Вісник Тернопільського національного технічного університету
https://doi.org/10.33108/visnyk_tntu
Scientific Journal of the Ternopil National Technical University
2020, № 1 (97) https://doi.org/10.33108/visnyk_tntu2020.01
ISSN 2522-4433. Web: ***visnyk.tntu.edu.ua***

UDC 004.4

SOFTWARE REQUIREMENTS PROFILE: LIFE CYCLE AND HIS RELATION WITH DEVELOPMENT PROCESSES

Oleksndr Gordieiev

Banking University, Kyiv, Ukraine

Summary. *Priority task of software development is forming of set of requirements for him. Process of forming such set of software requirements named software requirements profiling and his product (set of requirements) named software requirements profile. It should be noted, that profiling can use for other objects of profiling, for example, for forming software characteristics, potential risks of software, software tests, techniques and metrics of software quality verification. Analysis of software development processes, which describe in [1] are represented in article. Further, processes for which profiling is mandatory is defined. As result, between profiling different objects and software development processes establish some conformities. Following software life cycle models: cascade, V-shaped, spiral are analysed. Between software life cycle stages and different profiling objects is established conformity. Implementation of software requirements profiling i.e. forming software requirements profile (this is product of software requirements profiling process) is very important and responsible task for expert. It is connect to the fact that software requirements profiling defines software requirements profile quality – particular «basis» or «foundation» of developed software. The objective of the work is representation and describe life cycle of software requirements profile taking into account possible gaps. Software requirements profile life cycle, which includes following 6 stages: requirements profile in thinking (RPT), requirements profile formulated (RPF), requirements profile for representing (RPR), requirements profile for developing (RPD), requirements profile in software (RPS), requirements profile after verification (RPV) are represented in article. In during transition his from stage to stage according to life cycle of software requirements profile can be some differences (gaps) in software requirements profile. In this connection, full set of chains of differences in software life cycle profile requirements software was formed. Examples of describe such chains were represented also.*

Key words: *software requirements profile life cycle, software requirements profile, software profiling, software development processes, software requirements profile gaps chains.*

https://doi.org/10.33108/visnyk_tntu2020.01.133 *Received 12.04.2020*

Statement of the problem. Software development, from forming terms of reference (TOR) and to testing, is based on nomenclature of standardized and well known processes [1]. Within such processes to software developers exist necessity in tasks, which based on identical principles and actions. In turn, results of execution such tasks use in defined software development processes. For example, such identical tasks can be of following forming set of software characteristics, which are base for non-functional requirements; set of requirements, which describe particularities of functional of software; set of potential risks for software; set of tests; defects; set of software quality verification techniques and metrics. Execution of such identical tasks is named profiling of software [2, 3] and their result (product) – profile of software [2, 3] (software characteristics profile, software requirements profile, software tests profile, software defects profile, software verification techniques and metrics). Obviously, that software profiling process must be connect with software development standardized processes. Establishment such interconnection of software profiling with software development processes need additional more detail their analysis (such analysis will be done in the article).

Implementation of software requirements profiling i. e. forming software requirements profile (this is product of software requirements profiling process) is very important and responsible task for expert. It is connect to the fact that software requirements profiling defines software requirements profile quality – particular «basis» or «foundation» of developed software. Though, that software requirements profile is obvious product of profiling process, the software requirements profile evolves in according to own particular life cycle. Nomenclature, queue and interconnections of stages of software requirements profile life cycle require comprehension, separate research and formal representation.

Analysis of the available investigations. All existing works, which were analysed can divide on following groups:

– first group includes biggest part of articles, which describe of variants of formal represent of software requirements profile [4–8], but software requirements profile life cycle does not review;

– in the following group of articles, were made attempts of formal representation and describe life cycle of requirements profile [9–12], but in full volume, as finished result, works were not completed;

– in following group of in articles, in which software requirements profile model exist [13–14], differences in requirements profile in during the transition from stage to next stage do not review.

Thus, **the objective of the work** is representation and describe life cycle of software requirements profile taking into account possible gaps. According to the objective necessary do following tasks: define place of each software requirements profiling type in software development processes, describe software requirements life cycle taking into account possible gaps.

Profiling in general nomenclature of software development processes. Some additional following definitions will represented. Software profiling – is process forming (choice and harmonisation) subset of connected elements from general set of connected elements or from some connected subsets to general set of connected elements as taxonomic structure (hierarchical, facet or hierarchical-facet structure) such elements with consideration of semantic content of criteria of choice (criteria of profiling). Objects of profiling are software characteristics, software requirements, software risks, software tests, software defects, software verification techniques and metrics. Following examples of such types of profiling are represented:

– software characteristics profiling. It use for forming characteristics (sub characteristics), which accord to software non-functional requirements [15];

– software requirements profiling. It use for forming functional requirements for terms of reference, when need define set needed requirements accord to software functionality [3];

– software risks profiling. It use for forming potential risks in during developing and exploitation critical and business-critical software;

– software tests profiling. It use for forming set of tests for module or integration testing;

– software defects profiling. It use for assessment mechanisms of software fault tolerance, when need receive software defects profile for following their introduction in software in any software life cycle stage;

– software verification techniques profiling. It use for forming set of software verification techniques;

– software metrics profiling. Metric – is technique and scale of assessment. It use for forming for software quality assessment with use set of according metrics.

Software profile – is result of profiling process. Software profile – is result (product) of profiling, which represents as connected taxonomic structure (hierarchical, facet or hierarchical-facet) of elements with semantic content. Software profiles can be following: software characteristics profile, software requirements profile, software tests profile, software defects profile, software verification techniques and metrics.

Software profiling closely connected with software development processes. Such processes must be more detailed analyse, for execution which use profiling. Object for such research will be processes, which describe in standard «Systems and software engineering – Software life cycle processes» [1]. Visually, result such analysis represented in (Figure 1). Processes, which connected with profiling marked grey colour (Figure 1). Following received quantity values, which connected with such research are represented:

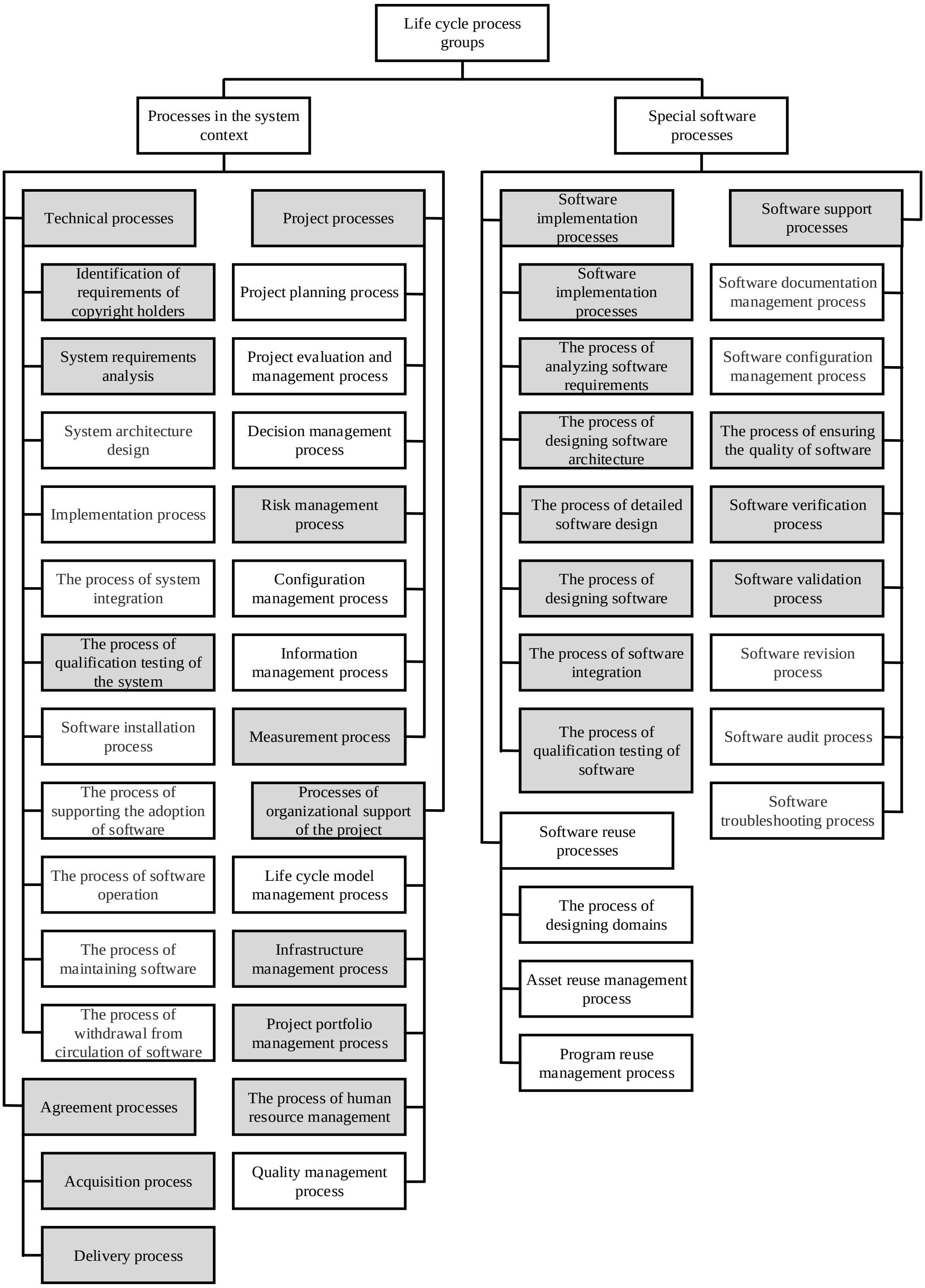
Life cycle process groups
Processes in the system context
Special software processes
Technical processes
Project processes
Software implementation processes
Software support processes
Identification of requirements of copyright holders
Project planning process
Software implementation processes
Software documentation management process
System requirements analysis
Project evaluation and management process
The process of analyzing software requirements
Software configuration management process
System architecture design
Decision management process
The process of designing software architecture
The process of ensuring the quality of software
Implementation process
Risk management process
The process of detailed software design
Software verification process
The process of system integration
Configuration management process
The process of designing software
Software validation process
The process of qualification testing of the system
Information management process
The process of software integration
Software revision process
Software installation process
Measurement process
The process of qualification testing of software
Software audit process
The process of supporting the adoption of software
Processes of organizational support of the project
Software troubleshooting process
Software reuse processes
The process of software operation
Life cycle model management process
The process of designing domains
The process of maintaining software
Infrastructure management process
Asset reuse management process
The process of withdrawal from circulation of software
Project portfolio management process
Program reuse management process
Agreement processes
The process of human resource management
Acquisition process
Quality management process
Delivery process

Figure 1. Software development processes [1]

– all 43 software process development are divided as 7 logical groups (Figure 1);

– software profiling apply in 20 from 43 software processes development (Figure 2), i. e. in 47% from all processes;

– processes of software profiling exist in 6 from 7 logical groups (Figure 2), moreover in 2 groups (agreement and implementation process) for all processes need profiling.

Interconnection between profiling and software development models. Let`s analyse software models development and correlate with him software profiling process for different objects (characteristics, requirements, risks, tests, defects, techniques and metrics verification). Farther need define place for each type of profiling in software development models. Following software development models were selected, which more often use: cascade, V-shaped and spiral. Visually, let`s represent results of anchoring of software profiling process for objects of profiling for cascade model software life cycle (Figure 4), V-shaped model software life cycle (Figure 5) and spiral model software life cycle (Figure 6).

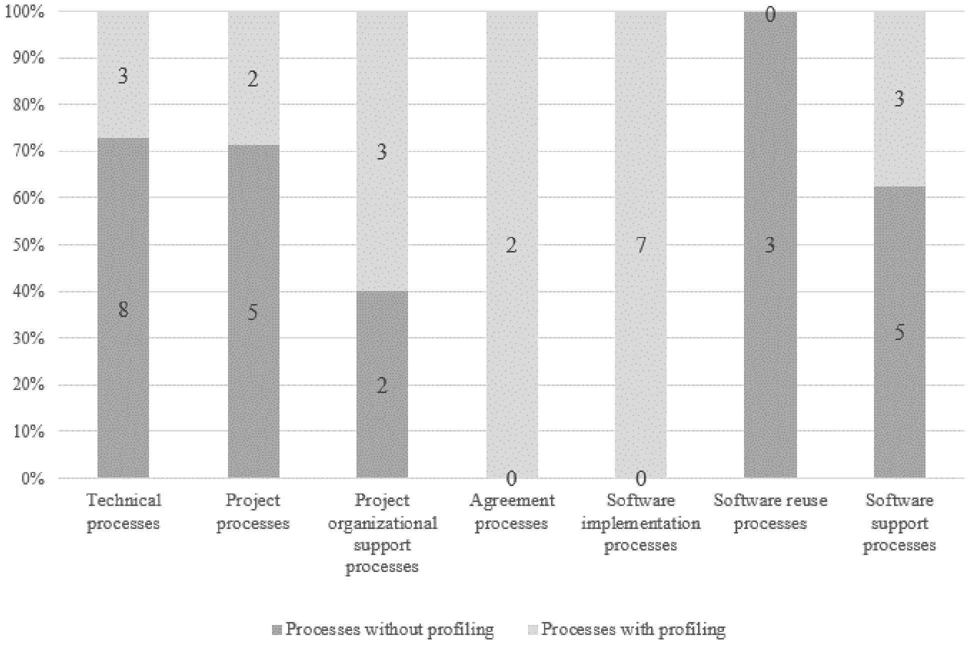

Figure 2. Quantity of processes, which use or do not use profiling

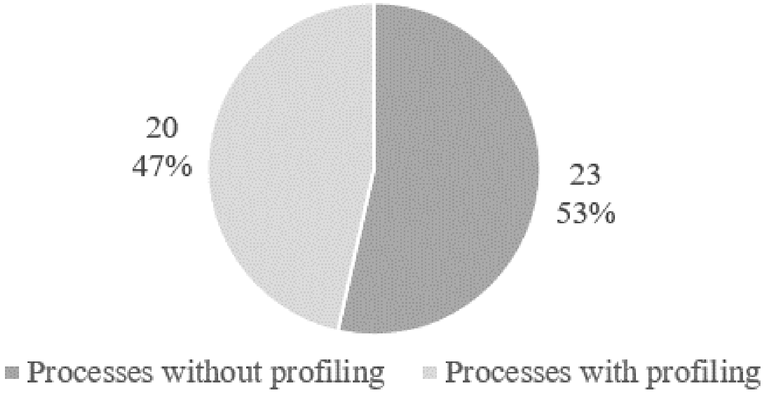

Figure 3. Percentage correlation of processes related to and not related to profiling

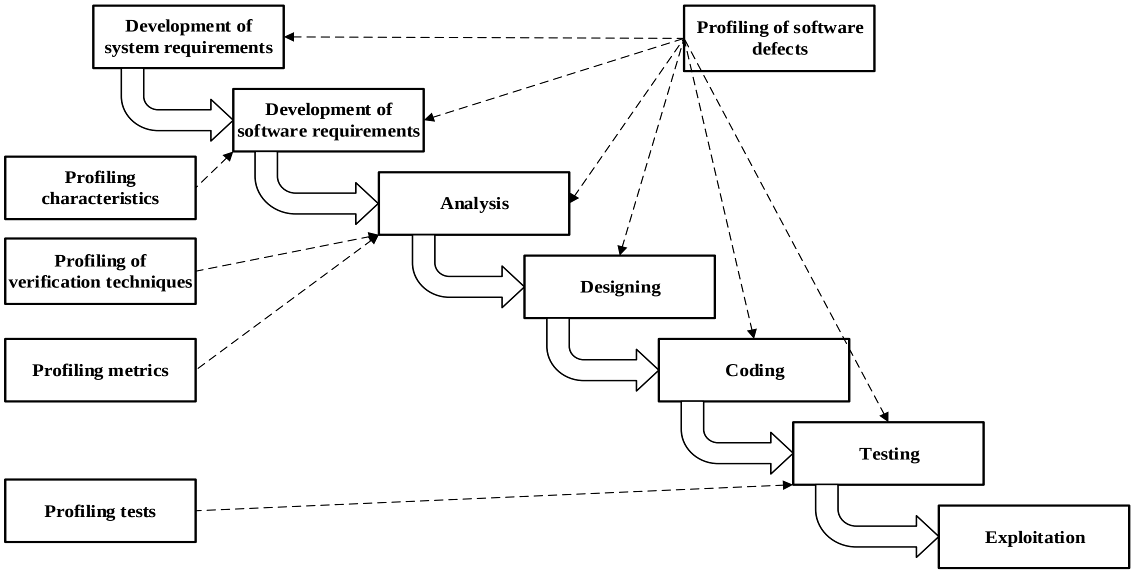
Development of system requirements
Profiling of software defects
Development of software requirements
Profiling characteristics
Analysis
Profiling of verification techniques
Designing
Profiling metrics
Coding
Testing
Profiling tests
Exploitation

Figure 4. Anchoring software life cycle cascade model and profiling types

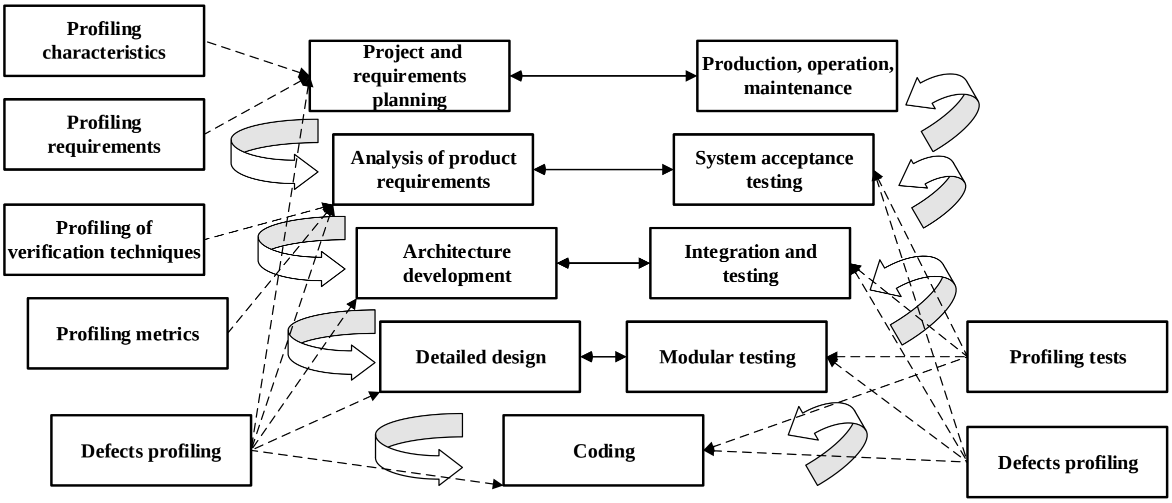
Profiling characteristics
Project and requirements planning
Production, operation, maintenance
Profiling requirements
Analysis of product requirements
System acceptance testing
Profiling of verification techniques
Architecture development
Integration and testing
Profiling metrics
Detailed design
Modular testing
Profiling tests
Defects profiling
Coding
Defects profiling

Figure 5. Anchoring software life cycle V-shaped model and profiling types

Let`s correlate results of anchoring software life cycle and profiling types by quantity of identical profiling types applications for software life cycle models and summary quantity of applications of profiling types. Results such anchoring represented in table 1, in which at intersection lines (software life cycle models) and rows (profiling types) defined quantity corresponding software profiling types for corresponding software life cycle model.

Analysis of data, which represented in table 1 in part applicable of software profiling types give us possibility of formulate following statements:

– all software profiling types apply for software development only in according to spiral software life cycle model. In cascade and V-shaped models do not use profiling of software risks;

– the most applicable software profiling type for all software life cycle models is defects profiling;

– leader of absolute quantity use any type of profiling of software is software life cycle spiral model 27 times, software life cycle V-shaped model – 17 times and software life cycle cascade model – 11 times;

– leader of absolute quantity use in part of software profiling type is software defects profiling – 24 times, software tests profiling – 9 times, software metrics profiling – 8 times software requirements and verification techniques – 4 times for each and software characteristics and risks profiling – 3 times for each.

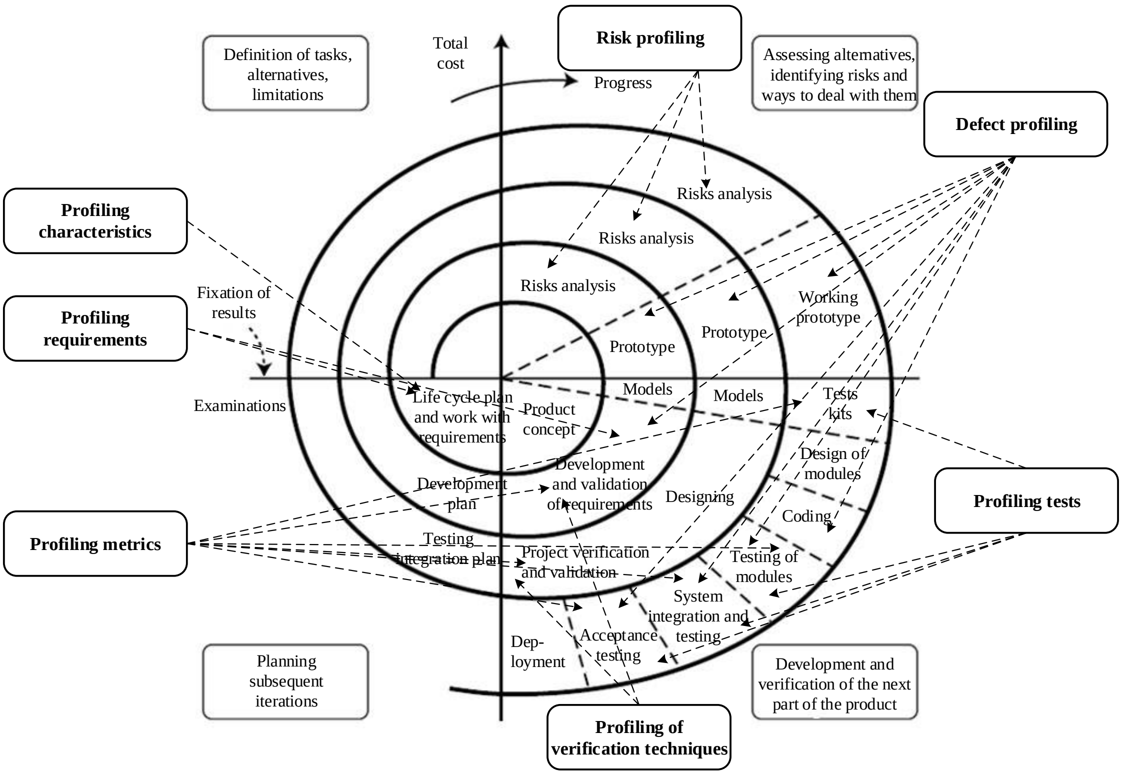
Risk profiling
Total cost
Progress
Definition of tasks, alternatives, limitations
Assessing alternatives, identifying risks and ways to deal with them
Defect profiling
Risks analysis
Risks analysis
Risks analysis
Profiling characteristics
Profiling requirements
Fixation of results
Working prototype
Prototype
Prototype
Models
Models
Examinations
Life cycle plan and work with requirements
Product concept
Tests kits
Design of modules
Development plan
Development and validation of requirements
Designing
Coding
Profiling tests
Profiling metrics
Testing integration plan
Project verification and validation
Testing of modules
System integration and testing
Dep-loyment
Acceptance testing
Planning subsequent iterations
Development and verification of the next part of the product
Profiling of verification techniques

Figure 6. Anchoring software life cycle spiral model and profiling types

Table 1

Anchoring software profiling types and software life cycle models

№	Software life cycle model	Software profiling types							Sum by software life cycle models
		Characteristics	Requirements	Risks	Tests	Defects	Verification techniques	Metrics	
1	Cascade	1	1	-	1	6	1	1	11
2	V-shaped	1	1	-	4	9	1	1	17
3	Spiral	1	2	3	4	9	2	6	27
Sum by profiling types		3	4	3	9	24	4	8	55

Software requirements profile life cycle (evolution). Software requirements profile, which is base for TOR and for product of profiling process has own life cycle. Such life cycle do not started by profiling process and not finished by them. Life cycle of software requirements profile includes stages before and after forming requirements profile for terms of reference. Let's review software requirements profile life cycle. Software requirements profile is being specified in during all own life cycle (all stages). Software requirements profile is comprehended and underwent certain changes in each stage of own life cycle. From one side, this is subjective process, when customer reflects about future of software, other side, such process is objective, because exist conditions and limitations require some changes for current software requirements profile. Thus, exist different (sometimes quite substantial) between requirements profile in customer thoughts and software requirements profile in product (software).

Let`s more detail research of software requirements life cycle is made (evolution) stage-by-stage (Figure 7). Software requirements life cycle includes following 6 stages:

– Stage 1. Requirements profile in thinking (RPT);
– Stage 2. Requirements profile formulated (RPF);
– Stage 3. Requirements profile for representing (RPR);
– Stage 4. Requirements profile for developing (RPD);
– Stage 5. Requirements profile in software (RPS);
– Stage 6. Requirements profile after verification (RPV).

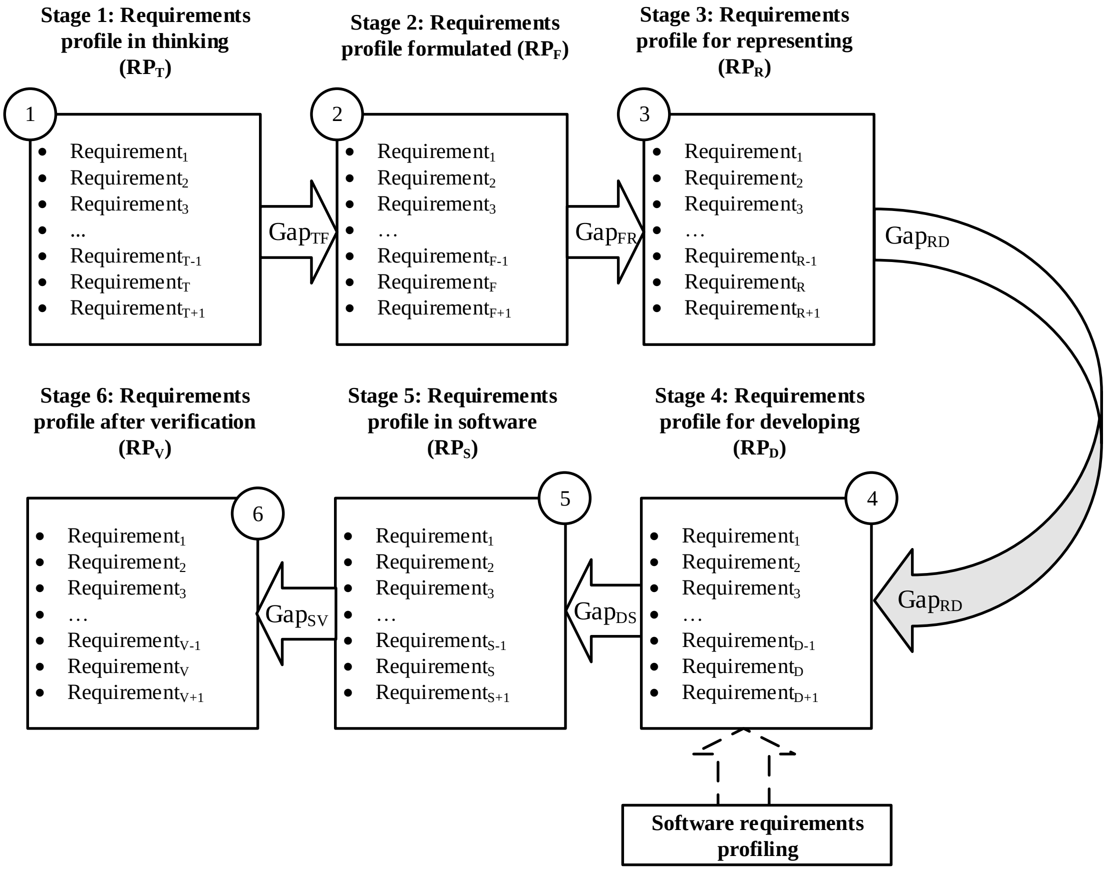

Figure 7. Software requirements profile life cycle (evolution) for software development

Let`s represent software requirements profile according to life cycle, as set of software requirements:

– $RP_T = \{T_i\}_{i=0}^{n}$ – set of requirements profile elements in thoughts of customer;

– $RP_F = \{T_i\}_{i=0}^{n}$ – set of requirements profile elements, which were formulated by customer;

– $RP_R = \{T_i\}_{i=0}^{n}$ – set of requirements profile elements, which were represented by customer on paper;

– $RP_D = \{T_i\}_{i=0}^{n}$ – set of requirements profile elements in terms of reference, which were approved by customer;

– $RP_S = \{T_i\}_{i=0}^{n}$ – set of requirements profile elements, which were realized in software;

– $RP_V = \{T_i\}_{i=0}^{n}$ – set of requirements profile elements, which were verified after of realization software.

Software requirements profile can evolve at transition from one to other stage. I. e. for each stage can forming accorded gap. In this case gap – it is differences between software requirements profile in during transition his from stage to stage according to software requirements profile life cycle. Let`s represents such gaps in formal view:

– $GAP_{TF} = \left\{gap_{TF,i}\right\}_{i=0}^{n}$ – set of differences between requirements in thoughts of customer and requirements, which were formulated by customer;

– $GAP_{FR} = \left\{gap_{FR,i}\right\}_{i=0}^{n}$ – set of differences between requirements, which were formulated by customer and requirements, which were represented by customer on paper;

– $GAP_{RD} = \left\{gap_{RD,i}\right\}_{i=0}^{n}$ – set of differences between requirements, which were represented by customer on paper and requirements in terms of reference, which were approved by customer;

– $GAP_{DS} = \left\{gap_{DS,i}\right\}_{i=0}^{n}$ – set of differences between requirements, which were approved by customer and requirements, which were realized in software;

– $GAP_{SV} = \left\{gap_{SV,i}\right\}_{i=0}^{n}$ – set of differences between requirements, which were realized in software and requirements, which were verified after of realization software.

Let`s represents and describes full set of chains of differences of software requirements profile (gaps) in during life cycle (table 2). Each chain of differences of software requirements profile in general view has following representation (1):

$$SRPGC = \left\{GAP_{TF}, GAP_{FR}, GAP_{RD}, GAP_{DS}, GAP_{SV}\right\}, \tag{1}$$

where SRGC – Software Requirements Profile Gaps Chain. I. e. each such chain represents set differences during the transition from stage to stage in the software requirements life cycle. For example, Let`s, chains of gaps with numbers 1, 22 and 32 are represented (from table 2):

– chain № 1, when all sets of differences in part of software requirements profile represent permanent value and equal of empty set:

$$SRPGC = \left\{GAP_{TF} = GAP_{FR} = GAP_{RD} = GAP_{DS} = GAP_{SV} = \varnothing\right\}; \tag{2}$$

– chain № 22, when all sets of differences in part of software requirements profile partly equal of empty set and partly include some differences:

Table 2

Full set of chains of differences of software requirements profile (gaps) in during life cycle

№	Transitions from stage to stage in the software requirements life cycle				
	$RP_T \to RP_F$	$RP_F \to RP_R$	$RP_R \to RP_D$	$RP_D \to RP_S$	$RP_S \to RP_V$
1.	$GAP_{TF} = \varnothing$	$GAP_{FR} = \varnothing$	$GAP_{RD} = \varnothing$	$GAP_{DS} = \varnothing$	$GAP_{SV} = \varnothing$
2.	$GAP_{TF} = \varnothing$	$GAP_{FR} = \varnothing$	$GAP_{RD} = \varnothing$	$GAP_{DS} = \varnothing$	$GAP_{SV} \neq \varnothing$
3.	$GAP_{TF} = \varnothing$	$GAP_{FR} = \varnothing$	$GAP_{RD} = \varnothing$	$GAP_{DS} \neq \varnothing$	$GAP_{SV} = \varnothing$
4.	$GAP_{TF} = \varnothing$	$GAP_{FR} = \varnothing$	$GAP_{RD} = \varnothing$	$GAP_{DS} \neq \varnothing$	$GAP_{SV} \neq \varnothing$
5.	$GAP_{TF} = \varnothing$	$GAP_{FR} = \varnothing$	$GAP_{RD} \neq \varnothing$	$GAP_{DS} = \varnothing$	$GAP_{SV} = \varnothing$
6.	$GAP_{TF} = \varnothing$	$GAP_{FR} = \varnothing$	$GAP_{RD} \neq \varnothing$	$GAP_{DS} = \varnothing$	$GAP_{SV} \neq \varnothing$
7.	$GAP_{TF} = \varnothing$	$GAP_{FR} = \varnothing$	$GAP_{RD} \neq \varnothing$	$GAP_{DS} \neq \varnothing$	$GAP_{SV} = \varnothing$
8.	$GAP_{TF} = \varnothing$	$GAP_{FR} = \varnothing$	$GAP_{RD} \neq \varnothing$	$GAP_{DS} \neq \varnothing$	$GAP_{SV} \neq \varnothing$
9.	$GAP_{TF} = \varnothing$	$GAP_{FR} \neq \varnothing$	$GAP_{RD} = \varnothing$	$GAP_{DS} = \varnothing$	$GAP_{SV} = \varnothing$
10.	$GAP_{TF} = \varnothing$	$GAP_{FR} \neq \varnothing$	$GAP_{RD} = \varnothing$	$GAP_{DS} = \varnothing$	$GAP_{SV} \neq \varnothing$
11.	$GAP_{TF} = \varnothing$	$GAP_{FR} \neq \varnothing$	$GAP_{RD} = \varnothing$	$GAP_{DS} \neq \varnothing$	$GAP_{SV} = \varnothing$
12.	$GAP_{TF} = \varnothing$	$GAP_{FR} \neq \varnothing$	$GAP_{RD} = \varnothing$	$GAP_{DS} \neq \varnothing$	$GAP_{SV} \neq \varnothing$
13	$GAP_{TF} = \varnothing$	$GAP_{FR} \neq \varnothing$	$GAP_{RD} \neq \varnothing$	$GAP_{DS} = \varnothing$	$GAP_{SV} = \varnothing$
14.	$GAP_{TF} = \varnothing$	$GAP_{FR} \neq \varnothing$	$GAP_{RD} \neq \varnothing$	$GAP_{DS} = \varnothing$	$GAP_{SV} \neq \varnothing$
15.	$GAP_{TF} = \varnothing$	$GAP_{FR} \neq \varnothing$	$GAP_{RD} \neq \varnothing$	$GAP_{DS} \neq \varnothing$	$GAP_{SV} = \varnothing$
16.	$GAP_{TF} = \varnothing$	$GAP_{FR} \neq \varnothing$	$GAP_{RD} \neq \varnothing$	$GAP_{DS} \neq \varnothing$	$GAP_{SV} \neq \varnothing$
17.	$GAP_{TF} \neq \varnothing$	$GAP_{FR} = \varnothing$	$GAP_{RD} = \varnothing$	$GAP_{DS} = \varnothing$	$GAP_{SV} = \varnothing$
18.	$GAP_{TF} \neq \varnothing$	$GAP_{FR} = \varnothing$	$GAP_{RD} = \varnothing$	$GAP_{DS} = \varnothing$	$GAP_{SV} \neq \varnothing$
19.	$GAP_{TF} \neq \varnothing$	$GAP_{FR} = \varnothing$	$GAP_{RD} = \varnothing$	$GAP_{DS} \neq \varnothing$	$GAP_{SV} = \varnothing$
20.	$GAP_{TF} \neq \varnothing$	$GAP_{FR} = \varnothing$	$GAP_{RD} = \varnothing$	$GAP_{DS} \neq \varnothing$	$GAP_{SV} \neq \varnothing$
21.	$GAP_{TF} \neq \varnothing$	$GAP_{FR} = \varnothing$	$GAP_{RD} \neq \varnothing$	$GAP_{DS} = \varnothing$	$GAP_{SV} = \varnothing$
22.	$GAP_{TF} \neq \varnothing$	$GAP_{FR} = \varnothing$	$GAP_{RD} \neq \varnothing$	$GAP_{DS} = \varnothing$	$GAP_{SV} \neq \varnothing$
23.	$GAP_{TF} \neq \varnothing$	$GAP_{FR} = \varnothing$	$GAP_{RD} \neq \varnothing$	$GAP_{DS} \neq \varnothing$	$GAP_{SV} = \varnothing$
24.	$GAP_{TF} \neq \varnothing$	$GAP_{FR} = \varnothing$	$GAP_{RD} \neq \varnothing$	$GAP_{DS} \neq \varnothing$	$GAP_{SV} \neq \varnothing$
25.	$GAP_{TF} \neq \varnothing$	$GAP_{FR} \neq \varnothing$	$GAP_{RD} = \varnothing$	$GAP_{DS} = \varnothing$	$GAP_{SV} = \varnothing$
26.	$GAP_{TF} \neq \varnothing$	$GAP_{FR} \neq \varnothing$	$GAP_{RD} = \varnothing$	$GAP_{DS} = \varnothing$	$GAP_{SV} \neq \varnothing$
27.	$GAP_{TF} \neq \varnothing$	$GAP_{FR} \neq \varnothing$	$GAP_{RD} = \varnothing$	$GAP_{DS} \neq \varnothing$	$GAP_{SV} = \varnothing$
28.	$GAP_{TF} \neq \varnothing$	$GAP_{FR} \neq \varnothing$	$GAP_{RD} = \varnothing$	$GAP_{DS} \neq \varnothing$	$GAP_{SV} \neq \varnothing$
29.	$GAP_{TF} \neq \varnothing$	$GAP_{FR} \neq \varnothing$	$GAP_{RD} \neq \varnothing$	$GAP_{DS} = \varnothing$	$GAP_{SV} = \varnothing$
30.	$GAP_{TF} \neq \varnothing$	$GAP_{FR} \neq \varnothing$	$GAP_{RD} \neq \varnothing$	$GAP_{DS} = \varnothing$	$GAP_{SV} \neq \varnothing$
31.	$GAP_{TF} \neq \varnothing$	$GAP_{FR} \neq \varnothing$	$GAP_{RD} \neq \varnothing$	$GAP_{DS} \neq \varnothing$	$GAP_{SV} = \varnothing$
32.	$GAP_{TF} \neq \varnothing$	$GAP_{FR} \neq \varnothing$	$GAP_{RD} \neq \varnothing$	$GAP_{DS} \neq \varnothing$	$GAP_{SV} \neq \varnothing$

$$SRPGC = \left\{ \begin{matrix} GAP_{TF} = \left\{ gap_{TF,i} \right\}_{i=1}^{n}, GAP_{FR} = \varnothing, GAP_{RD} = \left\{ gap_{RD,j} \right\}_{j=1}^{m}, \\ GAP_{DS} = \varnothing, GAP_{SV} = \left\{ gap_{SV,k} \right\}_{k=1}^{p} \end{matrix} \right\}; \qquad (3)$$

– chain № 32, when all sets of differences in part of software requirements profile not equal of empty and include differences:

$$SRPGC = \left\{ \begin{matrix} GAP_{TF} = \left\{ gap_{TF,i} \right\}_{i=1}^{n}, GAP_{FR} = \left\{ gap_{FR,q} \right\}_{q=1}^{s}, GAP_{RD} = \left\{ gap_{RD,j} \right\}_{j=1}^{m}, \\ GAP_{DS} = \left\{ gap_{DS,v} \right\}_{v=1}^{w}, GAP_{SV} = \left\{ gap_{SV,k} \right\}_{k=1}^{p} \end{matrix} \right\}. \qquad (4)$$

Conclusions. Article includes software development processes analysis and their correspondence with software profiling types. Life cycle of software requirements profile taking into account possible gaps in during transition his from stage to stage according are presented and described. In the future works author planning to describe of format of possible gaps in the software requirements profile and describe real example of life cycle of software requirements profile. Received theoretical results can more formal describe and review software requirements profile life cycle. And in future process of control gaps of software requirements profile in the context of his life cycle can be automatize.

References

1. International standard. ISO/IEC/IEEE 12207:2017 Systems and software engineering – Software life cycle processes. 2017. 145 p.
2. Humphrey W., Daughtrey T. The software quality profile. Fundamental Concepts for the Software Quality Engineer. American Society for Quality (ASQ). 2001. P. 3–17.
3. Kharchenko V., Gordieiev O., Fedoseeva A. Profiling of Software Requirements for the Pharmaceutical Enterprise Manufacturing Execution System. Applications of Computational Intelligence in Biomedical Technology. Studies in Computational Intelligence. 2016. Vol. 606. P. 67–93. https://doi.org/10.1007/978-3-319-19147-8_4
4. Andrashov A. A., Kremenchutskiy YU. A., Kharchenko V. S. Analiz modeley predstavleniya trebovaniy k programmnomu obespecheniyu pri ikh profilirovanii. Radioyelektronni i komp’yuterni sistemi. 2009. № 7 (41). P. 186–191.
5. Zhen Yu. Chen, Shengji Yao, Jian Qiang Lin, Yong Zeng, Armin Eberlein Yu Chen Zhen. Formalisation of product requirements: From natural language descriptions to formal specifications. International Journal of Manufacturing Research (IJMR). 2007. Vol. 2. No. 3. P. 362–387. https://doi.org/10.1504/IJMR.2007.014730
6. Shostak I. V. Butenko YU. I. Podkhod k avtomatizatsii protsessa formirovaniya normativnogo profilya pri sertifikatsii programmnykh produktov. Sistemy obrobki informatsii. 2010. № 8 (89). P. 122–126.
7. Andrashov A. A. Taksonomicheskiye modeli profilirovaniya trebovaniy informatsionno-upravlyayushchikh sistem kriticheskogo primeneniya. Radíoyelektronni i komp’yuterni sistemi. 2010. № 7 (48). P. 104–108.
8. Andrashov A. A. Fasetno-iyerarkhicheskiye semanticheskiye struktury v zadachakh obespecheniya kachestva programmnogo obespecheniya. Integrirovannyye tekhnologii v mashinostroyenii “ÍKTM-2008”: mater. Mizhnar. nauk.-tekhn. konf. (m. Kharkív, 2008.). Kharkív, 2008. T. 2. P. 204.
9. Nayan B. Ruparelia Software development lifecycle models. ACM SIGSOFT Software Engineering Notes. 2017. Vol. 35. Num. 3. P. 8–13. https://doi.org/10.1145/1764810.1764814
10. Meyer B., Bruel J.-M., Ebersold S., Galinier F., Naumchev A. Towards an Anatomy of Software Requirements: In proceedings of the 51st International Conference, TOOLS 2019 (Innopolis, Russia, October 15–17. 2019.). Innopolis, 2019. P. 10–40. https://doi.org/10.1007/978-3-030-29852-4_2
11. Richard F. Schmidt Understanding Software Requirements. Software Engineering: Architecture-driven Software Development. 2013. P. 121–137. https://doi.org/10.1016/B978-0-12-407768-3.00007-0

12. Nazaruka E., Osis J. The Formal Reference Model for Software Requirements. Part of the Communications in Computer and Information Science book series (CCIS). 2018. Vol. 1023. P. 352–372. https://doi.org/10.1007/978-3-030-22559-9_16
13. Hussain S. N., Siddiqui A. T. Software Requirements and Process Models. LAP Lambert Academic Publishing. 2017. 68 p.
14. Innab N., Kayed A., Sajeev A. S. M. An ontology for software requirements modelling: In proceedings of the IEEE International Conference on Information Science and Technology (Hubei, China, March 23–25, 2012.). Hubei, 2012. P. 485–490. https://doi.org/10.1109/ICIST.2012.6221694
15. Gordieiev O., Kharchenko V., Fominykh N., Sklyar V. Evolution of software Quality Models in Context of the Standard ISO 25010: In proceedings of the International Conference on Dependability on Complex Systems DepCoS – RELCOMEX (DepCOS) (Brunow, Poland, June 30 July 4, 2014.). Brunow, 2014. P. 223–233. https://doi.org/10.1007/978-3-319-07013-1_21

Список використаної літератури

1. International standard. ISO/IEC/IEEE 12207:2017 Systems and software engineering – Software life cycle processes. 2017. 145 p.
2. Humphrey W., Daughtrey T. The software quality profile. Fundamental Concepts for the Software Quality Engineer. American Society for Quality (ASQ). 2001. P. 3–17.
3. Kharchenko V., Gordieiev O., Fedoseeva A. Profiling of Software Requirements for the Pharmaceutical Enterprise Manufacturing Execution System. Applications of Computational Intelligence in Biomedical Technology. Studies in Computational Intelligence. 2016. Vol. 606. P. 67–93. https://doi.org/10.1007/978-3-319-19147-8_4
4. Андрашов А. А., Кременчуцкий Ю. А., Харченко В. С. Анализ моделей представления требований к программному обеспечению при их профилировании. Радіоелектронні і комп’ютерні системи. 2009. № 7 (41). С. 186–191.
5. Zhen Yu. Chen, Shengji Yao, Jian Qiang Lin, Yong Zeng, Armin Eberlein Yu Chen Zhen. Formalisation of product requirements: From natural language descriptions to formal specifications. International Journal of Manufacturing Research (IJMR). 2007. Vol. 2. No. 3. P. 362–387. https://doi.org/10.1504/IJMR.2007.014730
6. Шостак И. В., Бутенко Ю. И. Подход к автоматизации процесса формирования нормативного профиля при сертификации программных продуктов. Системи обробки інформації. 2010. № 8 (89). С. 122–126.
7. Андрашов А. А. Таксономические модели профилирования требований информационно-управляющих систем критического применения. Радіоелектронні і комп’ютерні системи. 2010. № 7 (48). Р. 104–108.
8. Андрашов А. А. Фасетно-иерархические семантические структуры в задачах обеспечения качества программного обеспечения. Інтегровані комп’ютерні технології в машинобудуванні «ІКТМ-2008»: матеріали Міжнар. наук.-техн. конф. (м. Харків. 2008.). Харків, 2008. Т. 2. С. 204.
9. Nayan B. Ruparelia Software development lifecycle models. ACM SIGSOFT Software Engineering Notes. 2017. Vol. 35. Num. 3. P. 8–13. https://doi.org/10.1145/1764810.1764814
10. Meyer B., Bruel J.-M., Ebersold S., Galinier F., Naumchev A. Towards an Anatomy of Software Requirements: In proceedings of the 51st International Conference, TOOLS 2019 (Innopolis, Russia, October 15–17. 2019.). Innopolis, 2019. P. 10–40. https://doi.org/10.1007/978-3-030-29852-4_2
11. Richard F. Schmidt Understanding Software Requirements. Software Engineering: Architecture-driven Software Development. 2013. P. 121–137. https://doi.org/10.1016/B978-0-12-407768-3.00007-0
12. Nazaruka E., Osis J. The Formal Reference Model for Software Requirements. Part of the Communications in Computer and Information Science book series (CCIS). 2018. Vol. 1023. P. 352–372. https://doi.org/10.1007/978-3-030-22559-9_16
13. Hussain S. N., Siddiqui A. T. Software Requirements and Process Models. LAP Lambert Academic Publishing. 2017. 68 p.
14. Innab N., Kayed A., Sajeev A. S. M. An ontology for software requirements modelling: In proceedings of the IEEE International Conference on Information Science and Technology (Hubei, China, March 23–25, 2012.). Hubei, 2012. P. 485–490. https://doi.org/10.1109/ICIST.2012.6221694
15. Gordieiev O., Kharchenko V., Fominykh N., Sklyar V. Evolution of software Quality Models in Context of the Standard ISO 25010: In proceedings of the International Conference on Dependability on Complex Systems DepCoS – RELCOMEX (DepCOS) (Brunow, Poland, June 30 July 4, 2014.). Brunow, 2014. P. 223–233. https://doi.org/10.1007/978-3-319-07013-1_21

УДК 004.4

ПРОФІЛЬ ВИМОГ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ: ЖИТТЄВИЙ ЦИКЛ ТА ЙОГО ЗВ'ЯЗОК З ПРОЦЕСАМИ РОЗРОБКИ

Олександр Гордєєв

ДВНЗ «Університет банківської справи», Київ, Україна

Резюме. *Першочерговим завданням розробки програмного забезпечення є формування множини вимог до нього. Процес формування такої множини вимог ПЗ називається профілюванням, а його продукт (безпосередньо множина вимог ПЗ) профілем вимог ПЗ. Слід зазначити, що профілювання застосуються для інших об'єктів профілювання також, наприклад, для формування характеристик ПЗ, потенційних ризиків ПЗ, тестових наборів ПЗ, технік і метрик верифікації якості ПЗ. У статті проводиться аналіз процесів розробки ПЗ, які описані [1]. Далі визначаються процеси, для яких обов'язковим є профілювання. В результаті між профілюванням різних об'єктів і процесами розробки ПЗ встановлюється певні відповідності. Аналізуються моделі життєвого циклу програмного забезпечення: каскадна, V-образна і спіральна. Встановлюється відповідність між етапами життєвого циклу ПЗ відповідних моделей і різними об'єктами профілювання. Виконання процесу профілювання вимог ПЗ, продуктом якого є профіль вимог ПЗ, є найвідповідальнішим для експертів завданням. Це пов'язано з тим, що якість процесу профілювання вимог ПЗ визначає якість профілю вимог ПЗ – своєрідну «основу» або «фундамент» програмного забезпечення, яке розробляється. Метою статті є представлення та опис життєвого циклу програмного забезпечення з урахуванням можливих розривів. Представляється і описується життєвий цикл профілю вимог ПЗ, який включає в себе наступні 6 етапів: профіль вимог у думках замовника; профіль вимог, який був сформульований замовником; профіль вимог, який був представлений замовником на папері; профіль вимог у ТЗ (специфікації), який був узгоджений з розробниками; профіль вимог, який був реалізований у програмному забезпеченні; профіль вимог, який був верифікований після реалізації програмного забезпечення. Було встановлено, що при переході від одного етапу до іншого, можливі певні відмінності у профілі вимог програмного забезпечення, так звані розриви. У зв'язку з цим, було сформовано повна множина ланцюжків відмінностей вимог у рамках життєвого циклу профілю вимог програмного забезпечення. Також були наведені приклади опису таких ланцюжків.*

Ключові слова: *життєвий цикл профілю вимог програмного забезпечення, профіль вимог програмного забезпечення, профілювання програмного забезпечення, процеси розроблення програмного забезпечення, ланцюжки розривів профілю вимог програмного забезпечення.*

https://doi.org/10.33108/visnyk_tntu2020.01.133 *Отримано 12.04.2020*