Towards an Approach to Formulating Personal Development Plan for Developers based on Competency Framework and Data Mining

Erchimen Gavriliev[1[0000-0001-7289-3969]] and Tatiana Avdeenko[1[0000-0002-8614-5934]]

[1] Novosibirsk State Technical University, K. Marksa ave., 20, 630073, Novosibirsk, Russia
erchimen_gavriliev@outlook.com

Abstract. Companies face difficulties in managing employees' professional growth due to the large variety of career paths available in the IT industry, as well as their high degree of flexibility and unpredictability. In this regard, a personal development plan is used to manage the career advancement of developers. However, managers struggle to create plans that align with employees' professional aspirations. A competency framework was developed to increase relevancy of the plans by providing a comprehensive view of possible career paths for developers and required competencies. The framework was implemented in a software development company to assess its validity. It was discovered that a significant proportion of developers had competencies related to communication, project teamwork, requirements analysis, as well as to solving technical tasks such as programming and debugging.

Keywords: software developer, competency matrix, competency model, data mining, natural language processing, personal development plan

1 Introduction

Human resources and professional development system determine organization's effectiveness. However, recent economic, technological, and social advances have resulted in considerable changes in organizational processes and job profiles, complicating the management process of employees' professional growth [1].

For instance, work environment, responsibilities, and skill requirements for IT specialists have changed over the years, which have fundamentally impacted many career paths, making them more flexible and unpredictable [2,3].

Personal development plans (PDPs) are a frequent option in IT organizations; they are meant to stimulate employee training, which in turn should increase staff performance while taking into consideration their personal professional aspirations [4].

Nevertheless, development and implementation of PDP are complex tasks, which is why only a small portion of tasks listed in the plans are completed, or implemented plans have little impact on professional growth [5,6]. Managers frequently struggle with PDP development: they are usually interested in using available specialists that already have expertise for each specific project and each position in it. However, these specialists may also prefer to obtain other skills and/or work in other projects, but these preferences are not stated in their PDPs.

One solution is to use a competency model, which provides a holistic view of different career paths and required competencies for each grade – a generalized position that unifies a group of professions.

However, existing studies mainly focus on programming-related competencies such as coding, knowledge of mathematical and engineering foundations for software development [7]. There are studies that consider soft competences such as team communication skills, planning, analytical skills and problem-solving abilities as well [8]. But focusing only on either technical or soft competencies can lead to problems in professional development, because software development involves different tasks: gathering and analyzing requirements, testing, mentoring and etc.

Furthermore, existing studies that provide a more comprehensive view of developer competencies do not specify approaches or indicators for assessing them. Activities in software development produce large number of artefacts, which can be analyzed using data mining methods and tools to gather information about employees' competencies.

The aim of this work is to increase the relevance of PDPs for developers by proposing a competency framework and method for assessing competencies based on the data mining of systems in which developers works on a regular basis.

In the rest of this paper, Section 2 gives background on publications related to developers' competencies. Section 3 presents the developed competency framework, methodology for its validation and the case company. Section 4 contains validation results, and Section 5 concludes the paper.

2 Literature Review

Personal development plan is a tool used in the career management cycle to collect and document information about the competencies that an employee has previously acquired, as well as about the competences that he (she) is planning to develop in the future. PDP includes following sections with posed questions [8]:

- main objectives – what competencies does an employee need to develop?
- list of tasks that an employee must complete, as well as deadlines– how does he (she) want to develop those competencies and in which timeframe?
- possible risks and list of employees, that can assist in completing the tasks – which support is needed?

Competency is important in professional development, but definitions vary [10]. We will adopt a performance-based approach in this study and define competency as an ability and willingness to apply knowledge and skills in solving professional problems in various fields [11].

Competency model is a hierarchy of competencies expected of an employee that contains descriptions of competencies as well as measurable or observable indicators that may be used to evaluate workers [12].

Relevant research on necessary competencies for software developers usually focuses on technical competencies, such as programming fundamentals and software design [13, 14]. This list of competencies also includes debugging [15] and abilities to analyze source code [16], because it is frequently required to make modifications to an already developed system.

There are also publications that mainly consider soft competencies related to teamwork and communications [8, 17]. Other works discovered attributes and characteristics, that are not related to coding, such as flexibility, proactivity¸ willingness to learn and independence [18].

Nonetheless, there are studies that provide an extensive view of developers' competencies, taking into account both technical and soft competencies.

Microsoft employees were polled about the attributes of a «great» software developer [19]. Based on the survey results, 54 attributes were found and divided into four groups: «Personality», «Decision-making», «Teammate Interactions» and «Software product». Attributes associated with software development, as well as the developer's individual characteristics were given the highest level of importance, such as paying attention to the quality of the source code, having necessary intellectual abilities to solve complex problems.

Based on a survey of 355 software developers and the literature on expertise and productivity, paper provides a conceptual theory of software development expertise (SDExp) [20]. It was discovered that developer's effectiveness is determined by one's level of expertise and experience, as well as a number of individual characteristics, such as openness and agreeableness.

Prior work on the methods for assessing developers' competencies is mainly focused on programming as well [21]. However, because developers have diversified their actives in other areas such as gathering and analyzing requirements, testing, mentoring and etc., they may not be successful in assessing their qualifications as a whole. Due to the low level of reliability and accuracy, another frequently used method based on the subjective manager's opinion may yield inaccurate results.

Data mining of repositories has been applied in several studies to measure the expertise of IT specialists [22]. It was argued that a specialist who has made a significant contribution to the project's repository may have a greater level of «quality».

Based on literature review, it can be concluded that assessment method based on systems' data mining has a higher degree of objectivity than the manager's opinion, because the former method is based on quantitatively measurable indicators and relevant data.

3 Development and Validation of Competency Framework

3.1 Description of Competency Framework

First version of developer competency framework was designed using the findings of prior studies [23] and literature review. It was a two-dimensional matrix that represented core expertise domains of software development. The first dimension of the matrix was defined by 3 domains:

1. «Tasks' complexity and approach to solving them», which included abilities to structure one's own approach to work, ability to understand complex processes, systems and ability to apply knowledge and skills to new situations.
2. «Interaction skills and collaboration with a project team», which included communicative competencies, competencies for professional collaboration.
3. «Development skills», which included knowledge of programming languages, software architectures, internet protocols, database management systems and etc.

The second dimension represented developmental process and expected changes in competency across four common grades:

1. «Junior» – entry-level developer.
2. «Middle» – employee with specialized knowledge and a broader range of responsibilities.
3. «Senior» – experienced specialist with advanced technical competencies.
4. «Lead» – experienced specialist with advanced technical and managerial competencies.

In order to refine and test competency framework, it was implemented in a medium-sized company that develops software for the banking industry. The company uses its own low-code application builder to develop software products. Tech stack of the application builder included:

- relational database (Oracle or PostgreSQL) as a data storage;
- Java application server (Payara or WildFly) as a web application server;
- Rhino (an engine that converts JavaScript code into Java classes) and United Data Model Script (an application builder's programming language) were two programming languages used in backend development;
- JavaScript and a built-in interface editor were used for frontend development.

To create a software product, the company's developers had to understand this tech stack. A number of interviews were conducted with lead developers and project managers in order to further refine the framework's set of competences. A new domain «Other technical skills» was added to the first dimension of matrix, because in the case company developers usually applied tools and technologies only from the mentioned above tech stack, and knowledge of other tools outside the tech stack was optional, but it could be useful in development. For example, developer could also work as a DevOps engineer, which included setting up tools and appropriate infrastructure, as well as selecting and deploying CI/CD solutions. As the number of competencies grew, it was decided to categorize them into subdomains to improve clarity. New version of framework consisted of 46 competencies and included following domains and subdomains:

1. «Tasks' complexity and approach to solving them»:
   a. «Task formulation» – ability to solve tasks with different levels of details in formulation.

   b. «Task management» – skills of administering tasks assigned to developer themselves, as well as to other members of project team. Ability to plan their time, set up schedules, and complete tasks on time.
   c. «Types of tasks» – characteristics of employee's activities.
   d. «Level of independence in decision making» – characteristics of decisions made regarding software product in development. Ability to accept responsibilities and solve problems individually.
   e. «Task management system» – ability to work and keep records in a task management system.
2. «Interaction skills and collaboration with a project team»:
   a. «Project team» – characteristics of communication and collaboration skills with a project team.
   b. «Client» – characteristics of communication and collaboration skills with a client or other vendors. Understanding the goals of project and its structure from different points of view.
   c. «Mentoring» –capabilities of teaching and adapting new employees, of giving feedback.
3. «Development skills»:
   a. «Technology stack» –skills and knowledge of the applied technology stack for development:
      (1) «Knowledge of UDMS».
      (2) «Knowledge of JavaScript».
      (3) «Knowledge of SQL».
      (4) «Debugging skills».
      (5) «Knowledge of application builder’s instruments for developing».
      (6) «UI development skills».
      (7) «Knowledge of application builder’s architecture».
   b. «Documentation» –level of documentation knowledge for an applied technology stack.
   c. «Integration with external systems» – knowledge and skills of setting up integration with external systems through various protocols:
      (1) HTTP.
      (2) JDBC.
      (3) SOAP.
      (4) LDAP.
      (5) JMS.
4. «Other technical skills»:
   a. «Infrastructure» – skills and knowledge of server administration:
      (1) «Knowledge of Linux operating systems and command line interface».
      (2) «Knowledge of administering Java application servers».
      (3) «Knowledge of administering Database Management Systems».
      (4) «Knowledge of tools for monitoring IT-infrastructure».
   b. «Particular technologies» –skills and knowledge of technologies and tools used in development: Git, ELK, Hazelcast etc.

A higher-level grade might include competencies from the previous grade. For example, subdomains «Task formulation» and «Technology Stack» from the developed framework are presented in Table 1.

Table 1 Fragment of the developer competencies framework

Domain	**Subdomain**	**Junior**	**Middle**	**Senior**	**Lead**
Tasks' complexity and approach to solving them	Task formulation	Can work with a detailed task formulation	Can work with a short task formulation Can solve task with a medium-detailed formulation individually with high quality and on time	Can work with a task without detailed formulation Can identify and formulate a task for himself	see Senior
Development skills	Technology stack	Basic proficiency in programming language Familiar with code writing culture in company	Intermediate proficiency in programming language Familiar with refactoring techniques Can build basic unit tests	Fluent in programming language Can build complex unit tests Can optimize code	see Senior

3.2 Validation Methodology

Activities in software development produce large number of artefacts, that can be processed with data mining methods and tools, which in turn can provide information about employees' competencies.

Developers use version control system, task management system and knowledge management system in their daily work. Version control system records changes in application' source code. Task management systems are used to organize the work of the project team. Knowledge management system helps in organizing processes of creating, storing and transferring knowledge.

Data mining techniques were used to process information from these systems in order to assess employees' competencies and grade, and also to test and refine framework. For extracting and analyzing data, previously developed decision support system (DSS) was modified. DSS consisted of two subsystems: data download subsystem and competency assessment subsystem [24].

Fig. 1 shows structural and functional model of the data download subsystem. At the end of the day, the subsystem migrated information from external systems via REST-services. Information about tasks, comments and work logs were downloaded from Atlassian Jira (task management system), data about pages, articles and user activity were extracted from Atlassian Confluence (knowledge management system), and source code of applications were downloaded from multiple application builder repositories (version control system).

Additionally, subsystem performed data transformation, i.e., replacing logical names with physical ones in SQL queries, translating UDML code to JavaScript code, constructing abstract syntax trees for SQL queries and JavaScript code, analyzing software complexity of constructed syntax trees and etc.

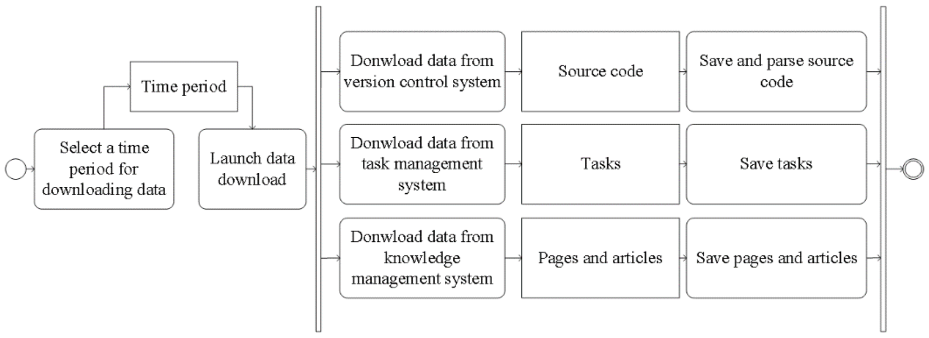

Time period
Select a time period for downloading data
Launch data download
Donwload data from version control system
Source code
Save and parse source code
Donwload data from task management system
Tasks
Save tasks
Donwload data from knowledge management system
Pages and articles
Save pages and articles

Fig. 1. Structural and functional model of the data download subsystem

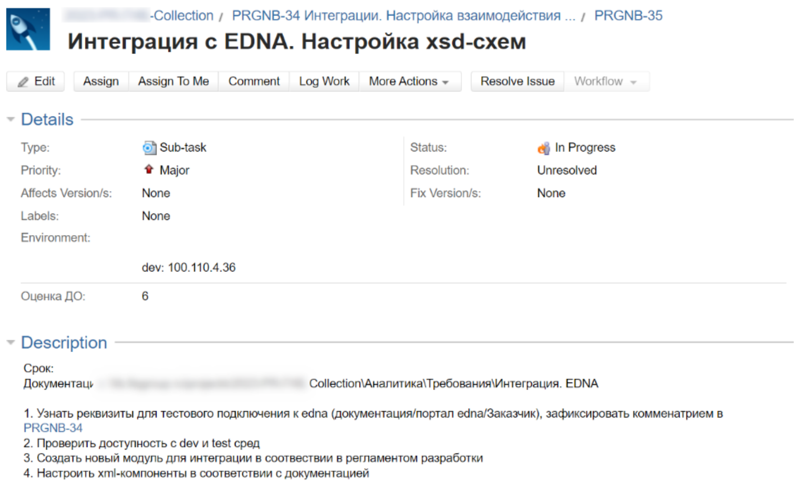
As a part of data transformation process, the subsystem processed tasks descriptions, comments and work logs in order to capture information about applied competencies. Example of task's summary and description is shown in Fig.2: developer has to prepare XML schemas and develop separate module for integration with external system «EDNA».

Tomita-parser was used for natural language processing and extracting facts. For example, in description of competency from subdomain «Types of tasks» it is indicated, that a «Middle» grade developer can install system updates on other environments. To find this fact, a rule in Tomita-parser was prepared, based on which the parser searched for agreed upon by case words in task descriptions and work logs, one of which should be mentioned in keywords set «release»:

```
ModuleImport -> Word<c-agr[1]>* Word<kwset=[release], c-
agr[1], rt>;
S -> ModuleImport interp (DevTechRelease.Term::not_norm);
```

Compiled sets of keywords included names of technologies and tools in both English and Russian:

```
TAuxDicArticle release
{
    key = "импорт модуль"
    key = "перенос модуль"
    key = "релиз"
    key = "module import"
}
```



[redacted]-Collection / PRGNB-34 Интеграции. Настройка взаимодействия ... / PRGNB-35

Интеграция с EDNA. Настройка xsd-схем

Details

Type:	Sub-task	Status:	In Progress
Priority:	Major	Resolution:	Unresolved
Affects Version/s:	None	Fix Version/s:	None
Labels:	None		
Environment:	dev: 100.110.4.36		
Оценка ДО:	6		

Description

Срок:
Документаци [redacted] Collection\Аналитика\Требования\Интеграция. EDNA

1. Узнать реквизиты для тестового подключения к edna (документация/портал edna/Заказчик), зафиксировать комменатрием в PRGNB-34
2. Проверить доступность с dev и test сред
3. Создать новый модуль для интеграции в соотвествии в регламентом разработки
4. Настроить xml-компоненты в соответствии с документацией

Fig. 2. Example of task formulation

Download subsystem recorded number of found facts by Tomita-parser in accordance with the prepared rule. A total of 44 rules for fact extraction were prepared.

Fig. 3 shows structural and functional model for competency assessment subsystem, which conducted a sequential four-step evaluation procedure.

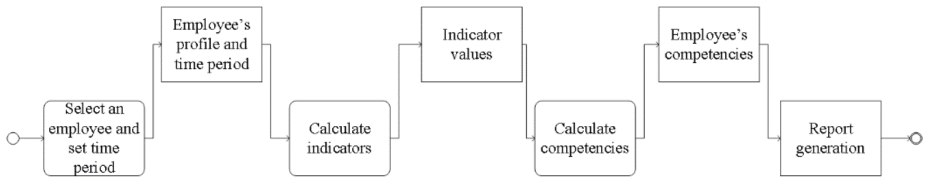


Fig. 3. Structural and functional model of the competency assessment subsystem

The first step was selecting an employee, whose competencies needed to be assessed, and setting evaluation time period.

The second step was calculation of indicators. For the selected set of competencies from the framework, 133 indicators were identified to determine whether or not a developer possessed necessary competencies. The subsystem used downloaded and prepared data for calculations.

Information from the task management system were used to calculate indicators related to the task solution: number of tasks solved within evaluation period; average number of tasks solved within and outside the estimated time frame; number of extracted facts in tasks, comments and worklogs, such as the number of tasks requiring knowledge of Web Services Description Language, Lightweight Directory Access Protocol and etc.

Data from version control system were used to calculate following indicators: code maintainability level; Halstead complexity measures; number of used Java libraries; number of used functions for integration with external systems; number of developed user functions; number of developed SQL search methods and etc.

Employee activity metrics such as the number of created pages and the number of page content updates were calculated using information from the knowledge management system.

The third step was calculation of competencies. In order to evaluate developer's competencies, the assessment subsystem compared values of calculated indicators from the previous stage with target values. For example, indicator «level of code maintainability» should be more than or equal to 155 to establish whether a developer had the competency «basic proficiency in programming language». Target values were determined based on the analysis of collected data and interviews with leading experts from the company. Because target values for several competences were not specified, it was decided that values must be strictly larger than 0. If at least one of the indicators did not fulfill the necessary requirement, it was recorded that the developer did not have competency.

The last step was generation of report, which presented the results of the assessment.

As a result, sample included 661 evaluations of 100 developers of grades «Middle», «Senior» and «Lead» from 2019 to 2022. For every evaluation result a date interval was determined, based on this date interval, decision support system downloaded data and calculated metrics. To carry out the calculations, information on 225657 tasks, 4232 pages was downloaded task management and knowledge management system accordingly and source code was downloaded from 41 repositories.

4 Results

The following competencies were confirmed to a considerable extent in a sample of 321 evaluation results of «Middle» grade developers:

- can consult the project team on current functionality. Can consult the customer on working with the system (confirmed by 99% of the developers of this grade);
- can analyze client requirements in order to design features (confirmed by 92%);
- knows how to develop new functionality, if its estimated effort is less than 40 hours (confirmed by 79%)
- can mentor a «Junior» developer and adapt an experienced colleague to a new project team (confirmed by 65%).

The competencies listed below were confirmed in the sample of 158 evaluation results of «Senior» grade developers:

- can analyze requirements and estimate the effort required for implementation (confirmed by 96%);

- can install updates on other environments and solve problems that arise during installation (confirmed by 94%);
- understands projects' business component. Understands project's current stage and makes decisions on release planning (confirmed by 80%);
- can configure integrations with external systems (confirmed by 60%);
- can administer Java application server (confirmed by 58%);
- can use Java language elements in the application source code (confirmed by 56%).

The competencies mentioned below were confirmed for the sample of 182 evaluation results of «Lead» grade developers:

- can consult the project team on current functionality. Can consult the customer on working with the system (confirmed by 99%);
- can analyze requirements and estimate the effort required for implementation (confirmed by 98%);
- can administer Java application server (confirmed by 77%);
- can mentor a «Junior» developer and adapt an experienced colleague to a new project team (confirmed by 76%);
- can use diagnostic and debugging tools. Can gather complete diagnostic information about the problem, monitor server status (confirmed by 70%);
- understands projects' business component. Understands project's current stage and makes decisions on release planning (confirmed by 70%);
- documents the functional elements of the system (confirmed by 65%).

According to the findings, communication and teamwork skills are important in determining a developer's qualification. The ability to communicate concisely and clearly with colleagues and clients is critical for problem-solving and decision-making. Therefore, competency related to consulting clients and project team was confirmed by the majority of developers of all grades.

Mentorship is a common way of teaching and onboarding of new employees. Software engineers who participate as mentors develop their knowledge and communication skills. During onboarding, mentees ask questions that allow mentors to take a different look at their professional skills and knowledge. However, this competence was confirmed by only 42% of «Senior» developers, possibly indicating that in the case company employees of this qualification are more focused on technical tasks.

Knowledge of development tools and how to use them in problem-solving are important aspects of a developer's qualification. As a result, competencies related to server administration, installing updates, using Java and configuring integration with external systems have been confirmed by most of the developers.

Gathering, analyzing requirements, and designing solutions also play a crucial role in the success of system's implementation. If errors in requirements are detected in the latter stages of project, it may take a lot of effort to correct them and modify software product. As a result, it is important for a developer to understand project's business component, analyze requirements and communicate with clients.

Due to the fact that the other competencies were not sufficiently confirmed, i.e., they were confirmed by less than 50% of the developers of the corresponding grade, it was decided to apply fuzzy logic for the assessment, as decision-making in this area is considered approximate, and results are expressed in linguistic terms.

Fuzzy logic model for assessing qualification level was developed in the MatLAB environment using the Fuzzy Logic Toolbox package. For the output variable «qualification level», the current version of the model examined two initial grades – «Junior» and «Middle», since the case company have been restructuring the grade system and only the first two grades remained unchanged.

Criteria system for calculating the qualification level included such variables as «Technology stack», «Other technical skills», «Task Formulation» which were calculated using individually designed sub-models that used data from the systems. A developer, for example, was regarded to have the «Middle» competency grade if they had completed a task related to this area of expertise an appropriate number of times and had utilized their skills and knowledge in development.

For instance, used indicators for calculation of «Other technical skills» variable are presented in Table 2.

Table 2. System of indicators for variable «Other technical skills»

Variable	Value Range	Term Set
Mentions of the designer update in assigned tasks and work logs	[0;10]	Junior Middle
Development of integration services	[0;20]	Junior Middle

A sub-model was developed to compute the second variable «Development of integration services», which calculated a grade based on values of following indicators:

- number of temporary types created;
- number of applied functions for interaction with external systems via HTTP protocol;
- mentions of Java programming language in assigned tasks, comments and work logs;
- number of uploaded XSD schemas;
- mentions of integrations in assigned tasks, comments, and work logs.

It is planned to further modify and validate the fuzzy logic-based system in order to use for formulation of competency list, that developers will need to acquire for desired grade.

5 Conclusion

The developer competency framework was developed and validated in the current work. It was found that competences related to coding, requirements analysis, teamwork, communication skills and mentoring are considered important for developer.

The framework allows software developers to determine which competencies are necessary for an experienced professional and which competencies they may need to acquire in order to become a more competent specialist.

Using the framework and developed system, managers can assess an employee's qualification and formulate relevant tasks for professional development. Mismatches between observed and desired grade competencies can be used to establish the main objectives of personal development plans.

The system will be refined in the future to evaluate qualification level using fuzzy logic to determine the necessary competencies that developers need to acquire.

References

1. Gastaldi L., Corso M. Smart healthcare digitalization: using ICT to effectively balance exploration and exploitation within hospitals. International Journal of Engineering Business Management, vol. 4, pp. 1–13 (2012).
2. Gubler M., Coombs, C., Arnold J. The gap between career management expectations and reality – empirical insights from the IT industry. Gr Interakt Org 49, pp. 12–22 (2018).
3. Loogma K., Ümarik M., Vilu, R. Identification-flexibility dilemma of IT specialists. Career Development International, 9 (3), pp. 323–348 (2004).
4. Beausaert S.A.J, Segers M.R.S, Grohnert T. Personal Development Plan, Career Development, and Training. The Wiley Blackwell Handbook of the Psychology of Training, Development, and Performance Improvement. pp. 336–353 (2014).
5. Kostrzewski A.J., Dhillon S., Goodsman D. Taylor, K.M.G. The influence of continuing professional development portfolio records on pharmacy practice. The International Journal of Pharmacy Practice, Vol. 17 No. 2, pp. 107-13 (2009).
6. Austin Z., Marini A., Desroches B. Use of a learning portfolio for continuous professional development: a study of pharmacists in Ontario (Canada). Pharmacy Education, Vol. 5, pp. 175-181 (2005).
7. Bourque P., Fairley R. E. Guide to the Software Engineering Body of Knowledge. Version 3.0. IEEE Computer Society, (2014).
8. Sedelmaier Y., Landes D. Software Engineering Body of Skills (SWEBOS). In 2014 IEEE Global Engineering Education Conference (EDUCON), pp. 395–401 (2014).
9. Beausaert S.A.J., Segers M.S.R., Gijselaers W.H. Using a Personal Development Plan for Different Purposes: Its Influence on Undertaking Learning Activities and Job Performance. Vocations and Learning. 4 (3). pp. 231–252 (2011).
10. Shippmann J.S., Ash R.A., Battista M. et al. The practice of competency modeling. Personnel Psychology, 53, pp. 703–740 (2000).
11. Bartram D., Robertson I.T., Callinan, M. Introduction: A Framework for Examining Organizational Effectiveness. In Organizational Effectiveness (2002).
12. Markus L.H., Cooper-Thomas H.D., Allpress K.N. Confounded by competencies? An evaluation of the evolution and use of competency models. New Zealand Journal of Psychology, 34, pp. 117–126 (2005).
13. Robillard M. P., Coelho W., Murphy G. C., Society I. C. How effective developers investigate source code: an exploratory study. IEEE Transactions on Software Engineering, vol. 30, no. 12, pp.889–903 (2004).
14. Surakka S. What subjects and skills are important for software developers? Communications of the ACM, vol. 50, no. 1, pp.73–80 (2007).
15. Ahmadzadeh M., Elliman D., Higgins C. An analysis of patterns of debugging among novice. SIGCSE, pp. 84–88 (2005).
16. Robillard M.P., Coelho W., Murphy G.C. How effective developers investigate source code: an exploratory study. Society IC, vol. 12, no. 30, pp.889–903 (2004).

17. Ahmed F., Capretz L. F., Campbell P. Evaluating the demand for soft skills in software development. IT Prof, vol. 14, no. 1, pp.44–49 (2012).
18. Matturro G., Raschetti F., Fontán C. A Systematic Mapping Study on Soft Skills in Software Engineering. Journal of Universal Computer Science, vol. 25, pp. 16–41 (2019).
19. Li P. L., Ko A. J., Begel A. What Makes A Great Software Engineer? 37th International Conference on Software Engineering, pp.700-710 (2015).
20. Baltes S., Diehl S. Towards a theory of software development expertise. Proceedings of the 2018 26th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering, ESEC/FSE 2018, 2018.
21. Bergersen G. R., Sjøberg D. I. K., Dybå T. Construction and validation of an instrument for measuring programming skill. IEEE Trans. Softw. Eng., vol. 40, no. 12, pp. 1163–1184 (2014).
22. Gousios G., Kalliamvakou E., Spinellis D. Measuring developer contribution from software repository data. In Proceedings of the 2008 international working conference on Mining software repositories, MSR ’08, pp.129-132 (2008).
23. Gavriliev E.I., Avdeenko T.V. Model and Procedure for Assessing the Qualification of a Software Developer. 2022 IEEE 23rd International Conference of Young Professionals in Electron Devices and Materials (EDM), pp. 303–307 (2022).
24. Gavriliev E.I., Avdeenko T.V Procedure for assessing the qualifications of a software developer. Nauka. Tekhnologii. Innovacii: Collection of scientific papers. In 10 parts, Novosibirsk, December 06–10, 2021. pp. 145–148 (2021).