A Risks management method based on the quality requirements communication method in agile approaches

Vasyl Yatsyshyn[a], Oleh Pastukh[a], Andriy Lutskiv[a] , Viktor Tsymbalistyy[a], Nataliia Martsenko[b]

[a] *Ternopil Ivan Puluj National Technical University, Ruska, 56 street, Ternopil, 46001, Ukraine*
[b] *West Ukrainian National University, Lvivska, 11, Ternopil, 46009, Ukraine*

Abstract

In this paper proposed a risks management method based on the quality requirements communication method and SEI risks model. The main value of the method is to increase completeness and traceability of risks during agile software development. This has been achieved by risks identification and its integration to the quality requirements models. Additionally, in the article proposed a formalization of SEI model and the structure of agile software development process, built hierarchical tree with the bipartite vertex at the attributes level (requirement-risk).

Keywords [1]

Risk, software, management, agile, quality requirements

1. Introduction

Software engineering characterized by using and application of different kinds of software life cycle models. Choosing of the concrete model is defined and dependent from the type of designed system, for instance, software for general market, critical and medical systems, real-time system, etc [1].

Software life cycle models, except for general processes, include a set of activities and sub processes, which are oriented to ensure the quality of the end-user software in the way of taking into account project budget and deadlines.

An important aspects and processes of software engineering are risks identifying, management and monitoring at the different stage of life cycle. Different results would have received at the each life cycle phase in the relation of methods and technologies, which are using to manage risks. In this article, we proposed the method of risks management and monitoring at the stages of software life cycle in case of using agile approach. The main content of this method assumes using and application of requirements communication method [10], which take into account SEI model [2].

In general, a risk of software may defined as a possibility of decreasing quality of final product, increasing in cost of software development, a delay in the completion of the development or a disruption of the project. It has relations to the inefficiency, imperfection, and immaturity of the methods, techniques and technology processes at the different stage of life cycle.

From the point of project development view, the most important risks are technical risks. Considering that, it is very fatefully to identify technical risks in the relation of the requirements, because they form a fundament of the future project development and evolution.

Generally, risks divide by two groups: internal and external [3,4].

Internal risks can be presented like a risk events, which has relations to the internal features of some software project development. Development team wield tools and technologies that help to identify, control and manage these events. For example, risk managers use CASE-tools to estimate a duration of some activities, to define deadlines, to make cost estimation and recruitment.

ITTAP'2022: 2nd International Workshop on Information Technologies: Theoretical and Applied Problems, November 22–24, 2022, Ternopil, Ukraine
EMAIL: vyatcyshyn@gmail.com (A. 1); oleg.pastuh@gmail.com (A. 2); l.andriy@gmail.com (A. 3); nata.martsenko@gmail.com (B)
ORCID: 0000-0002-5517-6359 (A. 1); 0000-0002-0080-7053 (A. 2); 0000-0002-9250-4075 (A. 3); 0000-0003-0947-5179 (B);


External risks can be defined as risk events, which appear outside of development team competition and team does not affect to these features (market of software, laws, etc.) [5].

Project Management Institute (PMI) defined system approach to the process of risks management and display it in Project Management Body Of Knowledge (PMBOK).

PMBOK define six main processes of risks management:

- Planning risks management;
- Risks identification;
- Qualitative risks analysis;
- Quantitative risk analysis;
- Planning risks reactions планування реакції на ризики;
- Risks Control and management.

There are many articles and publications devoted to the risks management research [6-8], some of these ideas and concepts are implemented in the standards of software development. The most useful and practical paradigm of risks project management is developed by Software Engineering Institute [7].

2. Analysis of risks management approaches

Many formal and non-formal methods are used to identify and evaluate risks of software failures. The choosing one or another method defines by software application (critical software, software for general market) and criticality of consequences when failures are on.

The most famous formal methods of risks failures analysis is Event Tree Analysis (ETA), Fault Tree Analysis (FTA) and Failure Modes and Effects Analysis (FMEA). Its application is effective in case of complex computer systems designing at the stage of requirements analysis, system architecture design and implementation. These methods help to analyze possible failure modes, develop designs and technical solutions to minimize the consequences of failures.

Event Tree Analysis method provide identification of components in case of failure events which have influence on the security of system functionality. The consequences of each event are traceable until failures of high level will not be define. During creating the chain of events identification, we can calculate probability for each event failure and events combination.

The tree events display a full set of possible consequences for some event of component failure, but do not define reasons of the event.

FMEA and it's edition are generally present as a table methods. Their application can be as additional tools to the previous described methods or can be used as an independent method. FMEA is assigned for the detection of possible kinds of software failures, and conditions of its appearance. These methods are built on the principles of inductive analysis theory, that is effective in the field of identification the conditions of failures appearance in the software components and consequences of defined failures for the whole system.

3. SEI model and methodologies

3.1 Software risks analysis, identification and management

General taxonomies of risks, which had proposed SEI, are useful and convenient for identification of general sources of software project risks, but they need to be adapted to the concrete situations.

Software Engineering Institute define three methodologies of project risks management:

- Software Risk Evaluation (SRE), which includes formal method of risk identification, analysis, monitoring and elimination; it uses in early stages of software development (before deal) and periodically during software life cycle;
- Continuous Risk Management (CRM) based on some principles of risks management during software life cycle and it is independent from concrete methods and tools of risks estimation and elimination;

- Team Risk Management (TRM) defines additional activities of risks management, which has relation to the collaborative working under the project from development team and stakeholders.

In general, workflow of risks analysis might be present in the next sequence:

- Assigning of an experienced team of experts;
- Preparation of special questions and appointment with experts;
- Choosing of risks analysis technique
- Defining risks factors and its priority
- Creating the mechanism of risks action
- Assigning relation between some risks and the total effect from its action
- Risks distribution between members of the project
- Preparation of risks report and its analysis.

Expert assessment method is usually uses in business projects and it can be implemented in the way of studying opinions of experienced specialists. When this method apply, it is important to define allowable, critical and disastrous factors, which take into account its level and probability. To improve results of expert assessment method, it is convenient to use fuzzy logic during risks level evaluation under the expert assessments.

To reduce the procedure of risks identification uses classification, which was proposed SEI. The classification based with taking into account main processes of software life cycle. It includes the most general areas of risks, which have relations to the software features and characteristics, environment and software development process, project constraints, etc.

3.2 Risks classification

The main attention in the article we pay to the software risks at the early stages of life cycle. The most important in this research are internal risks, which has communication with requirements and technical aspects of software development. In the table 1 presented classification of those risks [7].

Many methodologies can be used for gathering information about risks, but the most popular are:

- Expert survey.
- Brain storm.
- Delphi method.
- Crawford cards.

All of the presented before methodologies uses interviewing of experienced specialists, which can make preliminary evaluation of the software project. On our opinion, the most important component in each methodologies is detailed project description. Quality of gathered information define quality of expert conclusions. Because, it is necessary to create rules of software description at different stages of life cycle.

The task of building rules of software description is not trivial, but now there are a few methods, which allow to resolve it with the defined level of acceptability.

Primarily, these are methodologies and methods, which are using at the stage of software specification and design. The most common of them are:

- Diagrams of structured system analysis like ER-diagrams, SADT, IDEF.
- UML diagrams.

Currently, UML is a standard tool of software projects description and support features of object-oriented software development. So far, we can consider that UML is convenient and effective tool of description and gathering input information for risks identification.

The model of complex system, which built using UML notation, describes future software product and includes four main parts:

- logical view;
- implementation view;
- representation of system functionality;
- representation of system structure.

Each part of software description contains a few types of diagrams. These diagrams present some aspect of software model.

Table 1
Technical software risks classification

Class of risks source	Class description	Class element	Attribute of element
Technical aspects of software development	Related to the main processes of life cycle and quality characteristics of software	Requirements features	Stability Completeness Ability to implementation Credibility Reliability Availability Scalability Novelty
		Design and implementation features	Functionality Complexity Interfaces Productivity Testability Hardware constraints Reuse components
		Non-functional and quality characteristics	Maintainability Reliability Safety Security Human factors Usability

When experts define software risks at the stages of life cycle, they use kinds of diagrams, which are displayed in the table 2.

Table 2
UML diagrams application during software project development

Title of UML diagram	Project description
Use Case diagram	Requirements of software project
Class diagram	Preliminary software architecture
Sequence diagram, Activity diagram	Software algorithms and functions implementation
Composite structure diagram	Description of hardware configuration
Component diagram	A number of tools, which will be using during software implementation

UML gives possibility to describe software project and helps experts to receive information in clear and structured view, but it limits their information needs only in technical aspects of software. As a

result, data described with UML is not completeness and does not allow expert to carry out a full risks identification.

According to the SEI risks classification (table 1), UML gives opportunity to evaluate completely or partially only functional requirements and project characteristics. In the table 3, presented attributes of classes (risks), which cannot be identified with UML.

Table 3

Risks, which cannot be defined with UML

Class of risks source	Class description	Class element	Attribute of element
Technical aspects of software development	Related to the main processes of life cycle and quality characteristics of software	Requirements features	Novelty
		Non-functional and quality characteristics	Maintainability Human factors

During risks identification, experts need to receive as input information, except for detailed project description, reports about previously executed projects.

It is necessary to note, that UML is not a single modelling tool, which allows to describe future software. But when developers uses object-oriented methodology, UML application is more powerful and gives the most advantages.

Except for disadvantages of UML, which have relation to the technical aspects of software production like novelty, human factors and maintainability, there are some another: time, deadlines and project budget. In spite of such UML disadvantages, perspectives of this tool application are very essential to the process of risk identification.

3.3 Formal description of SEI model

SEI model defines eighteen the most common risks sources, which can be displayed in the formula 1, as a set of these sources

$$RS = \{rs_1, \dots, rs_{18}\} \quad (1)$$

where rs_i – potential source of risks ($i = 1 \dots 18$).

As a source of risks might be software technical risks, which can be displayed as a set of:

$$TRS = \{rs_1, \dots, rs_7\} \quad (2)$$

where $TRS \subset RS$ – a subset of sources of technical risks, where: rs_1 – functional characteristics; rs_2 – quality characteristics; rs_3 – reliability; rs_4 – applicability;rs_5 – productivity (time); rs_6 – maintainability; rs_7 – reuse components.

Except for technical risks, defined a risks subset, which have relation to the project budget:

$$CRS = \{rs_8, \dots, rs_{10}\} \quad (3)$$

where $TRS \subset RS$ – a subset of cost risks, rs_8 – a limit of total project budget; rs9 – an unavailable project cost; rs10 – a low degree of realism when estimating project costs.

The success and quality of the project are also affected by the risks, associated with the project planning process. Formally, they can be represented as a subset:

$$PRS = \{rs_{11}, \dots, rs_{13}\} \quad (4)$$

where $PRS \subset RS$ – a subset of planning risks sources, rs_{11} – features and possibilities of plan flexibility changing; rs12 – possibilities of violate defined deadlines at life cycle stages; rs13 – a low level of plan realism at the life cycle phases.

One more risks subset can be defined during project development which have relation to the additional and organizational procedures and processes of life cycle. That subset can be presented in the view:

$$MRS = \{rs_{14}, \dots, rs_{18}\}, \quad (5)$$

where $MRS \subset RS$ – a subset of risks sources for the project management processes and procedures, rs14 – project strategy; rs15 – project planning; rs16 – project evaluation; rs17 – project documentation; rs18 – project forecasting.

In general, we can display SEI model as a hierarchical structure (Fig.1).

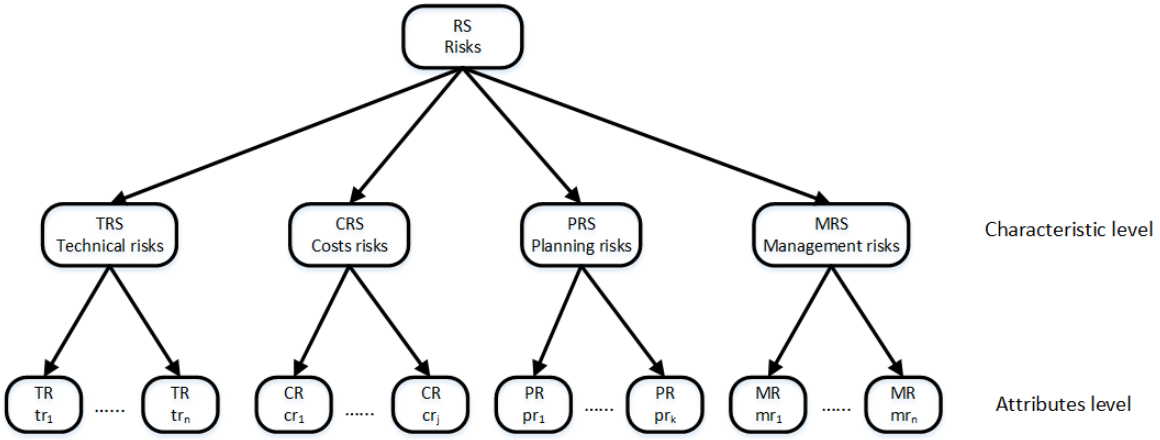

Figure 1: SEI model structure

For each subset, presented previously, defined sets of corresponding attributes:

$$\begin{aligned} TR &= \{tr_i\} \in TRS, i = \overline{1 \dots N} \\ CR &= \{cr_j\} \in CRS, i = \overline{1 \dots J} \\ PR &= \{pr_k\} \in PRS, i = \overline{1 \dots K} \\ MR &= \{mr_l\} \in TRS, i = \overline{1 \dots L} \end{aligned} \quad (6)$$

where TR – a set of attributes of technical risks, CR – a set of attributes of costs risks, PR – a set of attributes of planning risks, MR – a set of attributes of additional and organizational risks, N, J, K, L – a number of attributes of corresponding sets of risks.

In the next sections of the article, this model will be implement to the general process of software development using agile approach and integrated with requirements quality model.

4. Software quality model and requirements communication method

Software quality model is defined in ISO 25010 standard [9]. It includes three models, which display different aspects of software quality. The structure of the model can be presented as hierarchical model, like SEI model, but it includes more levels of hierarchy. The full structure of ISO 25010 models are showed in the Figure 2.

In the [10,11] had proposed method of requirements definition using ISO 25010 models (Fig. 3). The main value of this method is to provide possibilities of displaying requirements in the structured, standard and unified form.

Generally, the process of software requirements analysis and specification begins with the domain analysis, and after that goes sub processes of gathering, classification and defining priority for each requirement. As we watch in the Fig. 3, requirements presentation executed with supporting of three quality models, which allow to structure and provide communication between them with simultaneous standardization.

The main point of quality model using under the requirements to the software is provide a possibility to display them at the different stages of life cycle. For example, requirements, in the view of quality in use model, are useful at the stage of system testing. External quality model requirements can be used at the stage of software architecture design, internal quality model requirements – at the stage of software development (programming code).

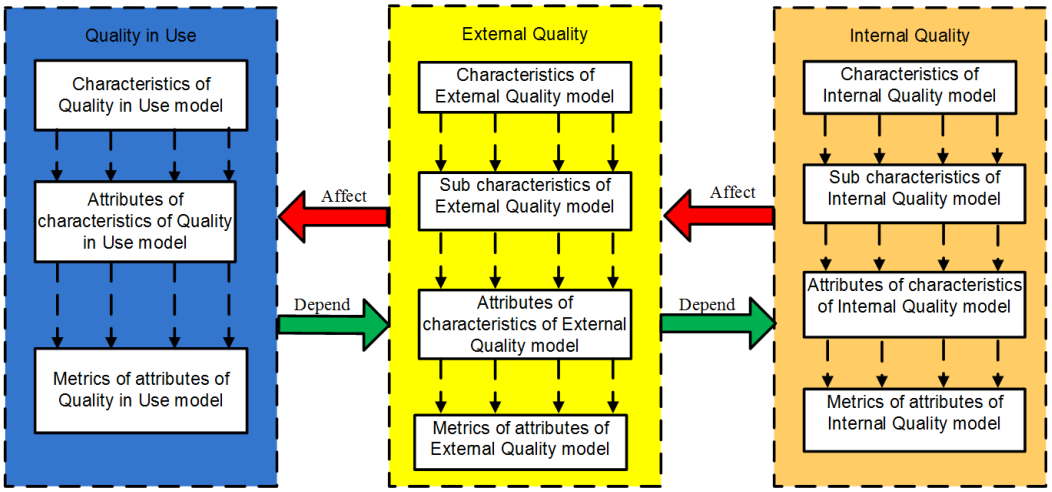

Figure 2. Structure of ISO 25010 quality models

Except for this, in the article [10] developed the method of software requirements communication, which helps to provide requirements tracing at the life cycle stages (Figure 4).

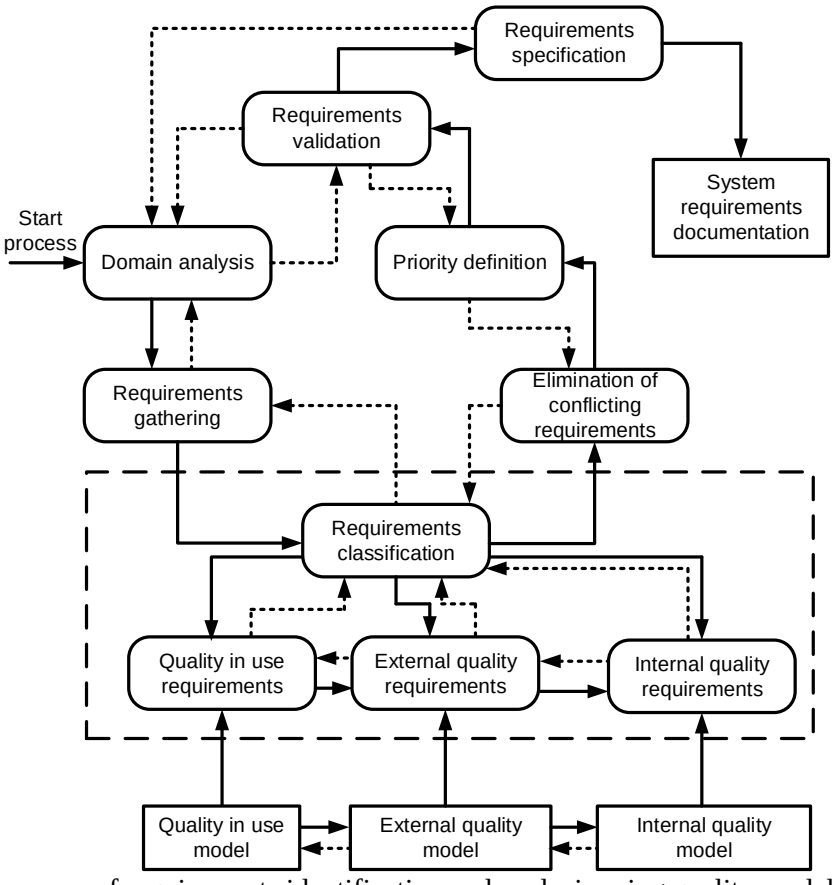

Figure 3. The process of requirements identification and analysis using quality models

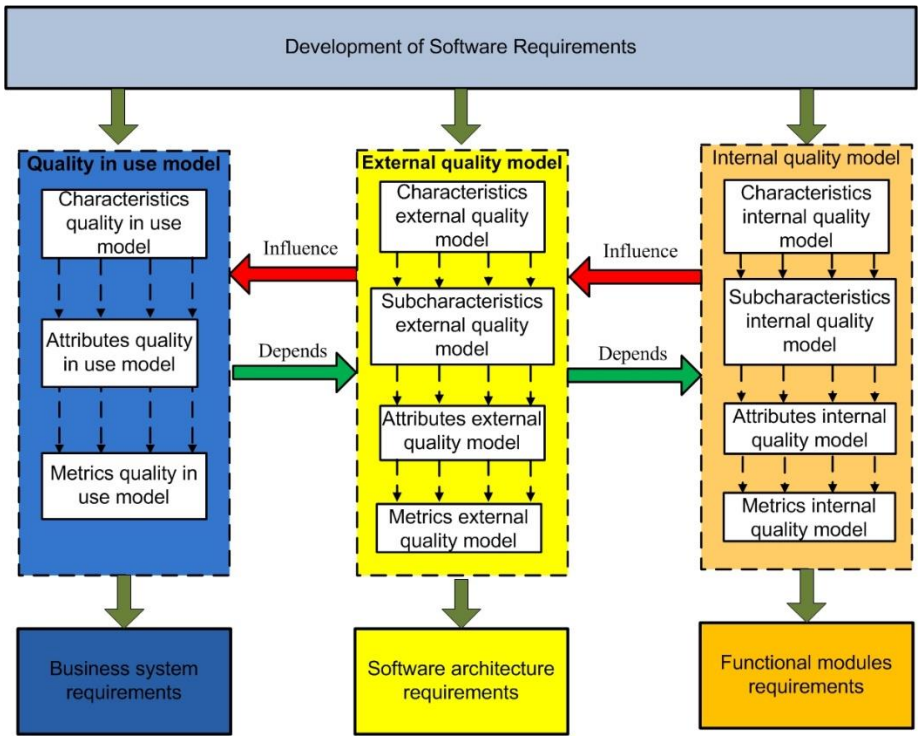

Figure 4. Displaying and communication requirements to the process at life cycle stages

After that, it is necessary to identify risks for each requirement [11]. This way, we will increase completeness of technical risks and provide their traceability during software project development. So far, we can also add risks defined before and included to the SEI model using expert classification methods. As a result, we receive hierarchical tree showed in the Figure 5.

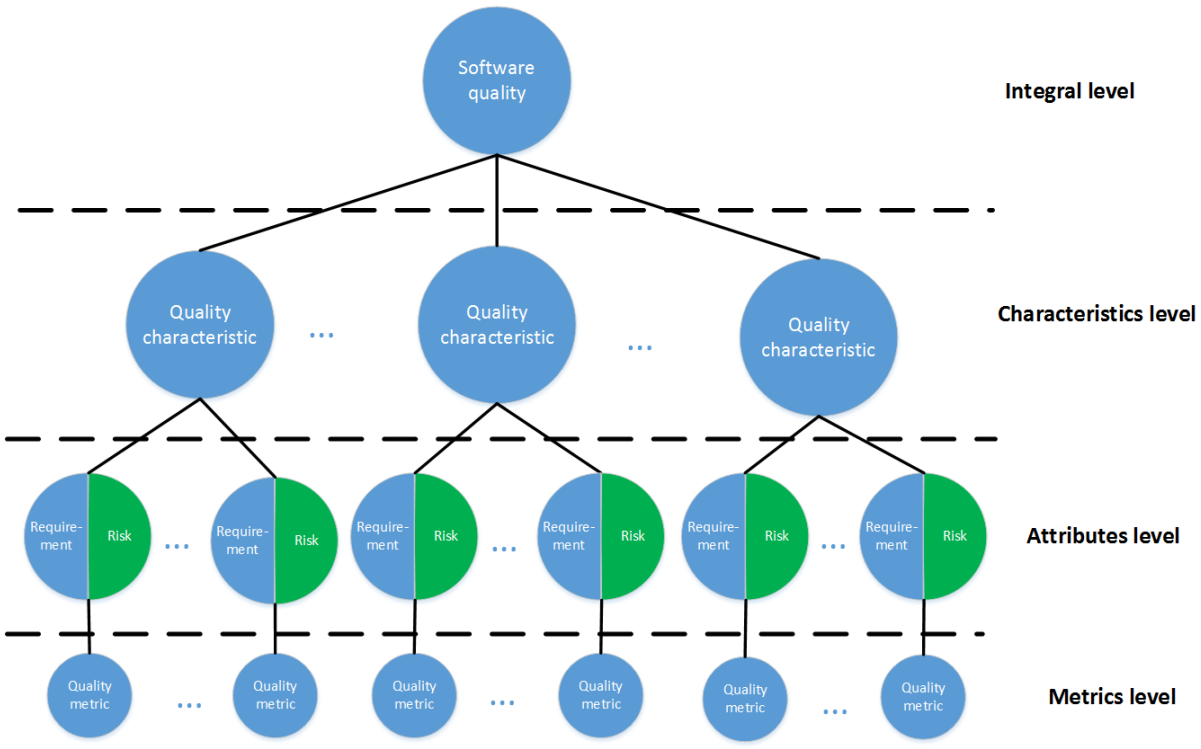

Figure 5. Quality model with integrated to the requirements' attributes SEI risks

Quality model with integrated risks we can look as a graph or a tree and we can apply the signature of graph theory to it. This will help us to define another features of risks, calculate their importance. SEI model, quality models with integrated risks, requirements communication method constitute the basis of proposed risks management method. In this article, we consider that relation between requirement and risk is one-to-one. The vertex of the graph is bipartite at the attributes level. But, in more general case, model, presented in the Fig. 5, must be investigate as a hypergraph. Because on each level, except for metrics level, exists connections between elements. In the next section of this article, we define formal description of agile software development process based on the proposed solutions and method.

5. Risks management method in agile software development

One of the early processes of software development based on agile approach is customer needs analysis [6]. As a result of this iteration received some necessary data, which can be presented as a set, which include two components: need and time label:

$$Needs = \{n_i, t_j\}, \tag{7}$$

where n_i – a customer need in the concrete time moment, $i = 1, N$ – needs number; t_j – a concrete time moment, $j = 1, T$ – a number of time moments,

At the next step, it is necessary to define detailed requirements to the software, which can be presented as:

$$Req = \{r_i, Needs_{ij}\}, \tag{8}$$

where r_i– a requirement to the software, $i = 1, R$ – a number of requirements; $Needs_{ij}$ – customer need, $j \in 1..N$ – a number of needs related to the requirement.

For each requirement, which belong to the Req, it is necessary to define it's priority and detailed requirements will be display as:

$$Req = \{r_i, Needs_{ij}, Weight_i\}, \tag{9}$$

where $Weight_i$ – a weight importance coefficient for the *i*-th requirement.

The values of weight importance coefficients may be define by expert methods, for instance, Saaty methods, methods of simple selection algorithms, methods of average weighted, etc.

Application of agile methodologies involves planning of architectural components at the most highest level. At this level, we can describe system architecture as a set of components:

$$ArcHigh = \{comp_i\}, \tag{10}$$

where $comp_i$– a component of software at the conceptual level.

Architecture components implement functional requirements from the set Req (9), so we can note that:

$$\{Req_i\} \rightarrow comp_j, or\ comp_i \rightarrow Req_{ij}, \tag{11}$$

where $\{Req_i\} \in Req, i = 1..H$ – a subset of requirements, which implement some subsystem; H – a number of requirements, which define component at the conceptual level.

To define priority of a software subsystem or a component proposed to calculate average weighted of requirements priorities, which will be implement in them:

$$Weight(comp_i) = \frac{1}{N}\sum_{j=1}^{H} Weight_j \tag{12}$$

where $Weight(comp_i)$– a weight importance coefficient for an architecture component at the conceptual level; $Weight_j$ – a weight importance coefficient for j-th requirement in i-th component. Defined weight importance coefficients for architecture components in future will be include to a sprint planning.

To implement software components we need to plan tasks of their realization. But before it is necessary to make decomposition of large components to the elementary units and after that include them to the sprint. In general, tasks we can present as a set:

$$Task = \{comp_i\{ecomp_{ik}\}, task_{ikj}\}, \tag{13}$$

where $ecomp_{ik}$ – an elementary unit of i-th component, $k = 1..K$, K – a number of elementary units; $task_{ikj}$ – tasks, which need to implement for i-th component development.

Sprint is one of the important part in agile methodologies, which can be displayed as a set:

$$Sprint = \{Task_i, \{Person\}, t_{start}, t_{end}\} \quad (14)$$

where $Task_i$ – a set of tasks during the sprint; $\{Person\}$ – a set of team members who implement tasks; t_{start} – time of sprint starting; t_{end} – time of sprint finishing.

As we proposed before, technical risks are related to the requirements and we can integrate them to the general agile process in the way:

$$Req = \{\{r_i, \{Risk_{iL}\}\}, Needs_{ij}, Weight_i\} \quad , \quad (15)$$

where $Risk_{iL}$ – a set of risks related to the requirement, $i = \overline{1,N}, L = \overline{1,r_i}$

$Risk_{iL}$ belong to the union of sets:

$$Risk_{iL} \in TRS \cup CRS \cup PRS \cup MRS \quad (16)$$

In future, software risks will be taken into account to each sprint and will increase the process of risks management.

CONCLUSION

In the paper the analysis of software risks management approaches and methods was carried out and defined their main advantages and disadvantages. Application of SEI risks model is very useful during modern software development and it was important to formalize it and integrate to the quality requirements communication method. It provide to improve quality of finish product in the way of effective risks management at the life cycle stages. In this paper we proposed to identify risks according to the requirements and add additional risks defined by SEI model. As a result, we received hierarchical structure, which was integrated with the ISO 25010 quality models. The vertex of the graph is bipartite at the attributes level. But, in more general case, the model must be investigate as a hypergraph and it will be our future task. Also, the paper describe a formal representation of agile software development process structure based on the proposed solutions and method.

References

[1] Ian Sommerville. Software Engineering, Global Edition. Pearson Higher Ed, 2016, 816 pages.

[2] Fenton N. Risk Assessment and Decision Analysiswith Bayesian Networks. CRC Press. –2012. – P. 33-38.

[3] Soumya Krishnan M. Software Development Risk Aspects and Success Frequency on Spiral and Agile Model /M. Soumya Krishnan // Int. Journal of Innovative Research in Computer and Comm. Eng. – 2015. - Vol. 3. – P. 122-129.

[4] Abdullahi M. Strength and Weakness of Software Risk Assessment Tools. International Journal of Software Engineering and Its Applications. – 2014. – Vol. 8. – № 3. - P. 389-398.

[5] Woody C. Supply-Chain Risk Management: Incorporating Security into Software Development. Software Engineering Institute Carnegie Mellon University. – 2010. – P. 166-178.

[6] Britkin A.I. Model estimate the duration of the iterative software development process. Open education, 2009. – №4. – P. 75.

[7] William R. Project Management Body of Knowledge. PMI Standard Committee. 2006. 182 p.

[8] Hijazi H. A Review of Risk Management in Different Software Development Methodologies. Int. Journal of Computer Appl. – 2013. – Vol. 48, № 3. – P.1441-1453.

[9] ISO/IEC 25010:2011ю Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models

[10] Yatsyshyn V, Kharchenko A, Galay I. The Method of Quality Management Software. Proceeding of the VIIth International Conference "Perspective technologies and methods in MEMS design" 11-14 May 2011 - Polyana, Ukraine: Publishing House Vezha&Co. 2011.- p. 228-230.

[11] Kharchenko A., Bodnarchuk I., Yatcysyn V. The Method for Comparative Evaluation of Software Architecture with Accounting of Trade-offs. American Journal of Information Systems, 2014, Vol. 2, No. 1, 20-25/Режим доступу - http://pubs.sciepub.com/ajis/2/1/5/