 2, issue 2, pp. 16-25, 2017.