UDC 004.083.8:004.853 **doi: 10. 20998/2522-9052.2020.4.06**

Rustam Gamzayev, Mykola Tkachuk, Daria Shevkoplias

V. N. Karazin Kharkiv National University, Kharkiv, Ukraine

KNOWLEDGE-ORIENTED INFORMATION TECHNOLOGY TO VARIABILITY MANAGEMENT AT THE DOMAIN ANALYSIS STAGE IN SOFTWARE DEVELOPMENT

Abstract. The **subject matter** of this paper is a research of issues related to the variability management at the stage of domain analysis (DA) in the full life cycle (FLC) of software products line (SPL). The **main goal** of this research work is the elaboration of a new knowledge-based information technology to support a variability management in DA as a most complex and weak-formalized stage in FLC of SPL. In order to reach this goal the following **tasks** were formulated and resolved: to analyze the variability issues in FLC on the example of the modern agile-development approach - in the Scrum-methodology; to study how the methods of knowledge handling can be used to support some users and domain-experts activities within the DA phase with respect to software variability modeling; to make the motivated choice of the suitable CASE-tools to elaborate an appropriate IT solution to support the knowledge-oriented approach to DA; to present this IT-solution in a structured form, to consider some its implementation issues, and to discuss the first testing results. The **methods** used in this research are: domain-driven design approach to software development, repertory grids method and ontologies for expert's knowledge handling, IDEF0 notation for specification of the proposed IT solution, feature-oriented domain analysis (FODA) for variability modeling. **Conclusions:** the results of this research shown that the special attention has to be paid to the DA in a FLC, especially with usage of knowledge-based methods. To perform this process in an effective way the repertory grids method is motivated chosen and analyzed. To support the usage of this method in DA the proposal is made to elaborate the special IT-solution using some already available CASE-tools. The essentials functionality features of two such systems: GridSuite and SOVA (Semantical and Ontological Variability Analysis) are considered, and basing on this result, the integrated IT-solution is elaborated and presented in form of the IDEF0 diagram. Finally, the main technological facets of these tools installation are studied and tested, and the test-case to show the possibility to generate the FODA-variability model for the "Smart-Home" application domain is provided.

Keywords: knowledge; software; life cycle; domain analysis; variability; repertory grids method; ontology; information technology.

Introduction. Research actuality and aims

Nowadays in the domain of Software Engineering a growth number of new advanced methodologies, methods and tools are proposed, in order to produce new software products with lower cost, in shorter time, and with respect to their quality attributes [1]. Because of a lot of changes in user needs and in software and hardware components used in these processes developers are supposed to provide variable design solutions at all phases of their full life cycle (FLC) (see, e.g. in [2, 3]). Especially, these problems are really important and complex at such complex and weak-formalized stage of software FLC as a Domain Analysis (DA).

Even the **short review of some information sources** given in [4-9] has showed, that exactly to resolve these sophisticated issues so-called knowledge-oriented models and methods can be applied successfully (see in [10] for more details). To automate these approaches, it is also extremely important to propose appropriate information technologies (IT) which should be elaborated with usage of some already available CASE-tools to be integrated and configurated taking into account main specific activities included into the DA phase of FLC. Such advanced IT-solutions should allow to support a handling of expert knowledge in an appropriated problem domain in order to get finally the needed result in form of the correct domain model, e.g. as a FODA diagram, etc. [2, 9].

That is why the research topic presented in this paper is really actual, and the main tasks to be resolved in this way are the following:

- to analyze the variability issues in software FLC on the example of one of the modern and sophisticated agile-development approaches: on the Scrum-methodology;
- to provide the real example how the methods of knowledge processing can be used to support some users and domain-experts activities within the DA phase, with respect to software variability modeling;
- to make the motivated choice of the suitable CASE-tools to elaborate an appropriate IT – solution to support the knowledge-oriented approach to DA;
- to present this IT-solution in a structured form, consider some its implementation issues, and to discuss the first testing results.

Finally, in the last section the conclusions and future works are discussed.

Variability management in full life cycle of software development: general framework and the usage in Scrum methodology

The problems of software variability issues in FLC, mentioned above, are really important in the domain of Software Product Lines (SPL) development, which are the special kind of software-intensive applications built using a set of configurable or adaptable components [2, 6]. According to the reference framework for SPL engineering (SPLE) proposed by K. Pohl et al. in [2], there are 2 logical levels in all their FLC activities:

(I) *Domain Engineering Level* (DEL) that provides a reusable operational platform to define the *commonality* and the *variability* features of future SPL components;

(II) *Application Engineering Level* (AEL) that is responsible for the development of target SPL-applications based on the operational platform established at the DEL. Each of these 2 levels: the DEL and the AEL, has a similar sequence of the the ordered FLC phases: *Requirements Engineering*, *Domain Analysis, Domain Design, Architecture Design and Implementation*, and there are a lot of appropriate development methods and CASE-tools to support them in an effective way [2, 3].

In order to consider the specific technological facets of the unified SPLE – framework [2], we propose to make a logical mapping of its main FLC phases to the structured scheme one of wide-used software agile-development approaches, namely to Scrum method [7], and this mapping result is shown in Fig. 1. Taking into account these mapping results, it is to mention that the main SPLE phases are ordered into 2 logically interconnected FLC – subschemas, namely (see in Fig. 1):

1) the FLC phases: *Domain Analysis, Requirements Engineering* and *Domain Design* build the first subschema in FLC, which represents the Customer's needs to be realized technologically in a target SPL, and in terms of the Scrum-methodology they are reflected in such its artefacts as *Product Backlog* and *Sprint Backlog* respectively;

2) the FLC phases: *Architecture Design, Domain Implementation and Component Implementation* from the second subschema in the FLC, which reflects the Developer's activities to be performed in such Scrum – processes as *Daily Scrum Meeting (DSM)* and *Sprint iteration (S)*.

These 2 FLC subschemas are represented in Fig. 1 as the two large dashed ovals: the *Sub-FLC(1)* and *Sub-FLC(2)* shown in Fig. 1 respectively, and they are shown overlapped graphically, because from the project development point of view, their processes are interconnected technologically via activities *Domain Design* and *Architecture Design*. Besides that, the 3 small ovals depicted with solid lines and also shown in Fig. 1, represent the following development artifacts (functions): "Customer's needs", "Expert's knowledge" and "Developer's activities". Obviously, exactly the artifact "Expert's knowledge" is a semantical "bridge" between other two ones (this issue is depicted with 2 bold dashed lines between them). That is why the appropriate IT - solutions should be elaborated exactly to support the handling of expert's domain knowledge, in order to integrate in an effective way other main processes within the whole FLC of SPLE.

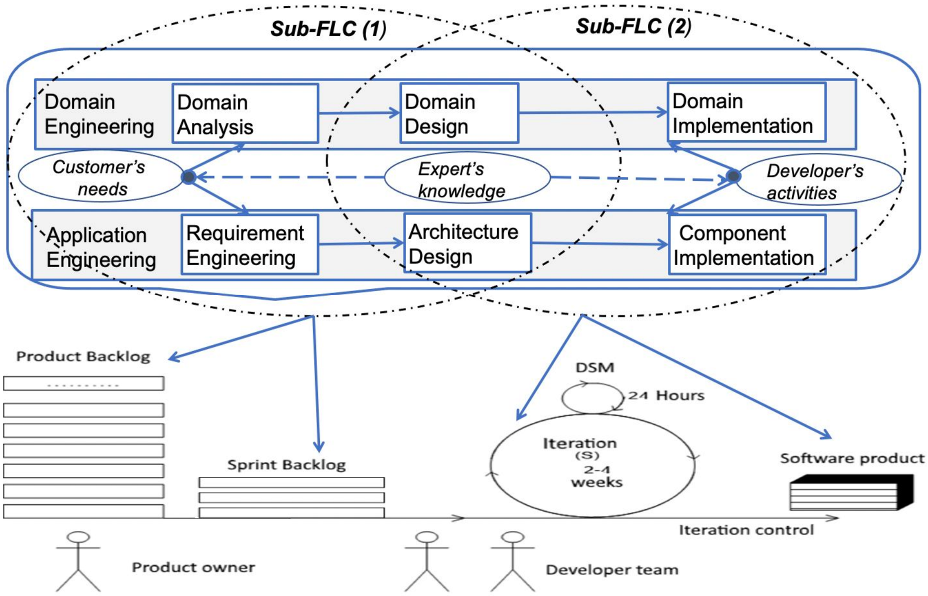


Fig. 1. The mapping results of the SPLE phases to the Scrum method activities (to compare with the SPLE - vision presented in [2])

An approach to handling of domain experts knowledge with repertory grid method

As it was already mentioned above, in [10] the recent overview of some methods for knowledge-oriented software development is presented, such as Decision Trees, Case-Based Reasoning, Heuristic-based learning and some others. From the point of view on multi-criteria and from an unified approach to its usage in different problem domains, the Repertory Grids (RG) method [13] was chosen for a further usage on expert's knowledge handling in FLC of SPLE. The main idea of this method supposes that a cognitive and collaborative thinking of experts and users in a given problem area can help to provide the domain analysis effectively, to decrease the software development costs, and finally to support an appropriate level of components variability in SPLE.

Any RG is represented as a matrix of the required dimensions (normally, not higher than 3x3), that reflects the different semantical contexts of a given domain (represented as the matrix columns), and the target software product characteristics (structured as matrix rows), which can be variable depending on the specific domain features. According to this vision, an appropriate RG includes 3 sets of basic items (*E, C, V*), and in [10] they are illustrated with the examples of the Smart-home systems (SHS) domain:

a) a set of *Elements (E)*: they can be defined based on the domain expert interviews which have to be divided into semantic objects to be represented as RG columns, (e.g.: *"Smart TV", "Smart light sensors"* ...);

b) a set of *Constructs (C):* they characterize the alternative variable features of RG elements, and they are placed in the RG rows depend on the appropriate context, (e.g. "*Smart TV should be turned on / turned off*"; "*Smart light sensors should be switch on / switch off*", ...);

c) a set of rating *Values (V)*: they are used to represent the behavior of the software system under the influence of certain elements in some range (usually from 1 to 5), where the lowest value of range represents the left bipolar construct, and the highest value of range the right bipolar construct, e.g. the element "*Smart TV* " *should be turned on* (the value of 1) *or turned off* (the value of 5), depend of a given context.

Taking into account these definitions (a) – (c), we can propose the following set-theoretic representation of the RG as a tuple:

$$RG = \{E, C, V\}, \tag{1}$$

where *E, C,* and *V* are the set of elements, set of constructs and set of rating values respectively.

The next advantage of the RG method is its adequate visualization of the expert knowledge handling. For example, in SHS these aspects (RG or dimensions) can be defined as [8]: 1) social aspects (any SHS has to be a user-friendly solution, 2) economical aspects (the project costs should be acceptable for SHS owners), 3) environmental aspects (any SHS should be an eco-friendly facility, etc.). Taking into account these 3 dimensions the appropriate problem space (PS) for the users and experts knowledge handling can be specified using RG method and the result is shown in Fig. 2.

To continue the formalization of the RG method, we can represent this space as a subset of the Cartesian product for the three sets defined in (1), namely:

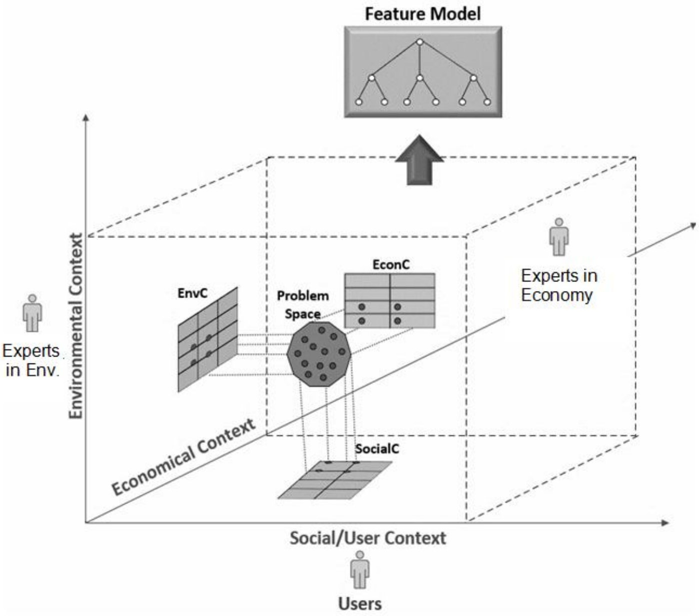


Fig. 2. The conceptual schema of RG method usage (based on [13])

$$PS(RG) \subseteq E \times C \times V \tag{2}$$

The further consideration of the RG method to handle the users and expert's knowledge on the example in SHS domain can be found for more details in [10].

Some CASE – tools for knowledge handling in domain analysis: GridSuite and SOVA systems in a nutshell

In addition to functional requirements elicitation, the RG methodology can provide an opportunity for analysis of experts' design approaches to system architecture. The RG technique can be used to explore the multidimensional space of the complex adaptive systems of the social-technical environment (e.g. "Smart-Home systems" (SHS) domain).

The whole process of variability analysis through the RG can be generalized into following steps [13], which are presented in Fig. 3.

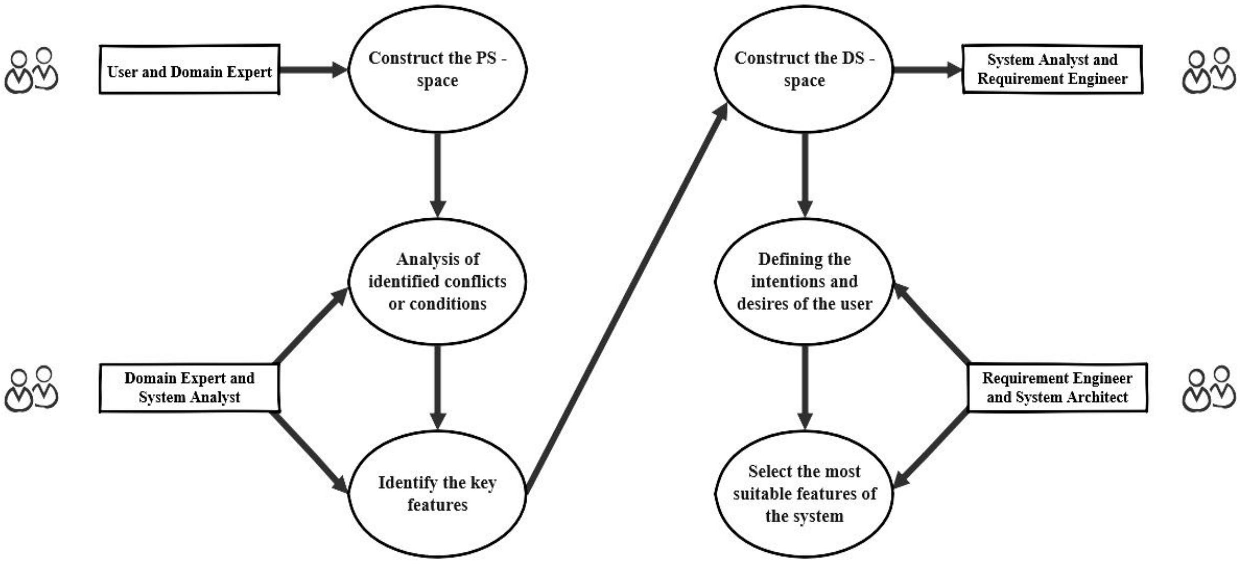

User and Domain Expert
Construct the PS - space
Analysis of identified conflicts or conditions
Domain Expert and System Analyst
Identify the key features
Construct the DS - space
System Analyst and Requirement Engineer
Defining the intentions and desires of the user
Requirement Engineer and System Architect
Select the most suitable features of the system

Fig. 3. Step-by-step process of variability modeling with RG method (to compare with [13])

Variability analysis process starts from construction of the problem space (PS) involving users and domain experts and then continues by analysis of conflicts or contradictions, identification of key features, construction of the design space (DS), definition of the intentions and user desires and selection of the most suitable features of the system involving the requirements engineers and designers through all the listed below stages.

Presented step-by-step variability analysis in the automated form can be implemented using the GridSuite tool [14] (CASE - tool) for elicitation, editing and analyzation of the interview-based RG. Comparing with other tools that are used to work with RG, GridSuite tool highlights user's role in RG creation as an interpreter of the interview data. Besides, GridSuite has additional build-in instruments for editing, analyzing and statistical calculations. The input data is a new or existing expert interview. The output data is an XML file with elements, functional system characteristics and their evaluations, or a created dendrogram, from which it's possible to elicit similarity, differences and variability indicators of elements/functional properties.

The main functionalities of the GridSuite tool are [15]:

- ability to present new or existing interviews as the inputs for further PS description;
- editing of one (adding, changing, deleting) or several RG's (merging grids);
- analysis of the obtained RG's by numerous methods (cluster analysis method, raw data and it's statistics method, main components analysis method and mouse sort method);
- parsing of the created RG's files (filters applying, reports creation, etc.).

The next step of experts' knowledge handling is the usage of the instrument, with help of which it is possible to build a FODA – model of domain space on the basis of the GridSuite initial data. This can be done with the SOVA (Semantic and Ontological Variability Analysis) tool [16] which is a CASE - tool combines semantical and ontological approaches to reflect system behavior rather than its implementation. The estimation of domain-focused behavior for existing or future system occurs by the representation of such system dynamical aspects in the form of ontology [17]. The appropriate semantic metrics are used to assess the similarity of the related system elements and to their variability analysis.

Fig. 4 represents the SOVA – tool functional properties for the facilitation of the ontology system creation, its behavior evaluation and similarity identification of the related elements, namely:

- text artifacts parsing using natural language processing methods (NLP) that consist of semantical role labeling, pronouns replacement and ontological model creation;
- similarity calculation of the analyzed behavior forms on the basis of entered into the system artifacts according to the criteria of initial conditions, external events and post-conditions;
- creation of a functional diagram on the hierarchical agglomerative clustering algorithm basis that represents system variability and composes the foundation of the domain model (DM).

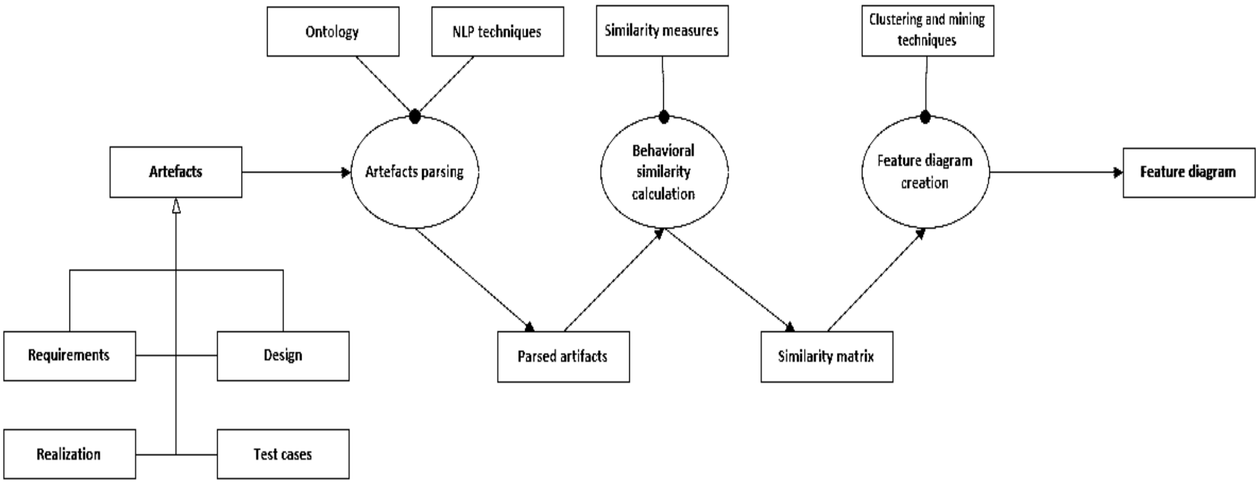


Fig. 4. SOVA tool basic functionality [17]

A technological scheme to implement the proposed approach

Based on the above-mentioned methodologies and CASE - tools, the knowledge elicitation process can be automated to increase efficiency of SPL development in the following way:

– construction of problem and domain space using experts (users) and domain experts (technical, social and economic) interviews;

– RG creation for certain aspects and contexts based on analysis of the constructed space;

– analysis of the obtained results, identification of similar, different, and variable components and functional characteristics;

– construction of variable FODA – model using semantic ontology approach and user-stories, received from the phase of knowledge elicitation, as a basis;

– applying of received feature-oriented domain analysis – model in XML format in the further development phases.

Fig. 5 shows the developed technological procedure in the form of IDEF0 notation diagram [18], which represents the whole automatization process of expert knowledge handling in the Domain Analysis phase of FLC.

The proposed procedure consists of 2 functional blocks (FB).

According to the rules of IDEF0 notation, each FB, namely A11 “Knowledge Elicitation” and A12 “FODA – model Construction”, has four interface types: “Input”, “Output”, “Control” and “Implementation mechanism”. This allows to represent the connection of all resources: inputs, models, methods, algorithms, experts and CASE – tools, that are needed for automatization of knowledge handling process.

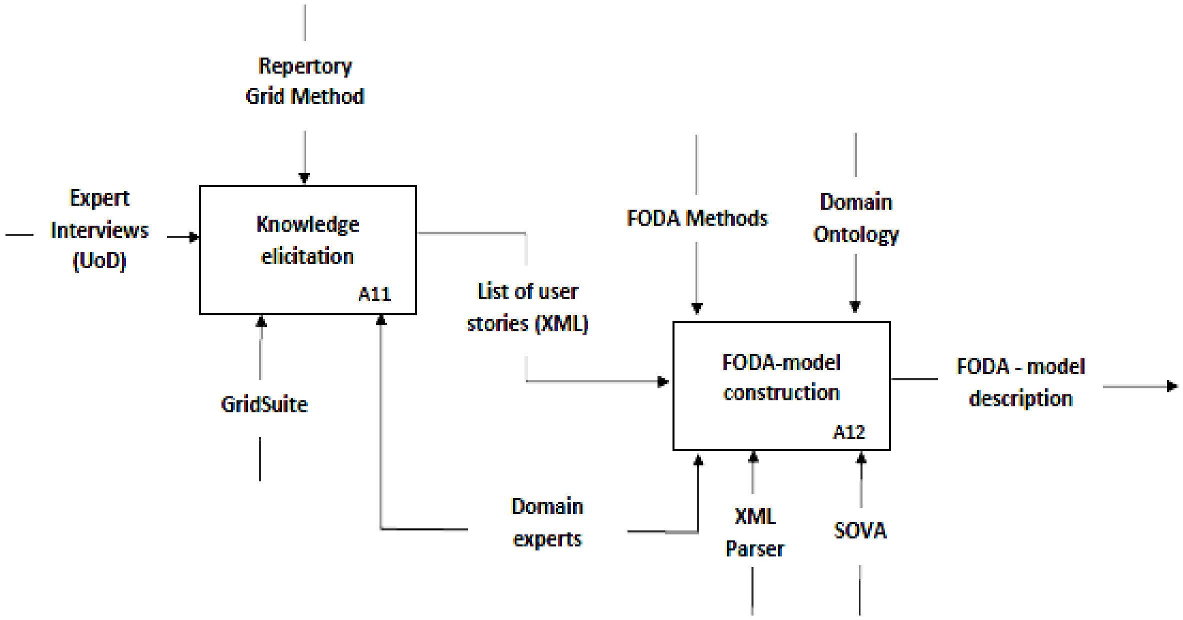


Fig. 5. The elaborated IT - solution for the combined usage of GridSuite and SOVA tools

The given scheme includes control interfaces and implementation interfaces. At the input of knowledge elicitation process, we receive experts’ interviews (UoD), which are generated using RG method, domain experts’ knowledge and GridSuite tool. As the output we get a list of user-stories [19] in XML format, that serves as an input for the next process of FODA – model construction. This process is carried out with the semantic ontology method and FODA methods, using SOVA tool for the variability graph creation, domain experts’ knowledge and XML – parser for input handling.

As the result, at the output of the last process we receive the description of FODA model in XML format, which can be used on the further phases of SPL development.

Tool installation issues and testing results in the development of “Smart – Home” systems

The full version of the GridSuite tool is not free, but for the basic functional properties testing it’s possible to install a trial demo-version, which provides the functionality for RG building, editing and complete analysis [15].

For the SOVA CASE – tool usage, the next steps have to be done:

- install the Protégé tool [20], which is a free ontological editor and open source framework for creating intelligent systems and building knowledge – based solutions;
- load additional PG ETI SOVA plug-in to work with SOVA tool [21];
- install additional HermiT OWL plug-in which is needed to determine, whether the created ontology is consistent or not and, also, to determine the connections between its elements, properties and axioms;
- activate the PG ETI SOVA that is a visualization tool for the construction of variability ontology graph with elements and their functional properties.

The RG construction begins with experts’ interviews handling.

The work will be demonstrated on a small PS dimension (4x4) for the “SHS” domain for the social context of the PS. To construct RG, an arbitrary range of values is needed to be selected within which, depending on the context, the need for functional properties is estimated.

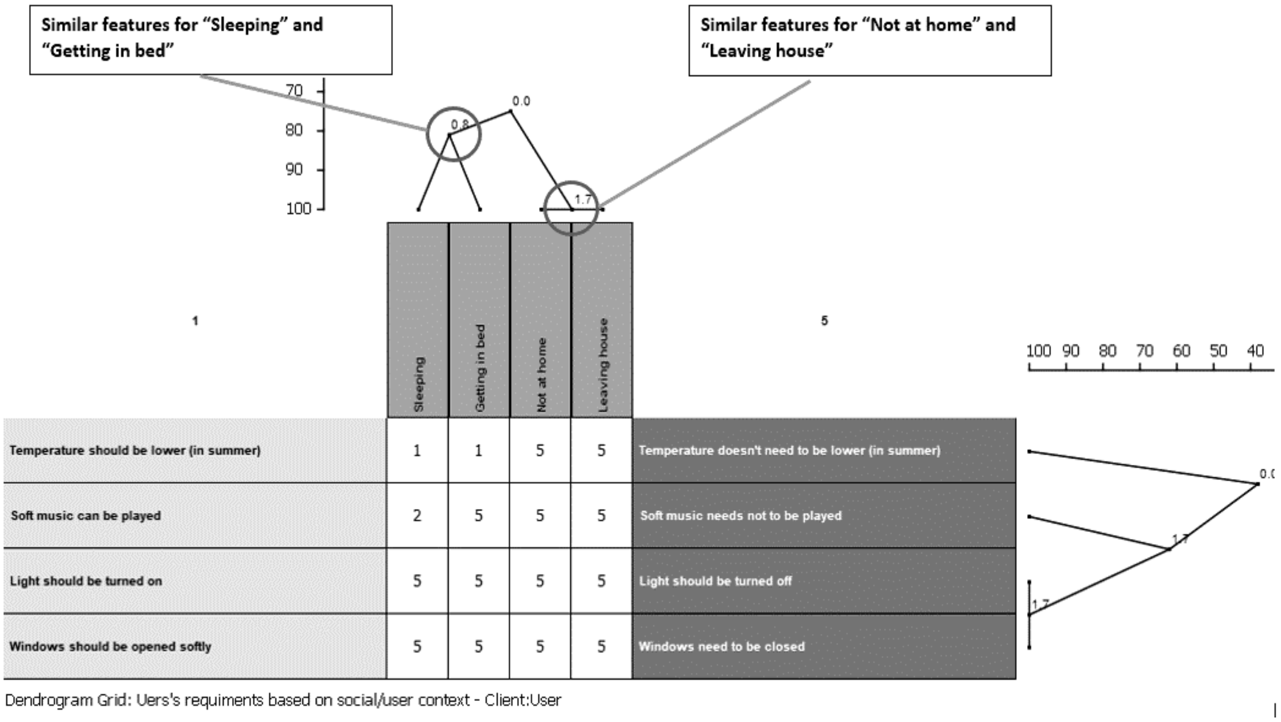
As a rule, values from 1 to 5 are selected, where 1 indicates that in the given context functionality is performed for the device, specified on the RG’s left side; 5 – for the device, specified on the RG’s right side. Values from 2 to 4 indicate the feature variability.

Besides, for the RG, elements of the corresponding context and constructs (certain functional devices’ properties of the SHS system) in the bipolar contradistinction form are needed to be specified.

After RG creation, it becomes possible to analyze the obtained data through the computer cluster analysis instrument that automatically changes the RG to eliminate contradictions and allows to inspect the similarities, differences and variability of elements and constructs visually and in percentage.

Fig. 6 shows the result of the RG cluster analysis in the PS based on the social context. Elements of the presented RG are the options of user activity, namely “Sleeping”, “Exercising”, “Leaving the house”, “Rest”, which in one or another way affect the need (or lack of it) for the certain usage of the devices’ functionality.

Constructs of the RG are the functional properties of the devices in the bipolar form, namely “TV should be turned on / TV should be turned off”, “Brightness of the light should be higher / Brightness of the light should be lower”, “Don’t notify about the movement inside / Notify about the movement inside”, “Increase the volume of audio speakers / Decrease the volume of audio speaker”.


Similar features for “Sleeping” and “Getting in bed”
Similar features for “Not at home” and “Leaving house”
70
80
90
100
0.0
0.8
1.7
1
5
100 90 80 70 60 50 40
0.0
1.7
1.7

	Sleeping	Getting in bed	Not at home	Leaving house	
Temperature should be lower (in summer)	1	1	5	5	Temperature doesn't need to be lower (in summer)
Soft music can be played	2	5	5	5	Soft music needs not to be played
Light should be turned on	5	5	5	5	Light should be turned off
Windows should be opened softly	5	5	5	5	Windows need to be closed

Dendrogram Grid: Uers's requiments based on social/user context - Client:User

Fig. 6. RG cluster analysis, built on the basis of social context for the PS

Analyzing the obtained results, it’s obvious that the “Sleeping” and “Getting in bed” contexts require the same or similar features (similarity point), since, based on the estimations for the given contexts, the features as “Temperature should be lower / Temperature doesn’t need to be lower” or “Light should be turned on / Light should be turned off” are invariable and have the only use in all conditions: for both contexts only features “Temperature should be lower” and “Light should be turned off” are needed. As for the “Soft music can be played / Soft music needs not to be played” feature in the context of “Sleeping”, the estimation of two means the variability of the feature as its usage depends on every user’s needs and desires. Alike RGs also can be constructed and analyzed for the economic and environmental contexts. As the output, in GridSuite tool we get built RGs with all elements and constructs as a list of user stories in the XML format.

On their basis or on the analysis of the dendrograms and percentages of the devices’ similarity/ differences/ variability, it becomes possible to build a variable FODA – model for the “SHS” domain using the SOVA tool. The ontology in the above-mentioned tool is built as follows:

- creation of the classes and subclasses representing RG elements;
- creation of the properties and connections between classes representing RG constructs.

Fig. 7 shows a variable FODA – model built with the SOVA tool on the basis of GridSute tool results.

The presented ontology graph elements have the following semantical meaning [20, 22]:

- the round node T is a “Thing” class or a root class, from which we start to build the target ontology. The “Thing” class defines that each subclass under it is represented as the real-world entity;
- the rectangular elements like as “Nighttime”, “Lamp” or “Comfort” are the model entities (or ontology classes), that represent all RG elements or conditions, and they have logical connections between each other;
- the rectangular elements like as “isUsedCondition” or “is A” represent of the property existence, and they define if the appropriate entity is a RG element or a condition;
- the directional arrows show the relationship between model entities (generalization, composition, aggregation, etc.).

As the output from the SOVA tool we get a visualized variable ontology graph and file in the XML format with the constructed variable FODA – model description. Fig. 8 represents the obtained result with the use of the SOVA tool in the form of a description that can be used for the further stages of SPL development.

E.g., in [9] is already shown, that basing on a given FODA model it is possible to generate a target source code framework with a predicted reusability coefficient value, and this approach is exactly useful for SPL development.

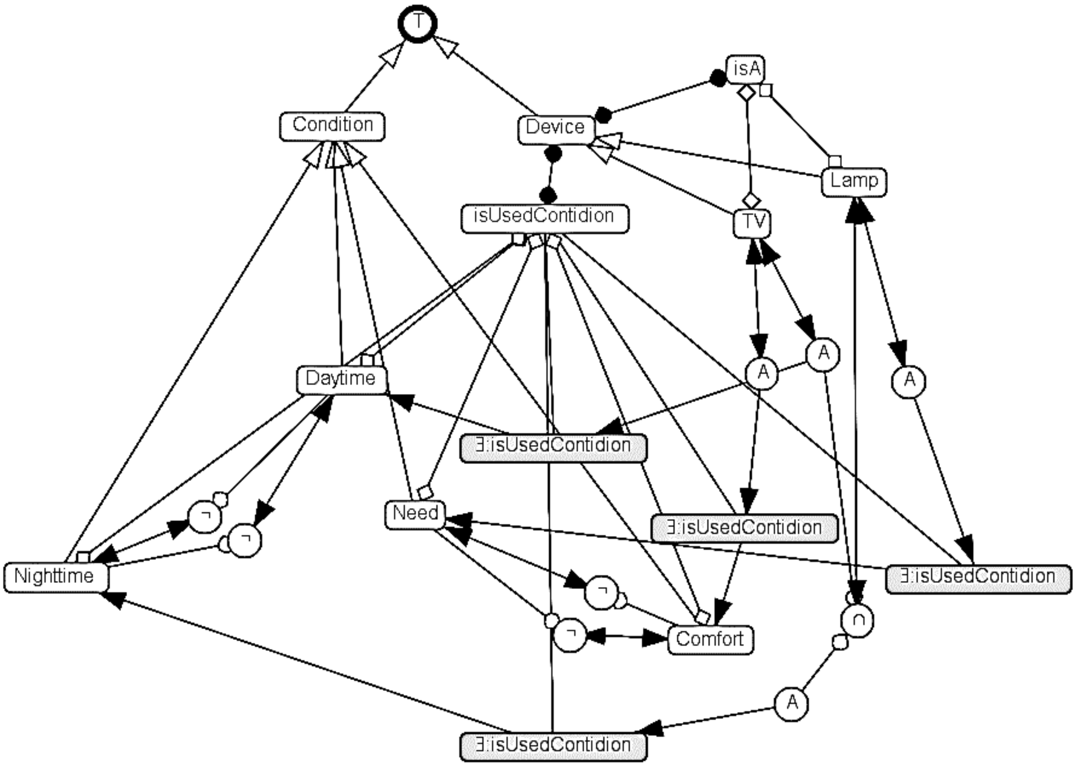
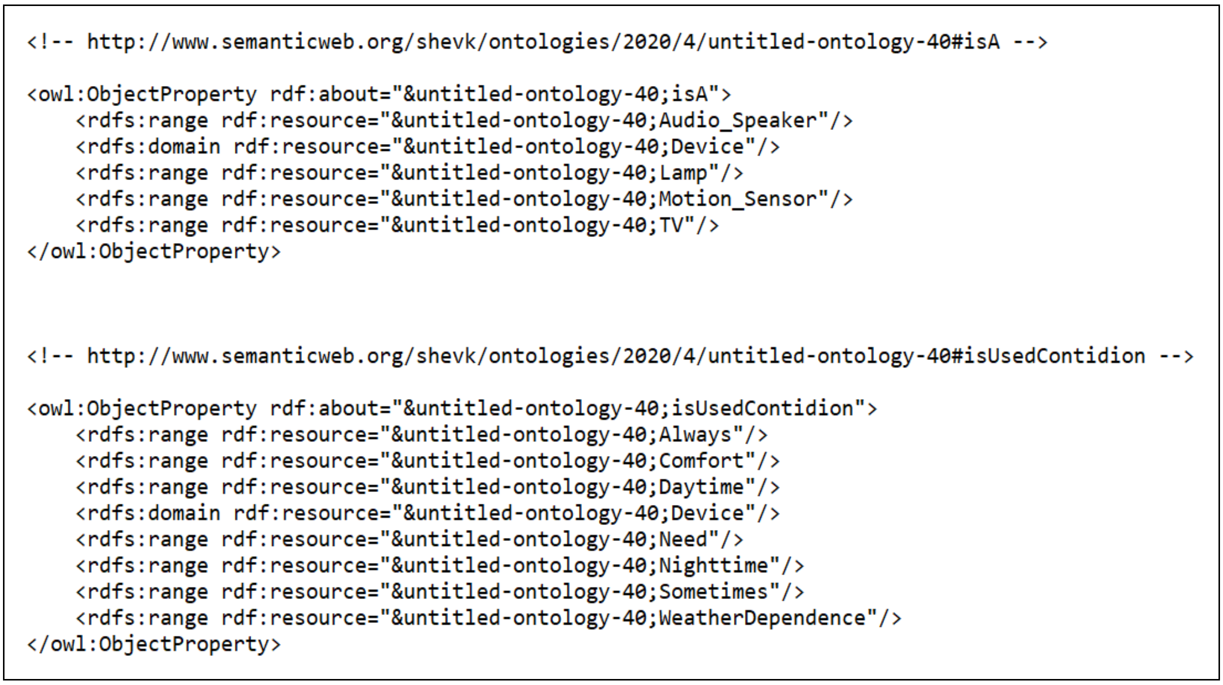


Fig. 7. Variable FODA – model for the “Smart – Home system” domain


```
<!-- http://www.semanticweb.org/shevk/ontologies/2020/4/untitled-ontology-40#isA -->

<owl:ObjectProperty rdf:about="&untitled-ontology-40;isA">
    <rdfs:range rdf:resource="&untitled-ontology-40;Audio_Speaker"/>
    <rdfs:domain rdf:resource="&untitled-ontology-40;Device"/>
    <rdfs:range rdf:resource="&untitled-ontology-40;Lamp"/>
    <rdfs:range rdf:resource="&untitled-ontology-40;Motion_Sensor"/>
    <rdfs:range rdf:resource="&untitled-ontology-40;TV"/>
</owl:ObjectProperty>



<!-- http://www.semanticweb.org/shevk/ontologies/2020/4/untitled-ontology-40#isUsedContidion -->

<owl:ObjectProperty rdf:about="&untitled-ontology-40;isUsedContidion">
    <rdfs:range rdf:resource="&untitled-ontology-40;Always"/>
    <rdfs:range rdf:resource="&untitled-ontology-40;Comfort"/>
    <rdfs:range rdf:resource="&untitled-ontology-40;Daytime"/>
    <rdfs:domain rdf:resource="&untitled-ontology-40;Device"/>
    <rdfs:range rdf:resource="&untitled-ontology-40;Need"/>
    <rdfs:range rdf:resource="&untitled-ontology-40;Nighttime"/>
    <rdfs:range rdf:resource="&untitled-ontology-40;Sometimes"/>
    <rdfs:range rdf:resource="&untitled-ontology-40;WeatherDependence"/>
</owl:ObjectProperty>
```

Fig. 8. FODA – model description in the XML format

Conclusion and future work

In this research the actual scientific and technical problem: the development of software product lines (SPL) with respect to variability issues at all stages of their full life cycle (FLC) is considered, especially using the recognized framework for FLC of SPL proposed by K. Pohl. To analyze this problem with respect to the both modeling and technological issues, the logical mapping of the main FLC phases to the structured scheme one of wide-used software agile-development, namely to the Scrum method is done.

As the result of this mapping approach the conclusion is made, that the special attention has to be paid to such most complex and weak-formalized stages in the FLC as a domain analysis (DA) stage. It is mentioned, that to perform this process in an effective way some methods and tools of expert knowledge handling can be used, and in this research the repertory grids method is motivated chosen. To support the usage of this method in DA the proposal is made to elaborate the special IT-solution using some already available CASE-tools. The essentials functionality features of two such systems: GridSuite and SOVA (Semantical and Ontological Variability Analysis) are considered, and basing on this result, the integrated IT-solution is elaborated and presented in form of the IDEF0 diagram. Finally, the main technological facets of these tools installation are studied and tested, and the test-case to show the possibility to generate the FODA-variability model for the "Smart-home" application domain is provided.

As the next step to be done in our research we are going to develop the special software components to apply a collection of quantitative metrics for quality assessment of the generated domain models.

References

1. Sommerville, I. (2015), *Software Engineering,* 10th edition, Addison-Wesley, NY.
2. Pohl, Klaus, Böckle, Günter, van der Linden, Frank J. (2005), *Software Product Line Engineering: Foundations, Principles, and Techniques*, Springer, 467 p.
3. Lavrishcheva, K. M., Koval, G.I., Babenko, L.P., Slabospytska, O.O. and Ignatenko, P.P. (2011), *New theoretical bases of technology production of software systems families in the context of generating programming:* monograph, Institute Program Systems, NAS Ukraine, Kyiv, 277 p. (in Ukrainian).
4. Tiihonen, Juha, Raatikainen, Mikko, Myllärniemi, Varvana and Männistö, Tomi (2016), "Carrying Ideas from Knowledge-based Configuration to Software Product Lines", *Proceedings of the 15th International Conference on Software Reuse*, Cyprus, June 5-7, , pp. 55–62.
5. Methods of knowledge acquisition automatization in expert systems: classification, current state, comparative analysis, *International scientific-practical journal "Software products and systems"*, available to: http://www.swsys.ru/index.php?page= article &id=1187&lang=.docs (in Russian).
6. Capilla, R., Bosch, J. and Kang, K. (2013), *Systems and Software Variability Management*, Springer, Berlin.
7. Karagiannis, D., Mayr, H.C. and Mylopoulos, J. (2016), *Domain-Specific Conceptual Modeling: Concepts, Methods and Tools*, Springer, Berlin.
8. Tune, N. and Millet, S. (2015), *Patterns, Principles And Practices Of Domain-driven Design*, 1st ed., John Wiley & Sons.
9. Tkachuk M., Gamzaev R., Martinkus I., Sokol, V. and Tovstokorenko, O. (2018), "Towards Effectiveness Assessment of Domain Modelling Methods and Tools in Software Product Lines Development", *Enterprise Modelling and Information Systems Architectures – International Journal of Conceptual Modeling*, Vol. 13, Germany, pp. 190-206.
10. Gamzayev, R.O., Tkachuk, M.V. and Shevkoplias D.O. (2020), "Handling of Expert Knowledge in Software Product Lines Development with Usage of Repertory Grids Method", *Bulletin of the V.N. Karazin Kharkiv National University: 'Mathematical Modeling. Information Technologies. Automated Control Systems*, No. 47, pp. 13-24.
11. Eleutério, J. (2017), *A Comparative Study of Dynamic Software Product Line Solutions for Building Self-Adaptive Systems*, 30 p.
12. Kumar, Manish and Dwivedi, R.K. (2020), *Applicability of Scrum Methods in Software Development Process*, available at: https://ssrn.com/abstract=3610759 or http://dx.doi.org/10.2139/ssrn.3610759.
13. Sangeeta, D. and Seok-Won, L. (2015), From Requirements Elicitation to Variability Analysis Using Repertory Grid: A Cognitive Approach, *Proceeding of RE*, Ottawa, ON, Canada, pp. 46-55.
14. (2020), *GridSuite*, Repertory Grid Software, 2003-2020, available at: https://www.gridsuite.de/45623/41431.html.
15. Fromm, M. (2017), *Manual for GridSuite 4 and 4+*, 54 p.
16. Thorsten, Berger, Marsha, Chechik, Timo, Kehrer and Manuel, Wimmer (2019), "Software evolution in time and space: unifying version and variability management", *Report from Dagstuhl Seminar*, 19191, May 5-10, p. 30.
17. Itzik, N. and Reinhartz-Berger, I. (2014), "SOVA - A Tool for Semantic and Ontological Variability Analysis", *CAiSE'14 Forum*, CEUR Workshop Proceedings 1164, pp. 177-184.
18. Lavrishcheva, K. M. (2008), *Software Engineering*, IPU, Kyiv, 319 p., available at:
URL: http://csc.knu.ua/uk/library/books/lavrishcheva-6.pdf (in Ukrainian).
19. Lucassen, G., Dalpiaz, F., Martijn, J. and Brinkkemper S. (2016), "Improving Agile Requirements: the Quality User Story Framework and Tool", *Requirements Engineering*, Vol. 21 pp. 383-403.
20. (2020), *Protégé System*, Stanford University, available at: https://protege.stanford.edu/products.php#desktop-protege.
21. (2020), *Simple Ontology Visualization API*, Jean-Philippe Lang, 2006-2013, available at: https://kask.eti.pg.gda.pl/redmine/projects/sova/wiki/Main_Page.
22. (2020), *SOVA-Visualization symbols*, Appendix C, available at: https://protegewiki.stanford.edu/wiki/File:SOVA-Symbols.pdf.

Received (Надійшла) 10.09.2020
Accepted for publication (Прийнята до друку) 04.11.2020

Відомості про авторів / About the Authors

Гамзаєв Рустам Олександрович – кандидат технічних наук, доцент, докторант кафедри моделювання систем і технологій, Харківський національний університет імені В. Н. Каразіна, Харків, Україна;

Rustam Gamzayev – PhD, Asc. Professor, Doctoral Student of the Department of Systems and Technology Modeling, V.N. Karazin Kharkiv National University, Kharkiv, Ukraine;
e-mail: rustam.gamzayev@gmail.com; ORCID ID: https://orcid.org/0000-0002-2713-5664.

Ткачук Микола Вячеславович – доктор технічних наук, професор, завідувач кафедри моделювання систем і технологій, Харківський національний університет імені В. Н. Каразіна, Харків, Україна;
Mykola Tkachuk – Dc. of Techn. Science, Professor, Head of the Department of Systems and Technology Modeling, V.N. Karazin Kharkiv National University, Kharkiv, Ukraine;
e-mail: mykola.tkachuk@karazin.ua; ORCID ID: https://orcid.org/0000-0003-0852-1081.

Шевкопляс Дар'я Олександрівна – студентка кафедри моделювання систем і технологій, Харківський національний університет імені В. Н. Каразіна, Харків, Україна;
Daria Shevkoplias – student of the Department of Systems and Technology Modeling, V. N. Karazin Kharkiv National University, Kharkiv, Ukraine;
e-mail: shevkoplysdarya23@gmail.com; ORCID ID: https://orcid.org/0000-0001-5870-771X.

Знання-орієнтована інформаційна технологія для управління варіабельністю на етапі доменного аналізу у розробці програмного забезпечення

Р. О. Гамзаєв, М. В. Ткачук, Д. О. Шевкопляс

Анотація. Предметом даної роботи є дослідження, пов'язаних з управлінням варіабельністю на етапі доменного аналізу (ДА) у повному життєвому циклі (ПЖЦ) лінійки програмних продуктів (ЛПП). **Основною метою** цієї дослідницької роботи є розробка нової інформаційної технології, заснованої на знаннях, для підтримки управління варіабельністю в ДА як у найскладнішому та слабко-формалізованому етапі в ПЖЦ ЛПП. Для досягнення цієї мети були сформульовані та вирішені **наступні завдання**: проаналізувати проблеми варіабельності у ПЖЦ на прикладі сучасного підходу до гнучкої розробки - у методології Scrum; вивчити, як методи обробки знань можуть бути використані для підтримки діяльності користувачів та експертів доменів на етапі ДА з урахуванням моделювання варіабельності програмного забезпечення; зробити мотивований вибір CASE-інструментів для розробки відповідного ІТ-рішення для підтримки знання-орієнтованого в ДА; представити це ІТ-рішення у структурованій формі, розглянути деякі питання його впровадження та обговорити перші результати тестування. **Методи**, що використовуються в цьому дослідженні: проблемно-орієнтований підхід до розробки програмного забезпечення; метод репертуарних сіток та онтології для обробки знань експертів, нотація IDEF0 для специфікації запропонованого ІТ-рішення, орієнтований на функції доменний аналіз (FODA) для моделювання варіабельності в ЛПП. **Висновки:** результати цього дослідження показали, що особлива увага повинна бути приділена ДА в ПЖЦ, зокрема, із застосуванням знання-орієнтованих методів. Для ефективного проведення цього процесу мотивовано обрано та проаналізовано переваги методу репертуарних сіток для обробки експертних знань. Для автоматизації застосування цього методу в ДА запропоновано розробити відповідне ІТ-рішення, використовуючи деякі вже доступні CASE-інструменти. Розглянуто основні особливості функціональності двох таких систем: GridSuite та SOVA (семантичний та онтологічний аналіз змінності), і на основі цього результату розроблене та представлене інтегроване ІТ-рішення у формі діаграми IDEF0. Розглянуто основні технологічні аспекти встановлення цих інструментів, а також надається тестовий приклад, який показує можливість генерування варіабельної FODA моделі для застосування в розробках систем «Розумний дім».

Ключові слова: знання, програмне забезпечення; життєвий цикл; доменний аналіз; варіабельність; метод репертуарних сіток; онтологія; інформаційна технологія.

Знание-ориентированная информационная технология для управления вариабельностью на этапе доменного анализа в разработке программного оюеспечения

Р. А. Гамзаев, Н. В. Ткачук, Д. А. Шевкопляс

Аннотация. Предметом данной работы является исследование вопросов, связанных с управлением вариабельностью на этапе доменного анализа (ДА) в полном жизненном цикле (ПЖЦ) линейки программных продуктов (ЛПП). **Основной целью** этой исследовательской работы является разработка новой информационной технологии, основанной на знаниях, для поддержки управления вариабельностью в ДА как в сложном и слабо-формализованном этапе в ПЖЦ ЛПП. Для достижения этой цели были сформулированы и решены **следующие задачи**: проанализировать проблемы вариабельности в ПЖЦ на примере современного подхода к гибкой разработки - в методологии Scrum; изучить, как методы обработки знаний могут быть использованы для поддержки деятельности пользователей и экспертов доменов на этапе ДА с учетом моделирования вариабельности программного обеспечения; сделать мотивированный выбор CASE-инструментов для разработки соответствующего ИТ-решения для поддержки знания-ориентированного в ДА; представить это ИТ-решение в структурированной форме, рассмотреть некоторые вопросы его внедрения и обсудить первые результаты тестирования. **Методы**, используемые в этом исследовании: проблемно-ориентированный подход к разработке программного обеспечения; метод репертуарных сетей и онтологии для обработки знаний экспертов, нотация IDEF0 для спецификации предложенного ИТ-решения, ориентированный на функции доменный анализ (FODA) для моделирования вариабельности в ЛПП. **Выводы:** результаты этого исследования показали, что особое внимание должно быть уделено ДА в ПЖЦ, в частности, с использованием знание-ориентированных методов. Для эффективного проведения этого процесса мотивировано избран и проанализированы преимущества метода репертуарных сетей для обработки экспертных знаний. Для автоматизации применения этого метода в ДА предложено разработать соответствующее ИТ-решения, используя некоторые уже доступны CASE-инструменты. Рассмотрены основные особенности функциональности двух таких систем: GridSuite и SOVA (семантический и онтологический анализ сменности), и на основе этого результата разработанное и представлено интегрированное ИТ-решение в форме диаграммы IDEF0. Рассмотрены основные технологические аспекты установления этих инструментов, а также предоставляется тестовый пример, показывающий возможность генерирования вариабельной FODA модели для применения в разработках систем «Умный дом».

Ключевые слова: знания; программное обеспечение; жизненный цикл; доменный анализ; вариабельность; метод репертуарных решеток; онтология; информационная технология.