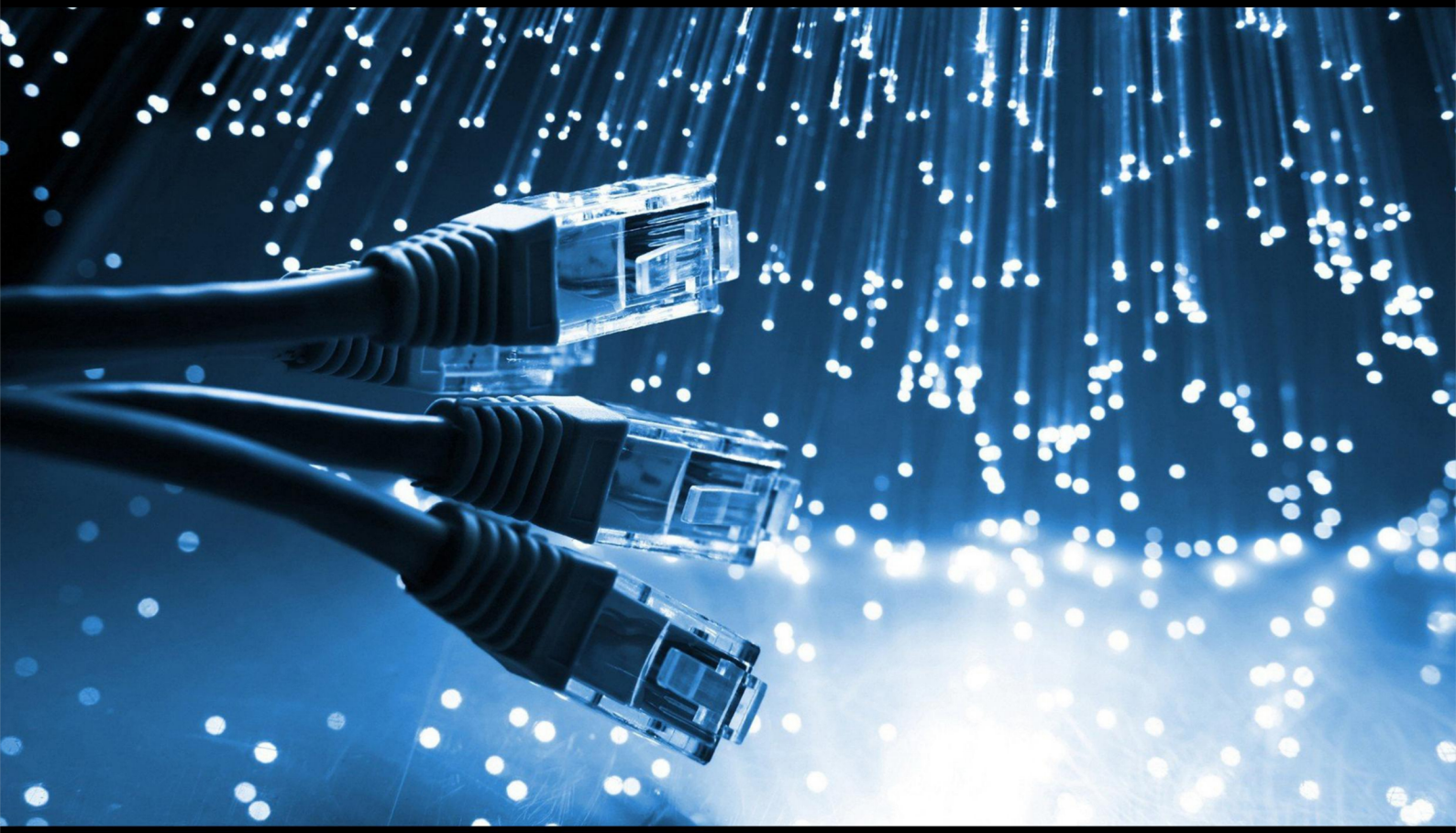
JCSI

Journal of Computer Sciences Institute

Volume 13/2019



Department of Computer Science
Lublin University of Technology

jcsi.pollub.pl

ISSN: 2544-0764

Redakcja JCSI
e-mail: jcsi@pollub.pl
www: jcsi.pollub.pl
Katedra Informatyki
Wydział Elektrotechniki i Informatyki

Politechnika Lubelska
ul. Nadbystrzycka 36 b
20-618 Lublin

Redaktor naczelny:
Tomasz Zientarski
e-mail: t.zientarski@pollub.pl

Redaktor techniczny:
Beata Pańczyk,
e-mail: b.panczyk@pollub.pl

Recenzenci numeru:
dr hab. inż. Dariusz Czerwiński, prof. PL
dr Mariusz Dzieńkowski
dr inż. Jakub Smołka
dr inż. Maria Skublewska-Paszkowska
dr inż. Marek Miłosz
dr inż. Piotr Kopniak
dr inż. Elżbieta Miłosz
dr inż. Maciej Pańczyk
dr inż. Sławomir Przyłucki
dr inż. Dariusz Gutek
dr inż. Grzegorz Kozieł
dr inż. Piotr Muryjas
dr inż. Jacek Kęsik

Skład komputerowy:
Monika Kaczorowska
e-mail: m.kaczorowska@pollub.pl

Projekt okładki:
Marta Zbańska

JCSI Editorial
e-mail: jcsi@pollub.pl
www: jcsi.pollub.pl
Department of Computer Science
Faculty of Electrical Engineering and Computer Science
Lublin University of Technology
ul. Nadbystrzycka 36 b
20-618 Lublin, Poland

Editor in Chief:
Tomasz Zientarski
e-mail: t.zientarski@pollub.pl

Assistant editor:
Beata Pańczyk,
e-mail: b.panczyk@pollub.pl

Reviewers:
Dariusz Czerwiński
Mariusz Dzieńkowski
Jakub Smołka
Maria Skublewska-Paszkowska
Marek Miłosz
Piotr Kopniak
Elżbieta Miłosz
Maciej Pańczyk
Sławomir Przyłucki
Dariusz Gutek
Grzegorz Kozieł
Piotr Muryjas
Jacek Kęsik

Computer typesetting:
Monika Kaczorowska
e-mail: m.kaczorowska@pollub.pl

Cover design:
Marta Zbańska

ISSN 2544-0764

Spis treści

Contents

JOURNAL of COMPUTER SCIENCES INSTITUTE

JCSI 13 (2019) 267-271
Wysłane: 2019-10-20
Przyjęte: 2019-10-28

Machine Learning as a method of adapting offers to the clients

Jacek Bielecki*, Oskar Ceglarski*, Maria Skublewska-Paszkowska

Department of Computer Science, Lublin University of Technology, Nadbystrzycka 36B, 20-618 Lublin, Poland

Abstract. Recommendation systems are class of information filter applications whose main goal is to provide personalized recommendations. The main goal of the research was to compare two ways of creating personalized recommendations. The recommendation system was built on the basis of a content-based cognitive filtering method and on the basis of a collaborative filtering method based on user ratings. The conclusions of the research show the advantages and disadvantages of both methods.

Keywords:recommender system; collaborative filtering; cognitive filtering; machine learning

*Corresponding author.

E-mail addresses: jacek.bielecki@pollub.edu.pl, oskar.ceglarski@pollub.edu.pl

Uczenie maszynowe jako metoda dostosowywania ofert do klientów

Jacek Bielecki*, Oskar Ceglarski*

Politechnika Lubelska, Katedra Informatyki, Nadbystrzycka 36B, 20-618 Lublin, Polska

Streszczenie. Systemy rekomendacji to aplikacje filtrujące dane, których głównym zadaniem jest dostarczanie spersonalizowanych rekomendacji produktów. Celem badań było dokonanie analizy i porównania dwóch metod uczenia maszynowego wykorzystywanych do generowania rekomendacji. System rekomendacji zbudowano na podstawie metody filtrowania kognitywnego opartej o treści oraz na podstawie metody filtrowania kolaboratywnego opartej o oceny użytkowników. Wnioski z przeprowadzonych badań pokazują wady i zalety obu metod.

Słowa kluczowe: System rekomendacji; filtrowanie kolaboratywne; filtrowanie kognitywne; uczenie maszynowe

*Autor do korespondencji.

Adresy e-mail: jacek.bielecki@pollub.edu.pl, oskar.ceglarski@pollub.edu.pl

1. Introduction

Recommendation systems (RS) are class of information filter applications whose main goal is to provide personalized recommendations of products, content and services to users. A recommendation system for an e-commerce site helps users to find products, such as movies, songs, books, gadgets, applications and restaurants that fit their personal preferences and needs [1]. Recommendation systems enhance e-commerce sales by converting browsers into buyers, exposing customers to new products, increasing cross-selling by suggesting additional products, building customer loyalty, increasing customers satisfaction based on their purchasing experience, and increasing the likelihood of repeat visits by satisfied customers. Each of these can be translated into increased sales and higher revenue [1]. In the age of e-commerce, it is important for companies to develop web-based marketing strategies such as product bundling to increase revenue. The e-commerce industry predominantly uses various machine learning models for product recommendations and analyzing a customer's behavioral patterns, which play a crucial role in exposing customers to new products based on their online behaviour [2]. Psychology studies show that if customers are shown products suited to their personality type or complementing their lifestyle, the chances of buying them grows considerably [2].

The most widely used filtering algorithms presented in the literature for the recommendation task are: collaborative filtering, demographic filtering, content-based filtering, and hybrid filtering [3]. The content-based method mostly takes into account the implicit rating by text mining process and makes a recommendation, whereas collaborative filtering considers only explicit ratings of users [4]. Furthermore, there are two types of collaborative filtering techniques frequently used in the recommendation system domain such as model-based and memory-based collaborative filtering. The model-based method develops a user model utilizing ratings of each user to evaluate the expected value of unrated items. On the other hand, memory-based method utilizes similarity measure computed from the explicit user rating to identify neighborhoods and perform prediction [1][4].

The content-based filtering makes recommendations based on user choices made in the past (e.g., in a web-based e-commerce RS, if the user has purchased comedy films in the past, the RS will likely recommend a newly released comedy that the user has not yet purchased on this website). The content-based filtering also generates recommendations using the content from objects intended for recommendation; therefore, specific content can be analyzed such as text, images, and sound [7]. The transformation of content described using a human-understandable language requires transformation into a machine-understandable language. This process is possible by using eg. Natural Language Toolkit [8]. These tools allow to save products in a multidimensional matrix. It is possible to determinethe similarity between them using mathematical functions [8]. Every recommendation method which based on users' profiles needs to create multidimensional matrices describing the user in a certain way. For the content based method, the attributes of the users' profiles are the movies that they rated [7]. The content-based

approaches the focus on measuring the functional similarities between the content of services and user queries using keyword search or semantics-aware search. Keyword search methods usually have many limitations due to the insufficiencies in identifying semantically relevant keywords. Semantics-aware search methods can be further divided to two subgroups: logical ones based on ontologies and non-logical ones based on latent factor models (also called topic models) such as LDA (Latent Dirichlet Allocation) [9]. Logical semantics-aware methods require well-defined ontologies and semantic annotation of services and user queries, which makes them hard to apply; while the non-logical methods are generally not very effective due to the coarse-grained semantics captured by topic models [9].

Similarity measure in the recommender system is the statistical measure of how two users and items are related to each other. There are several traditional similarity metrics such as Cosine(COS), Pearson's Correlation (COR), Constrained Pearson's Correlation (CPC), Mean Squared Difference (MSD), Jaccard, JMSD etc. [10]. Cosine similarity measures the angle between two rated vectors where the smaller angle indicates greater similarity and higher angle show lesser similarity [6][11]. The cosine similarity could by computed by formula:

$$Sim(u,v)^{cos} = \frac{\sum_{i \in I(u,v)} R(u,i) \cdot R(v,i)}{\sqrt{\sum_{i \in I(u,v)} R(u,i)^2} \cdot \sqrt{\sum_{i \in I(u,v)} R(v,i)^2}} \quad (1)$$

where $R_{(u,i)}$ is the rating of the item i given by user u and $I_{(u,v)}$ is the number of co-rated items of users u and v [11, 12]. The range of cosine similarity is 0 to 1, where higher value signifies the closest similarity between users u and v.

The collaborative filtering (CF) approach is considered one of the most popular and effective techniques for building recommender systems [5]. The basic idea is to try to predict the user's opinion about different items and recommend the "best" items, using the user's previous preferences and the opinions of other similar users [1]. Collaborative Filtering allows users to provide ratings about a set of elements in such a way that when enough information is stored on the system, recommendations can be made to each user based on information provided by other users that are thought to have the most in common with them [1]. There are two types of collaborative filtering techniques frequently used in the recommendation system domain such as model-based and memory-based collaborative filtering. Model-based method develops a user model utilizing ratings of each user to evaluate the expected value of unrated items [5]. On the other hand, memory-based method utilizes similarity measure computed from the explicit user rating to identify neighbourhoods and perform prediction [5]. The traditional CF techniques are said to be memory-based because the original ratings database is used directly for generating the recommendations or making the predictions [6]. On the other hand, model-based approaches use ratings database to learn a predictive model which can be used to predict ratings of users for new items. Memory-based CF methods can be further divided into two groups, namely user-based and item-based algorithms [12]. The user-based algorithms look for users (also called neighbours) similar to the active user, and calculate a predicted rating as a weighted average of the neighbor's ratings on the active item. On the other hand, item-based algorithms look for similar items for an active user [12].

In 2005, Lemire and Maclachlan proposed Slope One family of algorithms to make the CF prediction faster than memory-based algorithms [13]. It has been shown that the Slope One is reasonably accurate despite its simplicity, efficiency, easiness to implement, updatability and scalability [12]. However, if there are no users or only a few users have rated the active item, the accuracy of the algorithm will decrease considerably [13][14]. Slope one algorithm adopts an easy but effective concept as a simple linear regression model to predict ratings. Its original idea was on the basis of what the authors call popularity differential between users and items. In general, the problem is to find functions of the form $f(x) = x + b$ where b is a constant and x is a variable representing rating values [12][13]. The Slope One algorithm is a typical item-based CF and mainly considers the users rating the active item and the other items rated by the active user [14]. It uses these ratings of the users to predict the rating of the active item. The Slope one could also be user-based CF and in this case it uses users ratings of the product to predict the ratings for other products [14]. In the basic Slope One, the constant b is defined as the average difference between each item and the item to be predicted; computing among the users that have rated both items [12]. This average deviation for two items i and j is calculated as:

$$dev_{j,i} = \frac{1}{|U_{i,j}|} \sum_{\in U_{i,j}} \left(r_{u,j} - r_{u,i} \right) \quad (2)$$

From every co-rated item i, a prediction for item j of user u can be obtained as $dev_{j,i} + r_{u,i}$, where $r_{u,i}$ represents the rating value given by user u to item i [12, 13]. A simple approach for combining these individual predictions is to compute the average over all co-rated items as:

$$p_{u,j} = \frac{1}{|R_{u,j}|} \sum_{i \in R_{u,j}} \left(dev_{j,i} + r_{u,i} \right) \quad (3)$$

where $R_{u,j} = \{i \in Iu : |U_{i,j}| > 0\}$ is the set of relevant items [12][13].

However, current Slope-one based algorithms are all designed for static datasets, which are contradictory to real situations where dynamic datasets are mostly involved [15][16]. Note that in real applications, data increments can arrive at every millisecond, making a target dataset constantly change. Therefore, incremental recommenders which are able to address such data dynamics are greatly desired [16][17]. In this research, this problem was resolved by computing a prediction again when there was new ratings added to the dataset.

2. Research method

The research was made on the implemented movie store system, which allows user to watch movies and rate them. The movie store has been named "Intelligent Movie Store" and it bases on MovieLens latest datasets. The research was carried out on the group of 100 people of different age and different gender. Everyone, who participated in the research

had to register their own account in movie store system. Account registration required entering all the necessary demographic user data that was used to create recommendations. Each participant of the research had to give his gender, age and country of residence. The registration form to the Intelligent Movie Store system was shown on Figure 1.

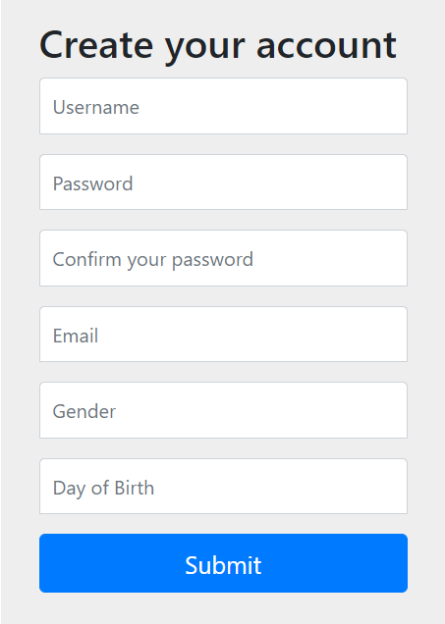


Fig. 1. Registration form to Intelligent Movie Store system

After registration, each participant of the research has to indicate about Wight movies known to him/her movies which he/she likes and rate them. This part of the research was necessary to create recommendations based on collaborative filtering by users.

In the next step of the research, users have been split into two equal groups. Each one consisted of 50 people. Each group of people got the movie recommendations generated by different methods. The first one was getting the recommendations generated by content based method. First the NTLK tools have been used to save all movies in the multidimensional matrix. Then the cosine similarity method, which is described by formula (1), was used to compute the similarity between movies. This approach allowed to find movies which were similar to best rated movies and then recommend them. The second group of people had the recommendations generated by collaborative filtering method. In this case the Slope one algorithm was used and the predictions were computed using the (3) formula.

Every participant of the study got about 8 movie recommendations. Analysing the effectiveness of the recommendation system was based on user's responsiveness to the recommended products. Every recommendation could be rated by the user in positive or negative way. User could also ignore the recommendation. Every user's integration with the recommendation have their own numeric equivalent which means a specific level of user's responsiveness to this recommendation. For example, when user clicked on the recommendation to rate it positively - the responsibility status of that recommendation got status "2" in numeric equivalent. But when user rated low the recommendation, it got status "-2". In case when user got movie recommendation that is known to him, he could rate this movie on a scale of 0 to 10. When the user's rate was higher than 5, that recommendation got status "1" and when the rate was lower or equal to 5 the recommendation got status "-1". The example of movie recommendation with all possible response use cases is shown on Figure 2.

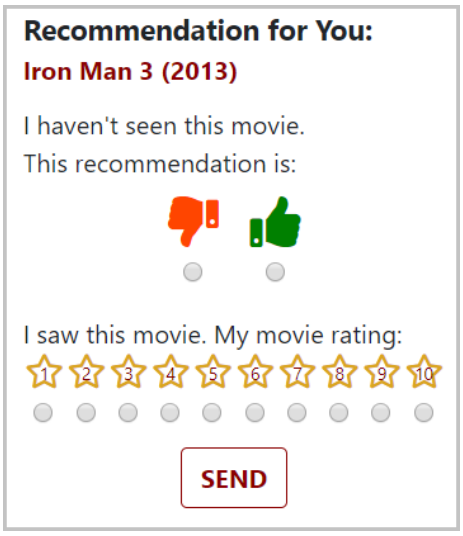


Fig. 2. Example movie recommendation in Intelligen Movie Store system

Such a grading scale allows to determine if the recommendations were well matched to the users preferences or not.

3. Results

In the Table 1. the percentage distribution of movie ratings was shown. This ratings were collected in the first stage of research, before the users got any recommendations. There separated data is shown for both group of participants of research which have got recommendation generated by different methods in the next stage of research.

Table 1. Percentage distribution of movie ratings from the first stage of research work

	Method	
Rating	**CB**	**CF**
1	0.62%	0.25%
2	0.62%	1.24%
3	1.24%	3.72%
4	3.30%	4.71%
5	4.74%	2.98%
6	12.78%	10.42%
7	23.51%	22.83%
8	31.55%	26.55%
9	18.14%	22.08%
10	3.51%	5.21%

These data were grouped for both methods and then represented in the graph to show the normal distribution of movie ratings. The percentage distribution of each movie rating in the system is shown on Picture 3.

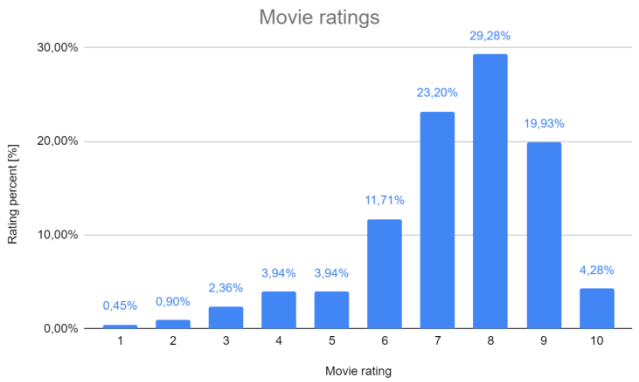


Fig. 3. The percentage distribution of each movie rating in the system

The content based method have got 43.42% of recommendations with status "2" and 30.48% of recommendations with status "1". On the other hand, the collaborative filtering method got 47.60% of recommendations with status "2" and 8% of recommendations with status "1". The percentage distribution of recommendations with different statuses is presented in Table 2.

Table 2. The percentage distribution of recommendations with different statuses

	Method	
Status	**CB**	**CF**
-2	25.00%	40.80%
-1	1.67%	2.80%
0	1.43%	0.80%
1	30.48%	8.00%
2	41.43%	47.60%

The percentage distribution of recommendations with different statuses have been also represented in graph which was shown on Figure 4.

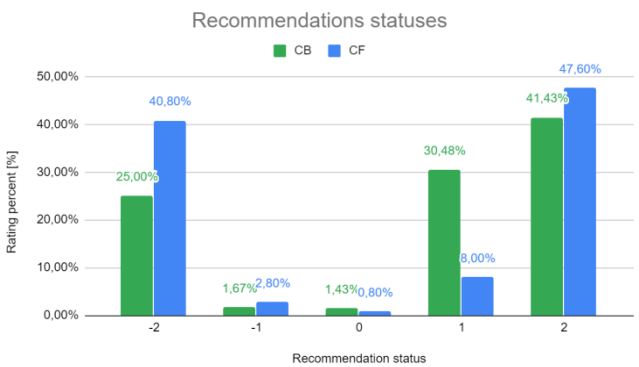


Fig. 4. The percentage distribution of recommendations with different statuses

The content based method got the effectiveness in level of 71.9% of all positive rated recommendations. A positively rated recommendation is a recommendation with the status 1 or 2. From the other side, the collaborative filtering method got the effectiveness only in level of 55.6% of positive rated recommendations. The ratio of positively rated recommendations to negatively rated ones according to both methods are shown on Figure 5.

Fig. 5. The ratio of positively rated recommendations to negatively rated ones according to both methods

There were 1.43% of non rated recommendations for content based method and 0.80% of non rated recommendations for collaborative filtering method and this also have been shown on Picture 5.

4. Discussion of results and conclusions

The normal distribution of movie ratings (Figure 3) could be shifted to the right because of fact that users have selected and rated movies which were well known to them. Also the fact that in database of movies used In the research was based on the 9000 most popular movies according to MovieLens lab. It means that in the Intelligent Movie Store database were only well known, popular and interesting movies.

The content based method got the effectiveness in level of 71.9% of all positive rated recommendations and rom the other side, the collaborative filtering method got the effectiveness only in level of 55.6% of positive rated recommendations. There is some disparities in effectiveness of different recommending methods and this has been shown in Figure 5. Such a disparities in recommendation effectiveness results arise from the problem with a cold start, which is related to collaborative filtering method because of fact that it needs a lot of data about user's ratings of products to achieve better results [18][19][20]. A solution to this problem could be creating a recommendation system that is using different types of deep learning methods in case of different data access status. Eg. on the starting stage of movie store system work, the content based method could be used to avoid the cold start problem. In case when a lot of data with users movie ratings have been collected, the recommending method could be switched to collaborative filtering that should be more effective [18][20].

Bibliography

[1] M. Beladev, L. Rokach, B. Shapira, Recommender systems for product bundling, Knowledge-Based Systems, 1 Nov. 2016.

[2] A. Marwade, N. Kumar, S. Mundada, Augmenting E-Commerce Product Recommendations by Analyzing Customer Personality, Conference: 9th International Conference on Computational Intelligence and Communication Networks (CICN); 16-17 Sep. 2017.

[3] J. Bobadilla, Recommender systems survey, Knowledge-Based Systems, 46 (2013), 109-132.

[4] S. Jiang, Q. Xueming, S. Jialie, F. Yun, Author topic model-based collaborative filtering for personalized POI recommendations, IEEE transactions on multimedia 17:6 (2015), 907-918.

[5] R. Burke. Hybrid recommender systems: survey and experiments, UMUAI, 12 (4) (2002), 331-370.

[6] S. Bag, SK. Kumar, MK. Tiwari, An efficient recommendation generation using relevant Jaccard similarity, Information Sciences, May 2019.

[7] D. McIlwraith, M. Haralambos, B. Dmitry, Inteligentna sieć, Algorytmy przyszłości. Helion, Gliwice, 2017.

[8] Natural Language Toolkit, https://www.nltk.org/api/nltk.html.

[9] F. Xie , J. Wang , R. Xiong , N. Zhang, An Integrated Service Recommendation Approach for Service-Based System Development, Expert Systems With Applications, 2019.

[10] B. K. Patra, R. Launonen, V. Ollikainen, S. Nandi, A new similarity measure using Bhattacharyya coefficient for collaborative filtering in sparse data, Knowledge-Based Syst. 82 (2015) 163–177.

[11] H. J. Ahn, A new similarity measure for collaborative filtering to alleviate the new user cold-starting problem, Inf. Sci. (Ny). 178 (2008) 37–51.

[12] M. Saeed, E.G. Mansoori, A new slope one based recommendation algorithm using virtual predictive items, Journal of Intelligent Information Systems, June 2018, Volume 50, Issue 3, pp 527–547

[13] D. Lemire, A. Maclachlan, Slope One predictors for online rating-based collaborative filtering, SDM. SIAM, 2005 (Vol. 5 pp. 1–5)

[14] W. Yongqiang, Y. Liang, C. Bing, Learning to Recommend Based on Slope One Strategy, Web Technologies and Applications. Proceedings of the 14th Asia-Pacific Web Conference, APWeb 2012 pp: 537-4.

[15] QX. Wang, X. Luo, Y. Li, Incremental Slope-one recommenders, Neurocomputing, Volume 272, Journal of Computer Sciences Institute, 10 January 2018, pp 606-618

[16] X. Luo., Y.-N. Xia, Q. Zhu, Incremental collaborative filtering recommender based on regularized matrix factorization, Knowl. Based Syst., 27 (2012), pp. 271-280

[17] X. Luo., Y.-N. Xia, Q. Zhu, Y. Li, Boosting the K-nearest-neighborhood based incremental collaborative filtering, Knowl. Based Syst., 53 (2013), pp. 90-99

[18] Nguyen P., Wang J., Kalousis A.: Factorizing LambdaMART for cold start recommendations. Machine Learning, 21 July 2016

[19] T. Schreiner, A. Rese, D. Baie, Multichannel personalization: Identifying consumer preferences for product recommendations in advertisements across different media channels, Journal of Retailing and Consumer Services, May 2019

[20] A. Fiasconaro, M. Tumminello, V. Nicosia, V. Latora, R. N. Mantegna, Hybrid recommendation methods in complex networks. American Physical Society, 14 July 2015.

JOURNAL COMPUTER SCIENCES INSTITUTE

JCSI 13 (2019) 272-278
Wysłane: 2019-11-18
Przyjęte: 2019-12-11

The insulin activity model based on insulin profiles

Tomasz Nowicki

Department of Computer Science, Lublin University of Technology, Nadbystrzycka 36B, 20-618 Lublin, Poland

Abstract. The purpose of the research was to propound an insulin activity model in a human body after a subcutaneous injection. A deterministic model in the form of a mathematical function was formulated. The research was based on pharmaceutic publicly available drug information published by the manufactures. The paper presents in detail the model. The obtained results can be used in computer simulations of diabetes mellitus therapy. They suggest that activity models may be assigned to types of insulin instead of separate products.

Keywords: insulin profiles; insulin activity; diabetes mellitus; computer therapy

*Corresponding author.

E-mail address: t.nowicki@pollub.pl

Uzyskanie modelu aktywności insuliny na podstawie profili insulinowych

Tomasz Nowicki

Politechnika Lubelska, Katedra Informatyki, Nadbystrzycka 36B, 20-618 Lublin, Polska

Streszczenie. Celem badań było opracowanie modelu aktywności insuliny w organizmie człowieka po wstrzyknięciu podskórnym. Sformułowano model deterministyczny w postaci funkcji matematycznej. Badanie oparto na publicznie dostępnych informacjach farmaceutycznych opublikowanych przez producentów. Artykuł szczegółowo przedstawia model. Uzyskane wyniki mogą być wykorzystane w komputerowych symulacjach terapii cukrzycy. Sugerują one również, że modele aktywności mogą dotyczyć grup insulin zamiast konkretnych preparatów.

Słowa kluczowe: krzywa insulinowa; aktywność insuliny; cukrzyca; terapia komputerowa

*Autor do korespondencji.

Adres e-mail: t.nowicki@pollub.pl

1. Introduction

The insulin activity model presented in this paper is intended to assist development of computer systems supporting type 1 diabetes therapy. The type 1 diabetes is an incurable (nowadays) autoimmune disease with global occurrence. The consequence of getting this type is the patient's body inability to produce the insulin hormone. In this sense, a person with type 1 diabetes is a disabled person. Insulin is a crucial hormone for a metabolism, so it must be supplied from the outside a lifetime because the lack of it leads in short time to a coma and subsequently to death. The therapy with insulin is difficult and awkward. Proper doses of an insulin analog (artificial insulin used as a medicine) have to be injected a few times a day daily. Each overdose results in hypoglycemia (low blood sugar), which may even lead to sudden death. Contrarily, each under-dose brings hyperglycemia (high blood sugar), which in a longer time brings destruction of blood vessels, nerve tissues and kidneys. On the other hand a proper therapy means that the disability resulting from type 1 diabetes is almost imperceptible by the patient and his/her environment. Such a person is able to live a successful life including the professional and family sphere. Therefore, carrying out research on insulin dosing algorithms is reasoned.

Today there are many insulin analogs in use. Administering such insulin to a patient is realized by subcutaneous injections. After an injection the whole dose of the insulin does not become immediately available to a human body, but is released into the blood system gradually. The process takes place in time and can be described with a insulin profile (Fig. from 2 to 7). The profile (sometimes called an insulin curve) is a characteristic of an insulin analog and tells how blood insulin concentration changes over time after the injection of a certain dose. Although the profile depends on many different factors, it is necessary to assume one when predicting an insulin action in a body.

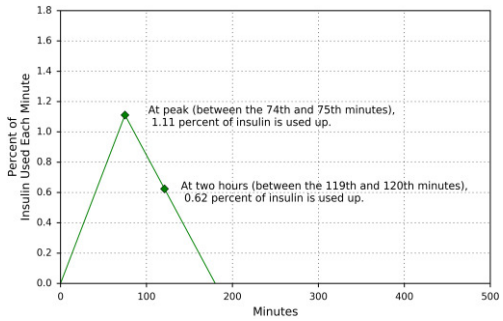


Fig. 1. Simplified insulin activity function applied in OpenAPS [1]

The insulin profiles themselves may be used in computer simulations but it would not be convenient because they refer to blood concentration. It is more usefully to know what fraction of the whole insulin dose will be released in subsequent minutes. Such a function is called the insulin activity function and because building one always needs making assumptions it is also named the insulin activity model.

Regrettably - from digital biohackers' point of view - pharmaceutical companies do not publish mathematical formulas for their insulin analogs. Therefore, software developers adopt simplified insulin activity functions. An example of such a function can be found in OpenAPS [1], which is a successful open-source artificial pancreas system that got ahead commercial equivalents a few years ago. The system has been created by volunteers and is publicly available under the MIT license. The system applies the simplest approximation that assumes linear increase and linear decrease in bioavailability of an insulin dose (Fig.1). The creators themselves agree that the model should be replaced with more precise one.

2. Analysis of insulin profiles

2.1. Common insulin analogs

Today the world demand for insulin is supplied by 6 private companies [2] (Table 1). From commercial medicine point of view it may be said that participation in the market of other suppliers is marginal. For this reason, this research has been limited to the products of the leading manufacturers. Tables form 2 to 7 enumerate insulin analogs offered by the group of interest excluding insulin mixes. In the tables next to the name of an insulin analog there is given its type and the information about its profile availability. Generally there are 4 types of insulin analogs defined by their time activity in a human body. The first group called „rapid acting" remains active up to 6 hours, the second one denoted „short acting" up to 9 hours, the next „intermediate acting" means about 12 hours of activity and the last group „long acting" stands over 20 hours. Having said that it must be noted that the given definition is rough.

Table 1. Major players in the world insulin market

Company	Origin	Web page
Novo Nordisk	Denmark	www.novonordisk.com
Sanofi	France	www.sanofi.com
Eli Lilly	USA	www.lilly.com
Biocon	India	www.biocon.com
Julphar	UAE	www.julphar.net

Table 2. Novo Nordisk essential insulin analogs offer

Name	Type	Profile published
Fiasp®	Rapid	Yes
NovoRapid®		
Actrapid®	Short	
Insulatard®	Intermediate	No
Tresiba®	Long	
Xultophy®		
Levemir®		

Table 3. Sanofi essential insulin analogs offer

Name	Type	Profile published
Apidra®	Rapid	Yes
Insuman Implantable®		
Insulin Lispro Sanofi®		No
Insuman R®	Short	
Insuman N®		
Lantus®	Long	
Toujeo®		

Table 4. Eli Lilly essential insulin analogs offer

Name	Type	Profile published
Humalog®	Rapid	Yes
Humulin R Short-acting®	Short	
Liprolog®		No
Humulin N®	Intermediate	
Abasaglar®	Long	

Table 5. Biocon essential insulin analogs offer

Name	Type	Profile published
Insugen®	Short	No

Table 6. Julphar essential insulin analogs offer

Name	Type	Profile published
Jusline N®	Intermediate	No
Jusline R®		

2.2. Recognition of insulin profiles

At the time of the research only for 7 insulin analogs there were insulin profiles included into the official information about the medical products [3-8]. The source graphs have been presented in the figures from 2 to 8.

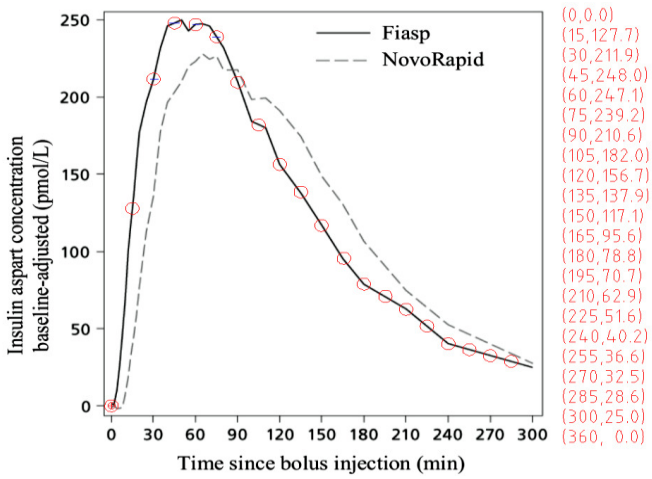


Fig. 2. Insulin profile for Fiasp including points (in red) used in polynomial fitting [3]

All the source graphs have been supplemented with points used in the calculations. The values have been obtain using the open-source graphical program LibreCad (www.librecad.org). According to the manufacturers the data came from clinical trials and was averaged before published. All the diagrams show how serum insulin concentration varies over time after a subcutaneous injection of a certain insulin dose. The diagrams are not standardized. The time is given in minutes or hours. The observation time ranges from 240 to 720 minutes. The first measurement is made at the time of the injection or before it. The insulin concentration is given in different units with or without baseline adjustment. Therefore there is a need to normalize the curves in order to use in any computer calculations.

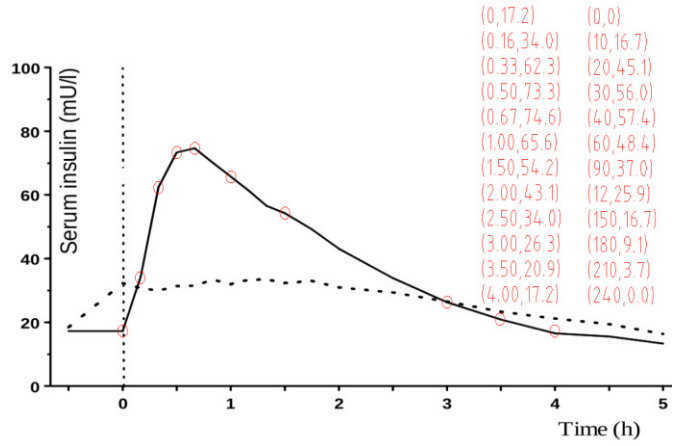


Fig. 3. Insulin profile for NovoRapid including points (in red) used in polynomial fitting. The left column gives data read from the graph, the right after baseline adjustment and time converted to minutes [4]

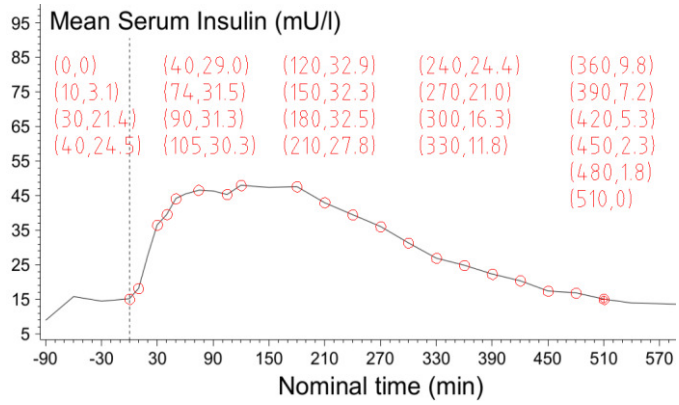


Fig. 4. Insulin profile for Actrapid including points (in red) used in polynomial fitting. Values given after baseline adjustment [5]

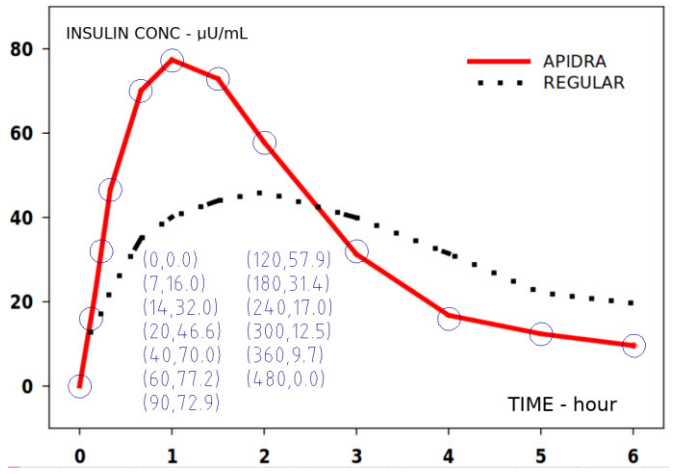


Fig. 5. Insulin profile for Apidra including points (in blue) used in polynomial fitting where hours has been converted to minutes [6]

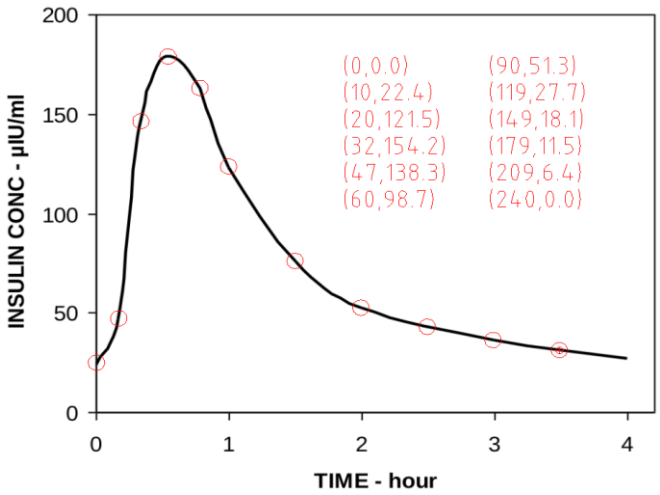


Fig. 6. Insulin profile for Insuman Implantable including points (in red) used in polynomial fitting- hours converted into minutes, concentration values given over the base level of 25 [7]

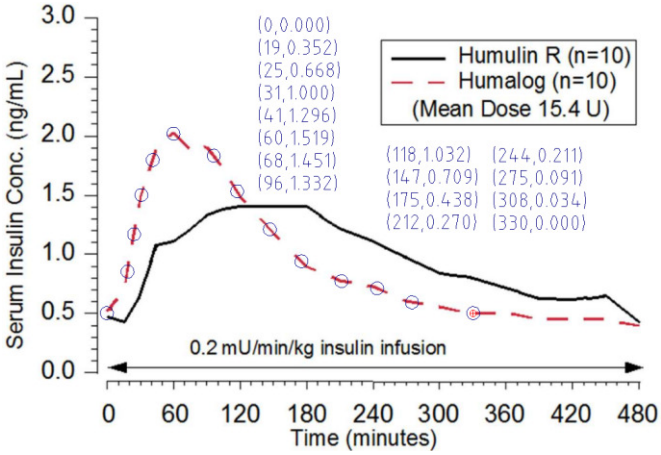


Fig. 7. Insulin profile for Humalog including points (in blue) used in polynomial fitting where serum insulin concentration is given over the base level of 0.5 [8]

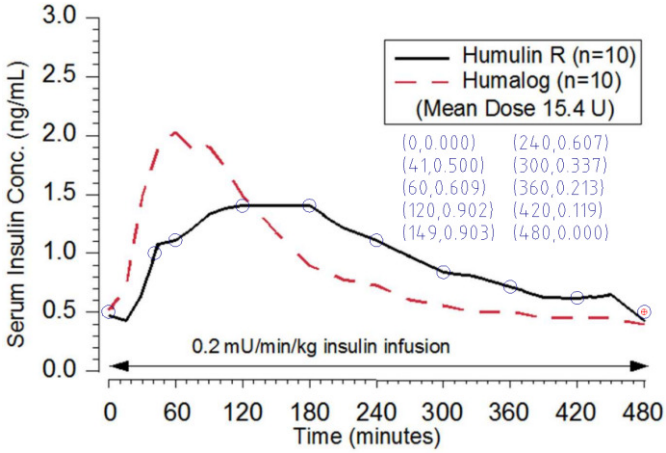


Fig. 8. Insulin profile for Humulin R Short-acting including points (in blue) used in polynomial fitting where serum insulin concentration is given over the base level of 0.5 [8]

It was assumed that the insulin curves can be approximated with polynomials. After preliminary tests (results unpublished) it was stated that the minimal degree of such a polynomial is 6:

$$C(T)=A_6\cdot T^6+A_5\cdot T^5+A_4\cdot T^4 +A_3\cdot T^3+A_2\cdot T^2+A_1\cdot T+A_0 \quad (1)$$

where: C is serum insulin concentration, T - time, A_i - individual coefficients for each insulin analog (i=0,1,..,6). The polynomial is valid for the range of insulin action starting with the time of application (injection) T_{APP} until it fades away at time point marked T_{FED}. The range of insulin action is denoted by $T_{SPAN} = T_{FED} - T_{APP}$. Within the considered range there is so called peak insulin activity. It is just maximum of the polynomial function and the time point for it is denoted by T_{PEAK} ($T_{APP} < T_{PEAK} < T_{FED}$). To simplify calculations, it was assumed that $T_{APP} = 0$.

Before fitting the polynomial (1) to the points from the diagrams (Fig. from 2 to 8) following restrictions had been imposed:

$$C(T_{APP}) = C(T_{FED}) = 0 \quad (2)$$

$$C(T_{APP} < T < T_{FED}) > 0 \quad (3)$$

$$C'(T_{PEAK}) = 0 \quad (4)$$

$$C'(T_{APP} < T < T_{PEAK}) > 0 \quad (5)$$

$$C'(T_{APP} < T < T_{PEAK}) > 0 \quad (6)$$

$$C'(T_{FED}) = 0 \quad (7)$$

The condition (2) along with (3) says that the increase in serum insulin concentration takes place from T_{APP} to T_{FED} and the injection itself cannot cause any decrease of the concentration. Satisfying the conditions (4), (5) and (6) together guarantees existence of one maximum within the range of insulin action i.e. existence of the insulin peak. Finally the condition (7) forces gradual asymptotic fading of the concentration, which can be noticed for all presented insulin analogs. Minutes were used as time unit and the point of the peak was searched for with 5 minutes resolution. This approach let keep the results practical with the therapeutic point of view. Moreover any greater accuracy would not have correspond with the source data, which had been after all averaged.

All the symbolic and numerical calculations were preformed using the open-source mathematics software SageMath (www.sagemath.org). The quality of fitting the polynomials to the points has been shown in figures from 9 to 15. The calculations resulted in finding the values of the A_i coefficients for all the insulins under study. Here the time T is given in minutes but the values of the polynomials correspond to the source units (see Fig. from 2 to 8). The symbol P in the descriptions under the figures stands for the area under the curve given in current units. These values were used in a normalization process.

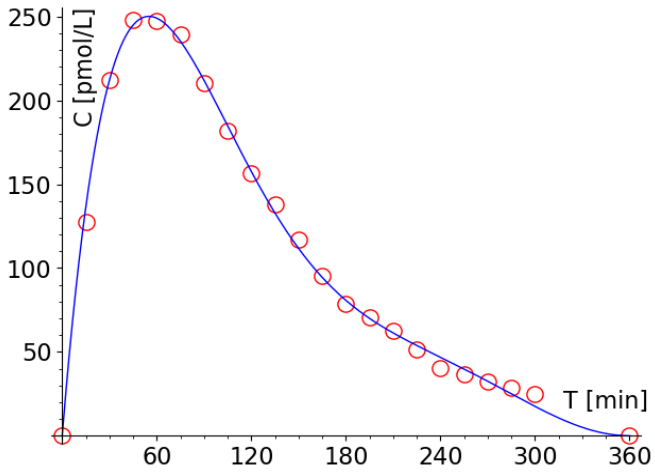


Fig. 9. Polynomial fitting for Fiasp: T_{PEAK}= 55, C(T) = -4.408×10^{-12} T^6 $+6.756\times10^{-9}$ T^5 -4.073×10^{-6} T^4 $+1.218\times10^{-3}$ T^3 -0.1840 T^2 $+11.60$ T, P=36452.5

3. The model of insulin activity

After the fitting calculations all the polynomials had been normalized. The normalization process consisted in transforming the polynomials to the range of [0,1] and reducing the area under the curve to 1. If the normalized form of (1) is described with (note small letters):

$$c(t)=a_6\cdot t^6+a_5\cdot t^5+a_4\cdot t^4 +a_3\cdot t^3+a_2\cdot t^2+a_1\cdot t+a_0 \quad (8)$$

where: c is unitless serum insulin concentration, t -unitless time in the range of [0,1], then the individual coefficients a_i for each insulin analog can be calculated using the formula:

$$a_i=\frac{A_i\cdot T_{SPAN}^{i+1}}{P} \quad (9)$$

Having the normalized insulin curves (Table 7) it is possible to determine insulin activity functions:

$$D(T)=\frac{DOSE}{T_{SPAN}}\cdot c\left(\frac{T}{T_{SPAN}}\right) \quad (10)$$

where: $DOSE$ is the amout of insulin injected (in any insulin unit). The model of insulin activity contains the following assumptions:

- Source diagrams (Fig. from 2 to 8) present the total consumption of the injected dose.
- Insulin serum concentration is directly proportional to the active fraction of the whole dose.
- The insulin dose size does not affect its profile.

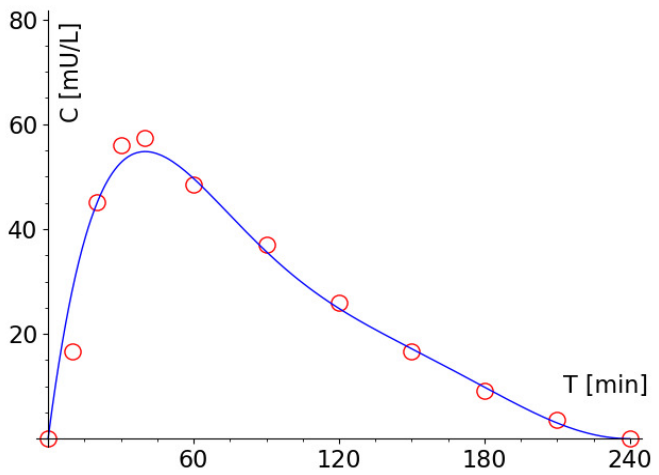


Fig. 10. Polynomial fitting for NovoRapid: T_{PEAK}= 40, C(T)= -1.034×10^{-11} T^6 $+1.035\times10^{-8}$ T^5 -4.076×10^{-6} T^4 $+8.018\times10^{-4}$ T^3 -0.08134 T^2 +3.576 T, P=6048.19

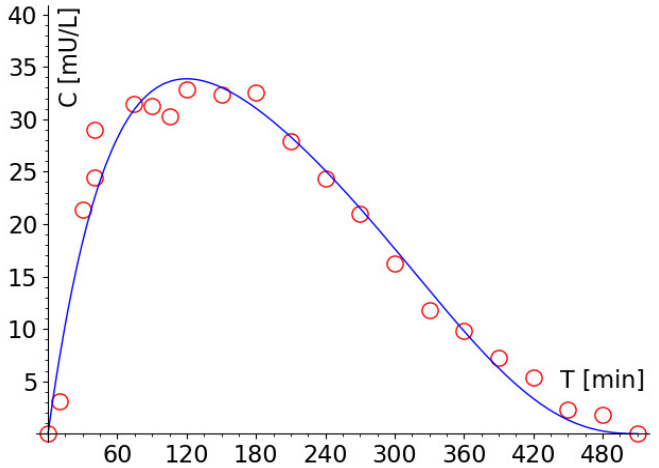


Fig. 11. Polynomial fitting for Actrapid: T_{PEAK}= 120, C(T)= -5.669×10^{-14} T^6 $+1.054\times10^{-10}$ T^5 -7.818×10^{-8} T^4 $+3.064\times10^{-5}$ T^3 -0.006997 T^2 +0.7952 T, P=9190.88

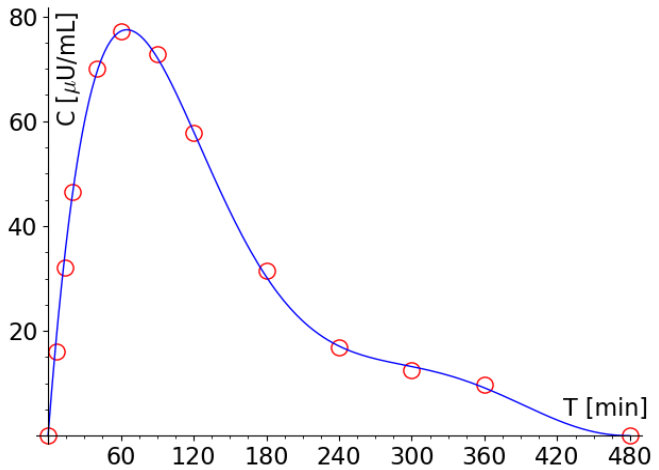


Fig. 12. Polynomial fitting for Apidra: T_{PEAK}= 65, C(T) = -3.638×10^{-13} T^6 $+7.160\times10^{-10}$ T^5 -5.521×10^{-7} T^4 $+2.090\times10^{-4}$ T^3 -0.03926 T^2 +3.000 T, P=13354.3

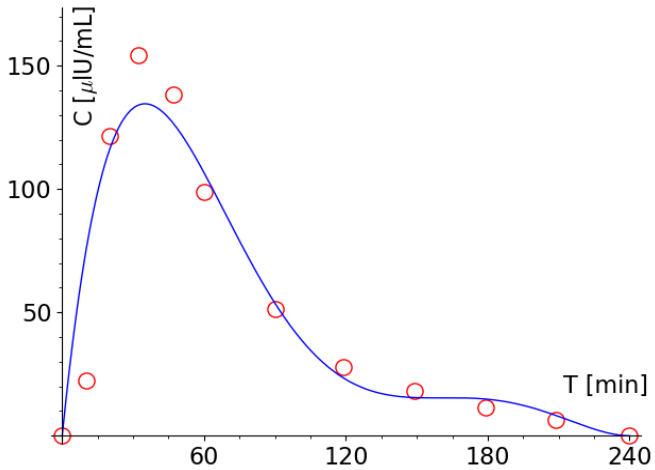


Fig. 13. Polynomial fitting for Insuman Implantable: T_{PEAK}= 35, C(T)= -1.351×10^{-11} T^6 $+1.846\times10^{-8}$ T^5 -9.040×10^{-6} T^4 $+2.060\times10^{-3}$ T^3 -0.2239 T^2 +9.519 T, P=11320.0

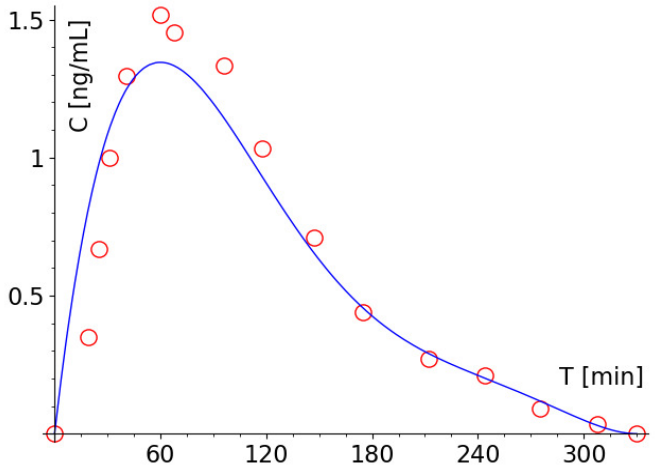


Fig. 14. Polynomial fitting for Humalog: T_{PEAK}= 60, C(T)= -1.670×10^{-20} T^6 $+9.683\times10^{-12}$ T^5 -1.040×10^{-8} T^4 $+4.214\times10^{-6}$ T^3 -0.0007752 T^2 +0.05588 T, P=192.583

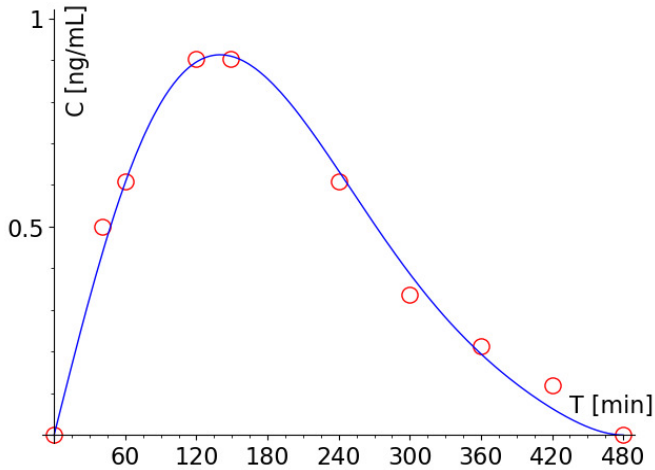


Fig. 15. Polynomial fitting for Humulin R Short-acting: T_{PEAK}= 140, C(T)= $+2.296\times10^{-15}$ T^6 -3.739×10^{-12} T^5 $+2.227\times10^{-9}$ T^4 -5.197×10^{-7} T^3 $+4.667\times10^{-6}$ T^2 +0.0113 T, P=221.561

4. Results

The most valuable results of this work are the normalized insulin profiles given by polynomial coefficients a_i (Table 7).

The profiles enable standarized comparison of insulin analogs. Moreover they, along with the time ranges of insulin action $T_{SPAN,}$ let formulate the insulin activity model according to (10). The normalized curves have been depicted in the following figures. The Fig.16 presents the curves for the rapid acting insulin analogs under study, whereas the Fig. 17 shows the curves for the short acting analogs. It can be noticed that within the the same group the normalized insulin curves are similar. The differences between them may not go beyond measurement scatter. Let's take into consideration Fig. 2 and Fig. 3. They both show insulin profiles for NovoRapid. Using the data from the figures two different normalized curves for the same analog have been draw up (see Fig. 18). The disagreement is similar to that one seen between different rapid acting analogs (Fig. 16). This observation indicates a possible direction in further searching for insulin activity models. It suggests that it may be possible to use one normalized insulin curve for each group instead of individual for each product.

Example activity functions have been calculated according to (10) for DOSE = 1U (1unit) and presented at Fig. 19. The functions describe the action of 1U of an insulin analog injected at time T_{APP}=0. The figure shows how the analogs activates in a human body over time. In this case different types of insulin can be presented at the same graph, because here unitless time is scaled to real minutes using the given periods of activity T_{SPAN}. Looking at the figure is easy to point out the rapid acting group and the short acting group.

The case of Humalg needs to be commented. Looking at the Fig. 19 one can notice that its curve deviates from the rest of the rapid-acting group. The graph suggests that Humalog is the fastest insulin. The effect is not present in the Fig. 16 because it is the activity time range T_{SPAN} that has decided on the look of the activity function. The range for Humalog (see Table 7) is shorter then ranges of other rapid-acting analogs. The range was read form Fig. 6. The author of this paper recommends caution in this case.

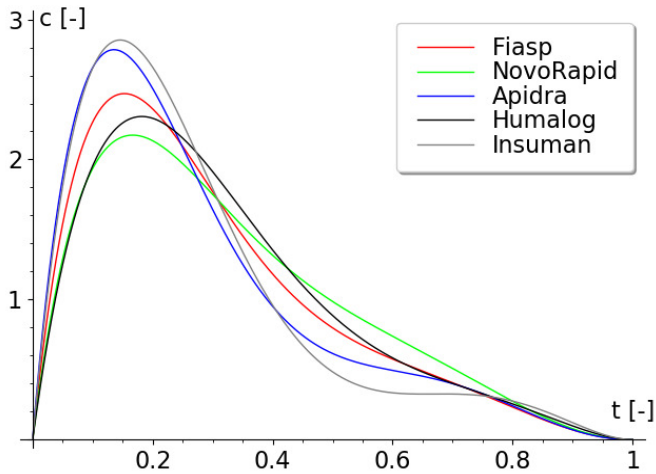


Fig. 16. Normalized insulin curves for rapid acting analogs under study

Table 7. Unitless coefficients a_i (i=1, 2, ..,6, a_0=0 for each case) of normalized insulin curves and time ranges of insulin action in minutes

Insulin	a_6	a_5	a_4	a_3	a_2	a_1	T_{SPAN}
Fiasp	−94.76	+403.4	−675.6	+561.2	−235.5	+41.25	360
NovoRapid	−78.44	+327.1	−536.6	+439.8	−185.9	+34.05	360
Actrapid	−55.35	+201.8	−293.5	+225.5	−101.0	+22.50	510
Apidra	−159.9	+655.7	−1053	+831.0	−325.2	+51.75	480
Insuman Implantable	−54.72	+311.7	−635.9	+603.9	−273.5	+48.43	240
Humalog	+0.000	+64.93	−211.3	+259.5	−144.7	+31.60	330
Humulin R	+60.84	−206.4	+256.1	−124.5	+2.329	+11.70	48

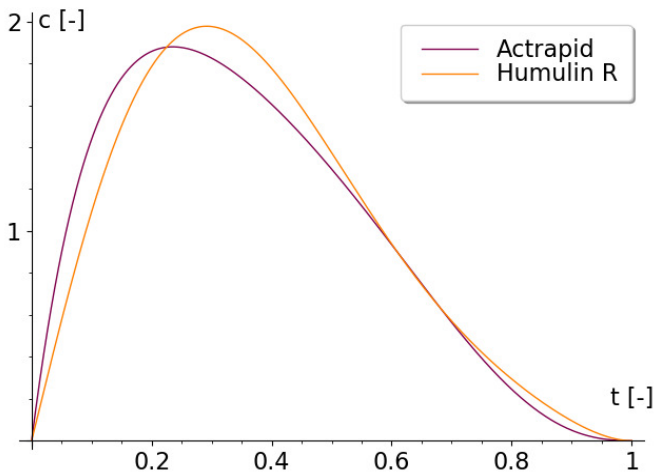


Fig. 17. Normalized insulin curves for short acting analogs under study

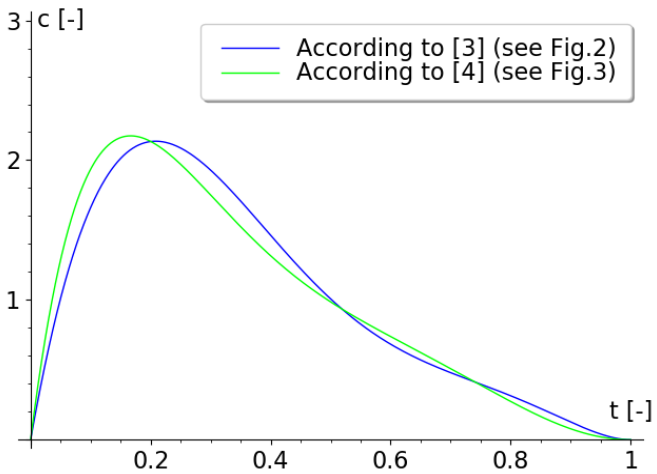


Fig. 18. Insulin profile for NovoRapid acc.to different data sources

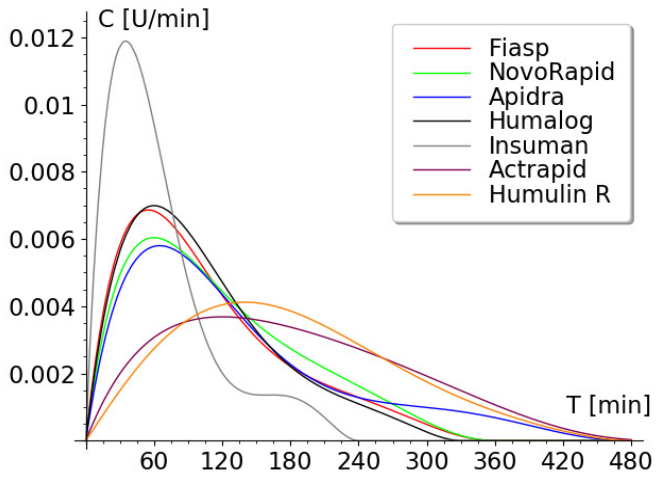


Fig. 19. Insulin activity functions acc.to (10) for the analogs under study after injection of 1U (unit) of insulin analog

5. Conclusion

In this paper 7 normalized insulin profiles have been evaluated based on the product information published by the manufactures. Followingly, the insulin activity model has been propounded. The model consists of a function formula which enables to predict insulin activity in a human body after an injection. The activity model may be used with limited trust (!) in computer aided therapy of type 1 diabetes. Moreover the results have opened a new direction for future research in the field of insulin activity models.

References

[1] OPENAPS.ORG, https://openaps.org, [05.11.2019]

[2] Global Human Insulin Drugs Market - Growth, Trends and Forecast (2019 – 2024), Mordor Intelligence, https://www.mordorintelligence.com/industry-reports/insulin-market, [05.11.2019]

[3] Fiasp : EPAR – Product Information, European Medicines Agency, www.ema.europa.eu/en/medicines/human/EPAR/fiasp, update 30/10/2019, [05.11.2019]

[4] NovoRapid: Product Monograph, Novo Nordisk Canada Inc., www.novonordisk.ca/our-products.html, [28.10.2019]

[5] Actrapid : EPAR – Scientific Discussion, European Medicines Agency, www.ema.europa.eu/en/medicines/human/EPAR/actrapid, update 03/04/2006, [05.11.2019]

[6] Apidra Product Monograph, Sanofi-Aventis Canada Inc., version 14.0 dated October 17 2017, http://products.sanofi.ca/en/apidra.pdf, [05.11.2019]

[7] Insuman : EPAR – Product Information, European Medicines Agency, www.ema.europa.eu/en/medicines/human/EPAR/insuman, update 21/05/2019, [05.11.2019]

[8] Humalog Product Monograph, ELI LILLY CANADA INC., November 28, 2017, http://pi.lilly.com/ca/humalog-ca-pm.pdf, [05.11.2019]

JOURNAL of COMPUTER SCIENCES INSTITUTE

JCSI 13 (2019) 279-282
Wysłane: 2019-12-07
Przyjęte: 2019-12-15

UML – a survey on technical university students in Lublin

Adam Ulidowski, Jan Wrzos, Bartłomiej Włodarczyk, Krzysztof Krocz, Patryk Drozd, Kamil Żyła*
Lublin University of Technology, Department of Computer Science, Nadbystrzycka 36B, 20-618 Lublin, Poland

Abstract. Unified Modeling Language (UML) is a commonly known OMG (Object Management Group) standard for designing software systems. However, practice shows that the usage of UML varies depending on the specificity of a software system and company. The authors decided to explore the perspective of students with experience in using UML by conducting an exploratory survey with them. Analysis of the data gathered revealed that they use UML diagrams as an additional help when developing software. The main risk turned out to be different diagram interpretations. At last, the main motivation to learn UML was obtaining a credit at university.

Keywords: UML; software engineering; information systems modeling; survey

*Corresponding author.

E-mail address: k.zyla@pollub.pl

UML – punkt widzenia studenta uczelni technicznej w Lublinie

Adam Ulidowski, Jan Wrzos, Bartłomiej Włodarczyk, Krzysztof Krocz, Patryk Drozd, Kamil Żyła*
Politechnika Lubelska, Instytut Informatyki, Nadbystrzycka 36B, 20-618 Lublin, Polska

Streszczenie. Unified Modeling Language (UML) jest powszechnie nauczanym i stosowanym standardem Object Management Group (OMG) służącym do opisu systemów informatycznych. Jednakże praktyka pokazuje, że użyteczność UML waha się w zależności od specyfiki projektu i systemu informatycznego. Autorzy zdecydowali się przeprowadzić ankietę badającą opinie studentów, mających styczność z językiem UML, na temat wykorzystania wykonanych w nim diagramów. Analiza zebranych ankiet wykazała, że diagramy UML pełnią zazwyczaj rolę pomocniczą, głównym ryzykiem ich użycia są różnice w ich interpretacji, a główną motywacją do nauki jest uzyskanie zaliczenia na uczelni.

Słowa kluczowe: UML; inżynieria oprogramowania; modelowanie systemów informatycznych; ankieta

*Autor do korespondencji.

Adres e-mail: k.zyla@pollub.pl

1. Introduction

Unified Modeling Language (UML) is a widely taught and applied Object Management Group (OMG) standard for the description (modeling, design) of IT systems. One of the main goals of its creation is to support the IT industry by providing a graphical general purpose modeling language, which could be used for business process modeling, as well as analysis, design, documentation and implementation of IT systems. It is used in such areas as banking, healthcare, aerospace, etc. Particularly interesting is the last field, where it is used to simulate and ensure the security of complex real-time systems. Such a specificity of applications requires appropriate standardization and interoperability of UML, which has been achieved, among others, by the formal definition of the metamodel using MOF, the technology-independent definition of semantics, as well as user-oriented syntax. [10]

UML is a constant element of the color of the industry that deals with aspects of producing all kinds of software, ranging from universities to enterprises. It is worth mentioning a study [8], in which UML is indicated as the most commonly used modeling language by university employees and software companies. It is also worth quoting M. Petre [6], who is quite critical of the universality of UML, but the way it is discussed confirms the important role of UML and its existence in the awareness of the IT community. Nor can it be denied an important role in model-driven engineering, although not all types of diagrams are used.

Issues of UML applications in IT projects have already been raised by other authors. Some characteristic works are presented below. In [4], two methods of modeling IT systems, based on user stories and UML, are compared, and a ranking of diagrams created in terms of the degree of use. The results shown partly coincide with the findings described in this article. In [2], research on the use of particular UML diagrams is presented, and what is particularly valuable is the question of the reasons for their non-use. The authors also point out that diagrams may be ambiguous in their interpretation, and their misunderstanding may constitute a significant barrier to their use. [5] describes research on the usefulness of specific UML diagrams in open source projects (10 such projects are included). It turns out that class diagrams are used most often, and much less frequently use case diagrams and activity diagrams. In [7], the results of surveys on the knowledge of UML diagrams, the source of knowledge about them and the objectives of their application are presented. The authors have compiled the data in terms of gender and job title. In [9], the concept of a lighter version of UML is discussed. According to the researchers, students learning UML are overwhelmed by the excessive complexity and multitude of different diagrams. A particularly interesting work is [1] – it is a compilation of studies on the use of UML over the past 15 years. The author addresses the issues of effectiveness, profits and costs of UML use, as well as the correctness of the research methodology. The results of examination with an eye-tracker are also presented. Finally, in [3], the authors conduct a thorough research, posing important questions about the degree of

adoption and the adequacy of UML diagrams to the code in open source projects. Of note is a relatively small percentage (about 1/3 of cases) where the UML model has been implemented in its entirety, without any changes. The authors also discuss the real help provided by diagrams for new employees and how diagrams help in the implementation of an IT system.

Literature analysis has shown great interest in the subject, which is dictated to a large extent by UML popularity in production of IT systems. In our opinion, it is worth exploring the perspective of technical university students who have contact with the UML language during their studies or professional work. This publication is a kind of insight into the state of the Lublin market, and the conclusions can serve as a guide for those who teach subjects thematically related to software engineering.

2. Study aim and methods

In view of the above, it was decided to carry out an exploratory survey whose questions were grouped in the following categories: profile of respondents, professional experience, UML diagram skills, attitude towards UML diagrams. The results and the questions will be discussed in the next chapter.

The following research questions were formulated:

RQ1. What is the main motivation for students to learn UML?

RQ2. Are UML diagrams frequently used?

RQ3. What are the challenges and opportunities when using UML?

The survey was then disseminated to technical university students in Lublin who learned the UML language. Although the geographical scope of the survey was local, one should not ignore the globalization of thoughts, ideas and the process of software development within IT enterprises. Nevertheless, authors of this article do not intend to generalize the results of survey on the whole population.

3. Research results

This chapter presents the results of the survey, which was answered by 79 people. The survey was started in 2018.

3.1. Education and professional experience of respondents

Almost two thirds of the respondents completed first degree studies (50 people). One third of the respondents was finalizing second degree studies (26 people). The survey was also attended by 3 people who had not yet obtained a university degree, although had a commercial experience in IT. All persons declared studies in computer science or a related course of study.

As a rule, people participating in the survey had a low experience in the IT industry: 24 people were employed for one year, 13 people for 2 years, 7 people for 3 years, 7 people for 5 years. The remaining people did not provide such data in the survey.

Over 12% of respondents (10 people) declared lack of contact with commercial IT projects. They were excluded from further analysis of professional experience of respondents. An IT project is understood as a process of an IT system development basing on the requirements of the person who ordered its creation. Most of the respondents participated in 1-3 commercial projects (42 people). In general, a large percentage of respondents is professionally active, which is typical for the IT industry. See Table 1.

Table 1. Number of commercial projects in which respondents participated

Number of projects	Number of respondents	Percentage of respondents
0	10	12.7 %
1	10	12.7 %
2	16	20.3 %
3	15	19.0 %
4	6	7.6 %
5	5	6.3 %
6	3	3.8 %
7	1	1.3 %
8	1	1.3 %
9	5	6.3 %
>9	7	8.9 %

Table 2 presents a typical size of a team developing an IT system in which the respondent participated. It should be noted that the most frequently (24 times) indicated number concerned 3-8 people, while 17 indications referred to the number of 8-10 people. The results presented in Table 2 may be influenced by the respondents' small professional experience, as well as the specificity of modern software development methodologies, e.g. Scrum.

Table 2. Typical number of project team members

Typical size of the team	Number of respondents	Percentage of respondents
1-3 persons	15	21.7 %
3-5 persons	12	17.4 %
5-8 persons	12	17.4 %
8-10 persons	17	24.6 %
>10 persons	13	18.8 %

Table 3 presents a summary of responses for which business sectors (more than one could be indicated) the respondent usually participates in software development. The sectors that clearly lead the way are: medicine, telecommunications, trade, transport and finance. Other sectors include: catering, clothing, politics, law and legislation, energy, entertainment and game development. Moreover, on the basis of the survey results, it can be concluded that the most frequently mentioned business sectors employ students who declared good theoretical knowledge of UML diagrams. However, only half of the respondents admit that they use these diagrams in practice.

Table 3. Software development by business sector

Sector name	Number of responses	Percentage of respondents
Medicine	18	26.1 %
Telecommunication	12	17.4 %
Economy	13	18.8 %
Trade	16	23.1 %
Transport	14	20.3 %
Finance	18	26.1 %
Others	20	29.0 %

3.2. Knowledge of UML diagrams among respondents

In this section, the analysis did not exclude respondents who declared a lack of commercial experience.

Table 4 shows which UML diagrams are known among respondents. Each respondent could indicate more than one diagram. The diagram of particular type is known to a person, when the person is able to read and create it. The most commonly known were use case and class diagrams. This may be due to the fact that the specificity of their construction is conducive to easier programming of IT system modules. The next places were taken by activity and sequence diagrams, and finally package, component and state diagrams.

Then the respondents were asked to indicate the diagrams they used in IT projects. The most frequently indicated were use case and class diagrams – almost half of the respondents. The following places were taken by activity, component, state, sequence and package diagrams. Numerical values seem to confirm the result of literature analysis, which states that the knowledge of diagrams does not go hand in hand with the universality of their use. Details can be found in Table 5.

Table 4. UML diagrams known to respondents

Diagram name	Number of responses	Percentage of respondents
Class	55	69.6 %
Use case	53	67.1 %
Sequence	38	48.1 %
Activity	35	44.3 %
Component	25	31.6 %
State	22	27.8 %
Package	18	22.8 %
Others	0	0.0 %

Table 5. UML diagrams used by respondents

Diagram name	Number of responses	Percentage of respondents
Use case	24	30.4 %
Class	19	24.1 %
Activity	16	20.3 %
Component	9	11.4 %
Sequence	7	8.9 %
State	7	8.9 %
Package	4	5.1 %
Others	0	0.0 %

Respondents also indicated the most frequently used editors to create UML diagrams. Table 6 contains a list of them. Visual Paradigm was the most popular. This may be caused by its availability at universities (graduates additionally transfer their habits from studies to professional work), as well as a high degree of compliance with the UML standard published by OMG. The second most important editor was Microsoft Visio, which is quite puzzling considering the specificity of its work. Perhaps the good result is caused by its availability. A high position of Microsoft Visio was also noted in study [8]. It is worth noting that modeling capabilities integrated with programming environments, such as Eclipse and NetBeans, are not used very much. Others category included StarUML and Enterprise Architect.

Table 6. Editors used to create UML diagrams

Editor name	Number of responses	Percentage of respondents
Visual Paradigm	31	39.2 %
Microsoft Visio	11	13.9 %
PlantUML	4	5.1 %
yEd	4	5.1 %
UML plug-in for Eclipse	2	2.5 %
UML plug-in for NetBeans	2	2.5 %
UML plug-in for IntelliJ IDEA	2	2.5 %
draw.io	2	2.5 %
Others	4	5.1 %

The main motivation for learning how to create UML diagrams was indicated as: obtaining a credit at university (70.1% – 56 people), acquiring skills necessary for professional work (12.7% – 10 people) and the need for self-development (6.3% – 5 people). 8 people (10.1%) were not interested in learning UML. The main motivation for learning was to complete the course; however, the knowledge of UML, although to a lesser extent than it is taught at universities, is useful when working on commercial projects of IT systems.

3.3. Opinion of respondents on UML

In this section, the analysis did not exclude respondents who declared a lack of commercial experience.

First of all, it was asked to what extent UML diagrams support the implementation of IT systems. The range of the scale of assessments was from 1 (useless) to 5 (necessary). The results are presented in Table 7. It may be stated that the respondents have different opinions on the usefulness of UML diagrams in designing IT systems. On the one hand, 30 people (38.0%) indicated high usefulness (grades 4 and 5), however, 41 people (51.9%) declared average usefulness (grades 2 and 3). Certainly, it can be seen that there is a certain dissonance between the theory of software engineering and good practices of software modeling, and the practice applied in IT companies. In addition, it may be suspected that this is a symptom that the failure to attach importance to the UML diagramming does not have a significant negative impact on the IT systems development. This is also noted by the authors quoted in the literature review.

Respondents were then asked whether they see benefits of using UML diagrams. 14 people (17.7%) said that UML diagrams were not beneficial and unnecessary. 21 people (26.6%) said the benefits were high. As many as 44 (55.7%) believed that the benefits were only minor. This may be due to the amount of time and resources needed to create UML diagrams. The second reason may be the scale of the projects the respondents work on. If it is a small team (2-5 people), the diagrams will not be as important. Another case is large teams (50-100 people), where people are divided into groups dealing with specific aspects of the IT system. Then UML diagrams help to illustrate the expectations towards particular groups and the software created.

Table 7. Assessment of the extent to which UML supports software development

Grade	1	2	3	4	5
Number of responses	8	20	21	24	6
Response rate	10.1 %	25.3 %	26.6 %	30.4 %	7.6 %

The next issue concerned the problems with creating UML diagrams. It turned out that 21 respondents (26.6%) noticed problems with creating diagrams even in people with long working experience. 20 respondents (25.3%) noticed such problems in people with low work experience, while 20 people (25.3%) did not notice that anyone around them had problems with creating UML diagrams. 18 people (22.8%) did not comment on this issue.

The respondents were then asked whether they consider UML diagrams to be clearly interpretable. As many as 47 respondents (59.5%) said they had problems interpreting UML diagrams in an unambiguous way. 24 people (30.4%) noticed that each person creates their own style and after understanding it the diagrams become unambiguous. A small number of respondents (10.1% – 8 people) stated that there was no problem with interpreting UML diagrams unequivocally from the start.

In response to the question whether UML diagrams reduce the risk of errors, more than half of the respondents (57.0% – 45 people) were willing to believe in the positive impact of UML diagrams on error avoidance. Every fourth person (24.1% – 19 people) believed that these diagrams had little impact. Only 10 people (12.7%) said that diagrams were important to avoiding making mistakes, while 6 respondents (7.6%) were convinced that there was no connection between the two issues.

Finally, respondents were asked if they believe that UML diagrams shorten the time of project implementation. Most of the respondents (51.9% – 41 people) were not sure about the impact of UML on saving time during project implementation. Every third opinion maker (34.2% – 27 people) was convinced of the positive impact of UML. In turn, 11 people (13.9%) said that UML diagrams had no significance in this respect.

4. Summary

The survey showed that UML diagrams are used in IT projects, but this is not the rule. People have different views on how to use them, but generally there is no common belief in their usefulness. They are often treated only as additions or are replaced by hand-written diagrams or diagrams in other notations, including informal ones. In addition, some respondents said that no diagrams were needed, a verbal description being enough. (RQ2)

According to the majority of the respondents, a broad knowledge of UML diagrams is necessary, first of all in order to be able to obtain a credit at university. In a professional career such broad knowledge is not always necessary, because, as shown by the survey, UML diagrams are not always used regularly in companies. (RQ1)

The results also showed that according to most respondents, the interpretation of UML diagrams is not clear and that designing systems with their help does not have any impact on a significant reduction in the risk of errors and acceleration of work on the implementation of the project. This also translates into a reduction in the frequency of using UML diagrams in IT projects and their frequent treatment as redundant, because they do not really contribute much to the project. (RQ3)

At last it cannot be forgotten that opinions gathered during the survey originate from persons with relatively low commercial experience and related to the Lublin region. Thus this paper only shows their perspective on using UML during a software development process, that not necessarily fully complies with the IT industry reality.

References

[1] Chaudron M. R. V.: Empirical studies into UML in practice: Pitfalls and prospects. Proceedings of the 9th International Workshop on Modelling in Software Engineering, MISE ’17, Buenos Aires, Argentina, May 20-28, 2017.

[2] Dobing B., Parsons J.: How UML is used. Communications of the ACM, 49(5)/2006, 109-113.

[3] Ho-Quang T., Hebig R., Robles G., Chaudron M. R. V., Fernandez M. A.: Practices and perceptions of UML use in open source projects. 2017 IEEE/ACM 39th International Conference on Software Engineering: Software Engineering in Practice Track (ICSE-SEIP), Buenos Aires, Argentina, May 20-28, 2017, IEEE 2017, 203-212, [DOI: 10.1109/ICSE-SEIP.2017.28].

[4] Madanayake R., Dias G. K. A., Kodikara N. D.: Use stories vs UML use cases in modular transformation. International Journal of Scientific Engineering and Applied Science, 3(1)/2016, 50-54.

[5] Osman H., Chaudron M. R. V.: UML usage in open source software development: A field study. International Workshop on Experiences and Empirical Studies in Software Modelling (EESSMOD 2013), MODELS, 23-32.

[6] Petre M.: “No shit” or “Oh, shit!”: responses to observations on the use of UML in professional practice. Software & Systems Modeling, 13(4)/2014, 1225–1235.

[7] Reggio G., Leotta M., Ricca F.: Who knows/Uses what of the UML: A personal opinion survey. Model-driven engineering languages and systems, LNCS, 8767/2014, Springer, 149-165.

[8] Störrle H.: How are conceptual models used in industrial software development?: A descriptive survey. Proceedings of the 21st International Conference on Evaluation and Assessment in Software Engineering, EASE’17, Karlskrona, Sweden, June 15-16, 2017, ACM, 160-169, [DOI: 10.1145/3084226.3084256].

[9] Wrycza S., Marcinkowski B.: A light version of UML 2: Survey and outcomes. Proceedings of the 2007 Computer Science and IT Education Conference, University of Technology Mauritius Press, 2007, 739-749.

[10] Object Management Group, Inc.: Unified Modeling Language Specification, Version 2.5. Adres: http://www.omg.org/spec/UML/2.5/ (formal-15-03-01.pdf). [20.11.2019]

JOURNAL of COMPUTER SCIENCES INSTITUTE

JCSI 13 (2019) 283-287
Wysłane: 2019-06-18
Przyjęte: 2019-06-27

Overview of Big Data platforms

Gabriel Wróbel [a,b,*], Maciej Daniel Wikira [a]
[a]Institute of Computer Science, Lublin University of Technology, Nadbystrzycka 36B, 20-618 Lublin, Poland
[b] University of Oulu, Oulu, Finland

Abstract. The primary purpose of this paper is to present and provide main advantages and disadvantages of most popular big data platforms as well as their comparison in terms of ease of installation, work, performance and price, in order to find the most suitable solution to work with big sets of data. Nowadays, the data is largely analyzed by scientists not related to IT, so the ease of use and presentation of data is extremely important. The purpose of the assessment was to indicate the best IT tool for analyzing data from the point of view of a young analyst or scientist graduating and entering the labor market.

Keywords: big data; data analysis; platform; tool comparison

*Autor do korespondencji.

E-mail address: g.wrobel@pollub.edu.pl

Przegląd platform Big Data

Gabriel Wróbel [a,b,*], Maciej Daniel Wikira[a]
[a]Politechnika Lubelska,Instytut Informatyki, Nadbystrzycka 36B, 20-618 Lublin, Polska
[b]Uniwersytet Oulu, Oulu, Finlandia

Streszczenie. Głównym celem niniejszej pracy jest prezentacja głównych zalet oraz wad najbardziej popularnych platform big data, jak również porównanie ich pod względami łatwości instalacji, funkcjonalności, wydajności oraz ceny co pozwoli na wskazanie rozwiązania najlepiej dostosowanego do pracy z dużymi zbiorami danych. W dzisiejszych czasach dane są przetwarzane przez analityków niezwiązanych z branżą IT, w związku z czym bardzo istotne są kwestie łatwości użytkowania i prezentacji danych. Celem oceny jest wyznaczenie najlepszego narzędzia z branży IT dla analizy danych z perspektywy młodego analityka lub naukowca kończącego edukację i wchodzącego na rynek pracy.

Słowa kluczowe:big data; analiza danych; platforma; porównanie narzędzi

*Autor do korespondencji.

Adres e-mail: G.wrobel@pollub.edu.pl

1. Introduction

For decades, data was being collected from large number of websites, devices and sensors that can be used to carry out analyses in various fields of science and life. Banking, telecommunications, tourism, insurance, e-business, and energy –these are just some of the industries in which Big Data analysis is presentnowadays. In case of banks, modern tools allow, for example, toexamine the age of customers who use credit cards most often.

Big Data makes it possible even to examine monthly averagebills of customers who have given up their services. For the analysis of big datasets,big set of data analysis software tools is currently being used.These tools themselves function very well but can be problematic for users who are not familiar with the technology. To facilitate the work with large data sets, big data platform solutions have been developed that are designed to facilitate the use of analysis tools and increase the efficiency of working with big data. Platform data isa collection of tools that allows complex data analysis, machine learning, data storage and visualization. There is a possibilityto use solutions from various companies that can operate locally, such as Cloudera, Hortonworks, MapR Platform, as well as cloud solutions such as Amazon AWS, GoogleCloud, Microsoft Azure. Platforms Cloudera, Hortonworks, MapRand HDInsight (Microsoft Azure).Main advantages and disadvantages of these solutions have beenpresented in this article as well as their comparison based on difficulty of installationand price[6].

2. Big Data Tools

To clearly understand why platforms are so important in working with big data,it is important to consider what features platform can include. Therefore, Hadoop is introduced as the most typical set of tools.

The simple definition of the Hadoop operating principle is saving files and processing data. It is easy to imagine afile larger than the disk capacity of a standard PC. By thetraditional way, it is not possible to save such a file. Hadoop allows one to save files larger than the common disk capacity, so they can be stored on a given server or matrix, by distributing data to multiple clusters (matrices, servers). The second element of Hadoop is the ability to process this data. If a huge amount of data should be made available, traditionally it must be sent to a local device, which usually cannot cope with this task. Hadoop reverses this situation and, thanks to the MapReduce component, moves the processing tools towards the data[5].

Hadoop is developed by the Apache Software Foundation. An important element of Hadoop is HDFS. HDFS (Hadoop Distributed File System) is a technology that provides effective scaling of the storage layer. HDFS is a Java-based file system that has been adapted to work even on hardware with little capacity. Another important element of Hadoop is

the Data Processing Framework, which in the form of MapReduce is used to work with data. MapReduce runs a series of processes, each of which is a separated Java application that searches data. It is worth to highlight that queries are not being used in this case as they are in a relational database. In Hadoop, there are tools such as Hive (initially developed by Facebook) that allowsone to convert the query language into MapReduce tasks[5].

Nowadays, in data platforms solutions like Spark or YARN (Yet Another Resource Negotiator) can be found, which are in fact new generations of Hadoop and MapReduce solutions.This solves some limitations, but the main principles remain the same [5].

3. Platform Presentation

A. MapR

The MapR is an enterprise-class distribution for the Hadoop and Spark platforms. The MapR platform has been designed to provide the highest ease of use, performance, and reliability for users who need tools to analyze large data sets. The solution includes a full set of tools necessary for work. It is possible to work using a file system designed specifically for MapR (MapR-FS).In addition, it isalso possible to manage the documentary database MapR-DBand work using the streams of MapR. All the tools available in the Hadoop environment can be used, so there is alsoan opportunity to use HDFS file system or mapReduce. In case of MapR there is an alternative to mapReduce, as the platform supports the YARN architecture that manages files and tasks between clusters[1].

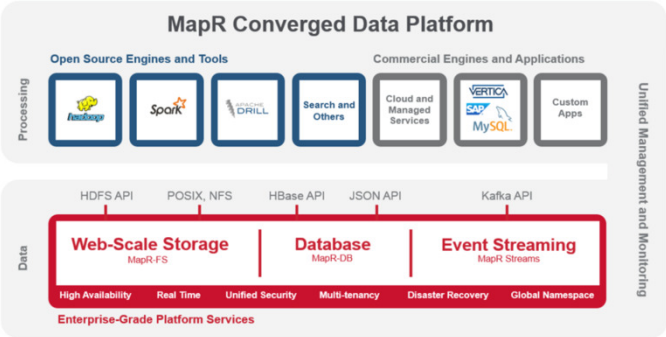


Fig.1. Visualization of MapR Data Platform architekture

Figure 1 shows the architectural layout of the tools contained in the MapR data platform as well as components and links between them[1].

MapR platform introduces snapshots that are remembering the situation of data at the specifiedmoment. This solution allowsto use the memory effectivelywhile keeping only the changes from the last snapshot. The abovementioned solution minimizes the possibility of data loss in the event of a failure.

MapR has a strong security based on the user's authorization, which occurs each time the user attempts to access the file. The platform checks whether a given user has access rights to the requested document[1].

MapR does not use name nodes to store file location information. Instead it provides high availability for MapReduce JobTracker and Direct Access NFS via server CLDB[1].

B. Cloudera

Cloudera provides a fully integrated and scalable platform designed to work with large and constantly growing, diverse data sets. The products and solutions introduced by Cloudera enable working with the Apache Hadoop tool and related tools for manipulating, analyzing, protecting and securing data[2]. The service architecture is shown on figure 2.

Cloudera provides the following products and tools:

- CDH – complete distribution of Apache Hadoop as well as other open-source projects such as Apache Impala or Cloudera Search. CDH is a solution, related to data security, providing software integration and hardware integration[2].
- Apache Impala – solution which enables parallel work with data using the SQL engine which in combination with Apache Hadoop makes available a wide range of BI capabilities. This is possible thanks to highly optimized architecture. The tool allows to direct SQL queries to files in the HDFS file system, divided between clusters using MapReduce or loaded into Hive tables. Impala has a YARN resource management component that makes it possible to construct SQL queries on a working cluster. Impala management is possible from the level of Cloudera Manager and secure the sentry framework[2].
- Cloudera Search –solution that allows searching the data placed in Hadoop or HBase systems in nearly real time. The search provides batch indexing, parallel text search, but does not require skills related to programming or skills related to the construction of SQL queries. Cloudera search is a fully scalable, flexible solution, and is included in the CDH. With this tool, we do not need to transfer data to perform business tasks[2].
- Cloudera Manager –the application used to monitor, manage, and diagnose problems occurring in the CDH platform. The solution works using the administrator or web client, which makes managing the platform much easier and more intuitive. Cloudera Manager contains an API with the help of which we can configure the Cloudera manager and also get information on the condition of clusters or matrices, and use this data in our own monitoring applications[2].
- Cloudera Navigator – complete tool for data management and data protection in the CDH platform. The solution enables administrators and analysts to explore data placed in Hadoop. Cloudera navigator makes it easy for businesses to store data according to the law through audits, data management, life cycle management and data encryption management [2].

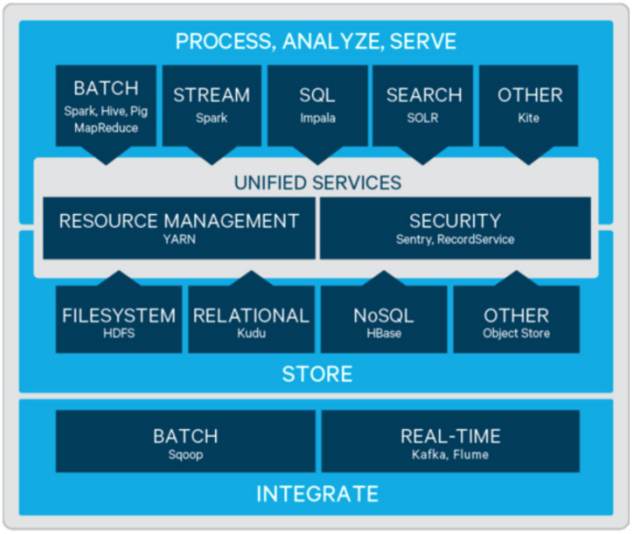


Fig. 2. Visualization of CDH Data Platform architekture.

C. Hortonworks

Hortonworks Data Platform (HDP) is an open support platform for Apache Hadoop, which provides a stable foundation for developing big data solutions in the Apache Hadoop ecosystem. HDP strongly focuses on container architecture which makes the solution more flexible and easy to scale. HDP includes tools such as TechPreview, TensorFLow, Apache Zeppelin or Apache Spark, which enablesapossibility to use machine learning and deep learning.These solutions can be very helpful in long-term data analysis. Moreover, the platform provides the possibility of deep learning through GPUs, which significantly streamlines the process[3].

Hortonworks introduces hybrid architecture so that the platform can be used in the cloud solution as well as on-premises[3]. The visualization of such architecture is presented on figure 3.

HDP provides higher availability of data with multiple name nodes, at significantly lower TCO with Erasure Coding.Erasure Coding enables certain features for data protection which until now have mostly been found in object data stores [3].

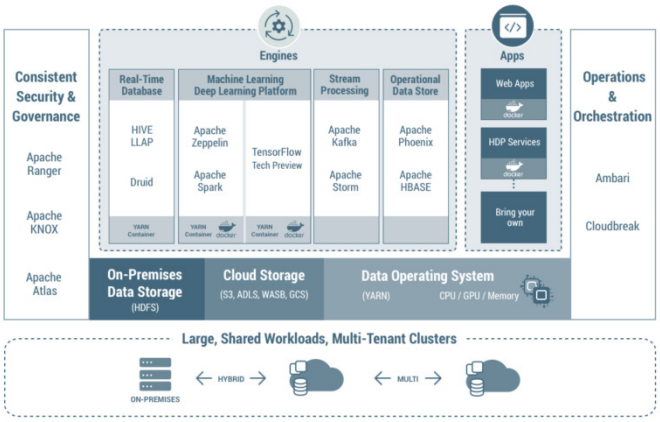


Fig.3. Visualization of HDP Data Platform architekture

In the last quarter of 2018, the Hortonworks and Cloudera corporations announced a merger, which meant that the two most competing solutions will now create one new solution [3].

D. HDInsight (Microsoft Azure)

Azure HDInsight is a cloud distribution of Hadoop service components from the Hortonworks Data Platform (HDP) platform. Azure HDInsight facilitates and speeds up the processing of huge amounts of data. It can be used with the most popular open source platforms such as Hadoop, Spark, Hive, LLAP, Kafka, Storm, R and more. With these platforms, implementations of various scenarios related to extracting, transforming and loading data, data storage, machine learning and the Internet of Things (IoT) are possible [4].

Using HDInsight, it is possible to perform interactive petabyte queries against structured or non-structured data in any format,and also to create models that combine them with business analysis tools.

Using the HDInsight service, streamed data can be processed, and received in real time from various devices [4].

There are following cluster types in the HDInsight service:

- Apache Hadoop –a platform using HDFS, YARN resource management, and a simple MapReduce programming model for parallel processing and analysis of batch data [4].
- Apache HBase – NoSQL database based on the Hadoop platform that provides random access and high consistency for large amounts of unstructured and partially structured data – potentially billions of rows multiplied by millions of columns[4].
- ML Service – server designed for hosting and managing parallel, distributed R processes. It enables data analysts, statisticians, and R language programmers to access scalable, distributed analysis methods in the HDInsight service[4] on-demand.
- Apache Storm – a distributed computing system that works in real time for a fast processing of large data streams. Storm is offered as a managed cluster in the HDInsight service[4].
- Apache Interactive Query preview (AKA: Live Long and Process) – cache memory in the memory for interactive and faster execution of Hive queries[4].
- Apache Kafka – open source platform that is used to create pipelines of streamed data, which also provides applications to handle this data. Additionally the Kafka platform includes a message queue function that allows users to publish and subscribe to data streams [4].

HDInsight is a comprehensive solution that does not require a technical person on the client's side because the support of all services remains within the service provider (Microsoft Azure). Due to the fact that HDInsight is a solution maintained in the cloud, all data is transferred to the servers of the service provider which relievesthe client of the need to have hardware architecture[4].

4. Platform Comparison

Undoubtedly, the big data platform solution makes it much easier to work with large data sets, but it is worth considering which platform should be used for that purpose. The following

paragraph will present several aspects that can facilitate the decision.

A. Knowledge of computer science

Knowledge in the field of computer science can be crucialin this case. If the user needs to perform more complex analytical activities and does not have system administration skills, he may need the help of a technical person. Such situationsgenerate higher investment costs. Knowledge about information systems refers mainly to the LINUX system in this context.The reason for that is the fact, that most platforms were designed to work on UNIX systems[5, 6].

In this case, it may be meaningful to compare the way and the complexity of preparing the work environment in which each platform operates. This comparison is shown in table 1.

Table 1. Table of installation complexity

Complexity of installation	
Cloudera	Installation and configuration of the CDH platform with the manager is an intermediate administrative task in which the installer will require knowledge of the UNIX system commands and basic knowledge in the field of operating system administration. Cloudera also provides images of virtual machines for a quick start of work as well as a file dockerfile through which a container can be run with an already configured platform. However, it is not a solution for enterprise and only for educational purposes[2 ,3].
MapRon-premise	Installation and configuration of the MapR platform is a complex task in terms of operating system administration skills. The producer provides a package with software in libraries available from the LINUX terminal level. The installation requires familiarity with the UNIX family of command systems and basic knowledge in the field of operating system administration[1].
MapR cloud	Customer is provided with an interactive form to choose the size of the solution he needs, depending on the price. Simple and basic service that does not require knowledge in the field of operating system administration[1].
HDInsight (Microsoft Azure)	The customer is provided with an interactive form to choose the size of a solution he needs, depending on the price.Simple operation providing a fairly extensive range of configuration options. The entire configuration does not require knowledge of the administration of operating systems and provides numerous configuration options[4].

In the comparisonabove, in terms of skill requirements and knowledge demanded from the client, HDInsight is on the lead, but right behind it, MapR is an equally interesting solution because it provides the opportunity to work in the cloud where the responsibility for administrative issues lies within the service provider. The CDH platform from Cloudera with Hortonworks comes up the worst in the given list since it is expects knowledge from the user and does not provide the tool.

B. Law Aspect

The usefulness of the solution is always contingent on the task to be performed and, in this case, some solutions may have advantages in one of the cases and be completely useless in another one. An example of such a phenomenon can be sensitive data that cannot be stored outside the state due to law provisions in a given country. In this case, the cloud solution may be unhelpful, because it may happen that a cloud service provider does not have a servers in a given country and the client is forced to invest in his own solution.Here, MapR has solution for working on AWS but also on-premises,what makes this platform very flexible. Cloudera solutions with Hortonworks can be more useful in such situation than solution introduced by Microsoft Azure (HDInsight) but it is not so flexible as MapR[1, 4].

C. Pricing

All producers analysedin this paper have plans for additional financing. The simplest and the cheapest offer packages were taken into account for the balancing of the statement.

In this case, Cloudera with Hortonworks offers its platform for commercial software solution for $ 2,000 per year, and also provides this platform for free for educational purposes for 60 days. MapR on-premises is available for free on trial license for 30 days and there is alsoan option of downloading the sand-box on one'sPC. The enterprise version is paid, and for a specific price,a contact with the producer must be made. For a cloud solution from MapR in the community version, the producer gives a price of $ 2.40 for the cluster's hour of work while working in the AWS cloud. HDInsight from Microsoft Azure does not provide a trial version. It is possible to ordera paid service from the cluster's working hours, which is about 10 € per hour and there is an option to enable and disable the service depending on your needs[1, 2, 4].

Regardless of the fact that the components of the platforms are mostly available under an open source license, the most reasonable solution seems to be Cloudera, which makes the software available in an annual subscription, regardless of use. Other producers do not quote prices on the commercial market or count the price per hour of work,which is normal in a cloud environment. However, in the cloud environment the leader turns out to be a MapR.

5. Summary

Nowadays, while working with large data sets,it is almost indispensable to see correlations in all areas of research.

With the help of open platforms, efficient work is possible, even for people who are not connected with computer science.

Platforms nowadays are very close to each other and very often overlap and cooperate with each other. It is very difficult to choose the right and, at the same time, the best one to work with Big Data because each of them has its advantages and disadvantages in various respects.

Bibliography

[1] MapR Producent website https://mapr.com/docs/51/MapROverview/c_overview_intro.html*(website)* [05/2019]

[2] Cloudera Producent website https://www.cloudera.com/documentation/enterprise/5-13-x/topics/introduction.html *(website)* [05/2019]

[3] Hortonworks Producent website https://hortonworks.com/products/data-platforms/hdp/ *(website)* [05/2019]

[4] Microsoft Azure Producent website https://docs.microsoft.com/pl-pl/azure/hdinsight/hadoop/apache-hadoop-introduction *(website)* [05/2019]

M. Siudziński, Hadoop article https://itwiz.pl/hadoop-czyli-przetwarzanie-rozproszone-open-source/*(website)*[05/2019]

[5] R.Wasiluk, P.Muryjas: The assessment of usefulness modern IT tools of data analysis Big Data, Institute of Computer Science, Lublin University of Technology, 2017

JOURNAL of COMPUTER SCIENCES INSTITUTE

JCSI 13 (2019) 288-292
Wysłane: 2019-06-19
Przyjęte: 2019-06-27

Rozwiązania do zarządzania projektami informatycznymi w chmurze

Grzegorz Szydłowski*
Politechnika Lubelska, Instytut Informatyki, Nadbystrzycka 36B, 20-618 Lublin, Polska

Streszczenie. W artykule omówiono pojęcie zarządzania projektem informatycznym, jego przestrzeń realizacji oraz zalety chmurowych rozwiązań do zarządzania projektami informatycznymi. Stworzono autorski ranking pięciu popularnych rozwiązań chmurowych do zarządzania projektami.

Słowa kluczowe: aplikacje do zarządzania projektem; chmura; ranking aplikacji

*Autor do korespondencji.

Adres e-mail: grzenekk@wp.pl

Solutions for managing IT projects in the cloud

Grzegorz Szydlowski*
Institute of Computer Science, Lublin University of Technology, Nadbystrzycka 36B, 20-618 Lublin, Poland

Abstract. The article discusses the concept of IT project management, its implementation space and the advantages of cloud solutions for managing IT projects. A ranking of five popular cloud solutions for project management has been created.

Keywords: applications for project management; cloud; application ranking

*Corresponding author.

E-mail address: grzenekk@wp.pl

1. Wstęp

Rozwiązania do zarządzania projektami znaleźć można w większości większych organizacji i korporacji, a także w mniejszych firmach, czy też u tysięcy domowych freelancerów. Korzystanie z takiego narzędzia pozwala na lepsze kontrolowanie procesu rozwoju i realizacji projektu i niejednokrotnie stanowi klucz do osiągnięcia sukcesu. Systemy zarządzania projektami wspomagają proces realizacji oraz monitorowania prac projektowych. Współczesne rozwiązania dają menadżerom kontrolę oraz wgląd w postęp prowadzonych prac, zwiększając tym samym poziom efektywności grup projektowych oraz kontroli nad dostępnymi zasobami i pracownikami wchodzącymi w skład zespołu projektowego. Menadżer odpowiadający za realizację projektu może sprawdzić i przeanalizować poziom zaangażowania pracowników w powierzone im zadania oraz efekt ich działań. Co więcej, wspomniana funkcjonalność umożliwia przeprowadzenie optymalizacji pracy, kontroli postępu na każdym etapie realizacji projektu i zadania i efektywniejszego rozliczania klientów.

Według statystyk tylko nikły procent projektów informatycznych kończy się sukcesem, a często sukces ten jest tylko połowiczny. Duża część przedsięwzięć, jeśli nawet doprowadzone zostaną do końca pociągają przekroczenie założonego na etapie projektowym budżetu, czasu, a systemy, które doczekają się premiery często pozostawiają wiele do życzenia, jeżeli chodzi o ich funkcjonalność, działanie, stabilność i wydajność [5].

Rozwiązania systemowe realizowane w postaci chmury to nowy trend, który dostarcza więcej możliwości w działaniu. Wzrost zapotrzebowania na systemy zarządzania projektami, powoduje, że tego typu, nowe rozwiązania pojawiają się jeden po drugim na rynku oprogramowania. Tak duża ilość dostępnych rozwiązań powodować może problemy w odpowiednim doborze systemu do potrzeb danej firmy i organizacji.

W artykule przedstawiono ranking pięciu popularnych programów do zarządzania projektami w środowisku chmury.

2. Pojęcie zarządzania projektem, przestrzeń realizacji projektem informatycznym, zalety chmurowych rozwiązań do zarządzania projektem

2.1. Pojęcie projektu informatycznego

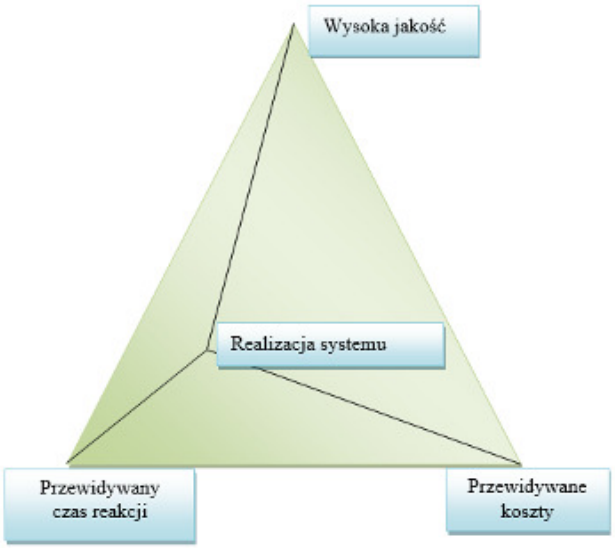
Pod pojęciem projektu rozumieć należy zestaw zadań, które podlegają zarządzaniu. Części projektu są podporządkowane osiągnięciu celu w ramach zdefiniowanych ograniczeń. W projektach, w których wykonywane jest jedno zadanie, przez jednego człowieka mówienie o procesie zarządzania nie ma sensu. Proces zarządzania związany jest z wieloma zadaniami, które są ze sobą powiązane bez względu na to przez ile osób są one wykonywane. W praktycznych rozwiązaniach planowanie rozpoczyna się od zdefiniowania listy zadań i wyboru odpowiednich wykonawców. Podkreślić należy, że zbyt duża ilość osób, które są zaangażowane w proces rozwiązania problemu może znacznie utrudnić lub uniemożliwić ukończenie zadań. Nakład pracy poniesiony na

planowanie i koordynację może, więc przerosnąć złożoność zadania [1].

Jak już wspomniano projekt informatyczny to szereg zadań i czynności w wyniku, których otrzymywany jest produkt, system komputerowy. Dodatkowe produkty powstające w trakcie wytwarzania oprogramowania to sprawozdania i raporty, dokumentacja projektowa, dokumentacja użytkownika oraz materiały szkoleniowe. Projekty zawsze ewoluują w trakcie ich rozwoju, zmianie podlegają wymagania oraz uwarunkowania zewnętrzne czy też technologie. Dlatego też proces zarządzania nieodzownie związany jest z podejmowaniem odpowiednich decyzji. W przypadku niewielkich projektów, zarządzanie jest zazwyczaj częścią pracy jednego członka zespołu, w przypadku większej ilości obowiązków funkcja zarządzania pochłania całkowicie czas jednej lub nawet kilku osób.

2.2. Przestrzeń realizacji projektu informatycznego

Zarządzanie projektami jest procesem trudnym, ponieważ proces ten realizowany jest w ramach zdefiniowanych ograniczeń, do których zaliczyć należy ograniczenia czasowe i dostępności zasobów zarówno finansowych jak i ludzkich, zaś wytwarzany produkt musi cechować się wysokim współczynnikiem wydajności.



Rys. 1. Problemy występujące podczas realizacji projektu [3]

Powstające w trakcie realizacji projektu informatycznego problemy przedstawiono na Rys 1. Prezentuje on przestrzeń, w jakiej możliwa jest rzeczywista realizacja projektu, czyli, na jakie kompromisy należy pójść w celu wyprodukowania systemu informatycznego. Wierzchołki trójkąta to optymalne wykonanie jednego z trzech kryteriów. Jeżeli proces realizacji oddala się od wierzchołka, tym trudniejsze staje się spełnienie kryterium przypisanego danemu wierzchołkowi.

Struktura zespołu to istotny element wpływający na jego poprawne funkcjonowanie. Najpopularniejszą jest struktura sieciowa, gdzie członkowie zespołu wzajemnie współpracują, kontrolują swoją pracę, wzajemnie informując się o postępach i komplikacjach, dzięki czemu w miarę szybko osiągnąć można odpowiednie standardy jakości. Sieciowa struktura pozwala na łatwe i szybkie przejęcie obowiązków jednego z członków zespołu przez innego jak i wspólną pracę nad określonymi problemami. Wadą tego rozwiązania jest ograniczenie ilości osób wchodzących w skład zespołu, gdyż zespół zawierający powyżej 10 pracowników powoduje wzrost ilości interakcji miedzy pracownikami i może wpłynąć negatywnie na wydajność zespołu [2].

Zespoły „wirtualne" wykorzystujące rozwiązania chmury to najnowszy aspekt zarządzania projektami informatycznymi. W skład takiego zespołu wchodzą pracownicy mieszkający w różnych krajach, kontynentach, którzy pracują nad tym samym projektem, komunikują się poprzez sieć Internet. Dostęp do narzędzi World Web umożliwia nie tylko realizację sprawnej komunikacji, ale także współdzielenie zasobów i realizację na odległość dużych projektów. Odpowiednie narzędzia pracy grupowej i globalna sieć Internet nie tylko pozwalają na realizację projektów przez członków zespołu rozsianych po całej sieci, ale także zwiększenie wydajności, ponieważ dzień pracy trwa 24 godziny, gdy pracownicy znajdują się w różnych strefach czasowych. Sprawna organizacja pracy takiego zespołu wymaga zastosowania wspólnej metodyki postępowania, stosowania jednorodnych procedur a członkowie zespołu winni posiadać podobny poziom wiedzy ogólnej.

Zaletą zespołów „wirtualnych" jest możliwość rekrutacji do zespołu najlepszych specjalistów, bez względu na to gdzie się znajdują a co za tym idzie zwiększa się innowacyjność i kreatywność zespołu. Tak dobrany zespół jest w stanie wytwarzać tańszy produkt niż zespoły lokalne gdyż minimalizacji poddawane są wydatki na aranżowanie spotkań, podróżowanie i jednocześnie zwiększa się możliwość zatrudnienia tańszych specjalistów. W dobrze zarządzanych zespołach „wirtualnych" liczyć można na skrócenie cyklu wytwórczego produktu dzięki dostępowi do wiedzy ekspertów i szybkiej wymianie informacji [2].

2.3. Zalety chmurowych narzędzi do zarządzania projektami

Pomimo wielu korzyści lokalnych narzędzi do zarządzania projektami, rozwiązania chmurowe oferują dużo dodatkowych korzyści takich jak [4]:

- niższe koszty - wykorzystanie lokalnego oprogramowania do zarządzania projektami wiąże się z koniecznością poniesienia wielu wydatków dotyczących zakupu serwerów, licencji na oprogramowanie, które wykorzystywane będzie na poszczególnych stanowiskach komputerowych, pracy specjalistów IT oraz długotrwałego procesu wdrożenia. W przypadku rozwiązań chmurowych konieczne jest poniesienie rocznej lub miesięcznej opłaty za dostęp do systemu,
- dostęp do programu w dowolnej chwili i z dowolnego miejsca – dostęp do oprogramowania chmurowego realizowany jest przez stronę WWW, dlatego też użytkownik posiadający dostęp do łącza internetowego, smartfona lub tabletu będzie miał możliwości skorzystania z programu.

- lepszy kontakt między członkami zespołu – rozwiązanie chmurowe pozwala pracownikom dzielących wiele kilometrów poczuć się częścią zespołu. Wszyscy użytkownicy otrzymują niezbędne informacje, dzięki czemu nie powstaje ryzyko, że ważny uczestnik projektu nie zostanie poinformowany lub pominięty.
- automatyczne i natychmiastowe aktualizacje - narzędzia chmurowe są aktualizowane w sposób automatyczny w odróżnieniu od lokalnego oprogramowania, w przypadku, którego konieczne jest pobranie aktualizacji na każdy komputer,
- decyzje podejmowane na podstawie aktualnych danych – dążenie do generowania zysków przez firmę wymaga podejmowania odpowiednich decyzji na podstawie aktualnych danych, co zapewniają rozwiązania chmury, w których to dane są aktualizowane błyskawicznie.
- komunikacja w czasie rzeczywistym – największą zaletą korzystania z oprogramowania zarządzania projektami w chmurze jest możliwość szybkiego przekazywania zwrotnych informacji zawsze, gdy tylko pojawią się nowe zawartości w obszarze roboczym projektu.
- większa produktywność – centralne przechowywanie zadań, dokumentów, kalendarzy i podsumowań w chmurowych narzędziach do zarządzania projektami pozwala oszczędzić czas a tym samym przekłada się na wzrost poziomu produktywności.

3. Cel badań

Jednym z głównych celów badania było sporządzenie autorskiego rankingu aplikacji chmurowych pozwalających na zarządzanie projektami. Wybrano pięć popularnych programów do zarządzania projektami informatycznymi pracujących w środowisku chmury. Po wykonaniu analizy ich możliwości, wyodrębniono najważniejsze cechy i funkcjonalności tych programów decydujące o ich wykorzystaniu w procesie zarządzania projektem. Programom tym przydzielono punktację za posiadanie tych określonych cech i funkcjonalności a następnie stworzono ranking.

Narzędzia, które zostały zbadane:

1) Trello [9],
2) Jira [8],
3) BlueAnt [7],
4) Bitrix24 [6],
5) Wrike[10].

Oceny dokonano pod względem :

1) interfejsu programu,
2) podstawowej funkcjonalności,
3) opcji zarządzania i obsługi projektu.

Ostatecznie przeprowadzono normalizację zgromadzonych danych w celu sprawdzenia czy zmienią się uzyskane wyniki.

4. Wyniki

Funkcjonalność oraz możliwości analizowanych programów podzielone zostały na kategorię. W każdej kategorii umieszczone zostały odpowiednie cechy programów z zakresem punktowym.

Na wstępie zostało przygotowane kryterium oceny każdej z kategorii. Autor indywidualnie wybrał najbardziej interesujące cechy i funkcjonalności, które mogą być dostępne w każdym z badanych narzędzi. Tabela 1 przedstawia sporządzone kryterium dla kategorii – interfejs programu.

Tabela 1. Interfejs programu – kryteria punktowe

BADANA CECHA	ATRYBUTY	ILOŚĆ
Responsywny charakter systemu	tak	1 pkt
	nie	0 pkt
Interfejs w języku polskim	brak	0 pkt
	częściowy	1 pkt
	całkowity	2 pkt
Pulpit systemowy	edycja układu	1 pkt
	zarządzanie gadżetami	1 pkt
Wygląd pulpitu	zmiana kolorystyki	1 pkt
	zmiana formatu daty/godziny	1 pkt
	wybór motywu obszaru roboczego	1 pkt
Aplikacja mobilna	tak	1 pkt
	nie	0 pkt
Interfejs aplikacji	desktopowa	1 pkt
	przeglądarka WWW	1 pkt

Dla kategorii podstawowa funkcjonalność wybrano uniwersalne opcje dostępne w większości tradycyjnych aplikacji webowych. Tabela 2 przedstawia ustalony system punktowy dla tej kategorii.

Tabela 2. Podstawowa funkcjonalność – kryteria punktowe

NAZWA BADANEJ CECHY	ATRYBUTY	ILOŚĆ
Menadżer kopii zapasowych	tak	1 pkt
	nie	0 pkt
Importy danych z pliku	tak	1 pkt
	nie	0 pkt
Opcje przywracania systemu	z załącznika	1 pkt
	z kopi zapasowej	1 pkt
	z opcją scalenia danych użytkownika z pliku eksportu z danymi zawartymi w chmurze	1 pkt
	zastąpienie i ponowne utworzenie danych użytkowników	1 pkt
Pomoc administracyjna	tak	1 pkt
	nie	0 pkt
Zarządzanie załącznikami	tak	1 pkt
	nie	0 pkt

Ostatecznie sporządzono kryterium dla opcji zarządzania projektem, w którym uwzględniono cechy wspomagające zarządzanie zadaniami oraz pracownikami. W tabeli 3 zestawiono system punktowy.

Tabela 3. Opcje zarządzania i obsługi projektów – kryteria punktowe

NAZWA BADANEJ CECHY	ATRYBUTY	ILOŚĆ
Zarządzanie danymi projektu	zmiana nazwy projektu	1 pkt
	wybór kategorii projektu	1 pkt
	rodzaj dostępu (pełny / ograniczony / prywatny)	1 pkt
Użytkownicy projektu	dodaj / usuń użytkownika	1 pkt
	określ/ zdefiniuj rolę użytkownika	1 pkt
	avatary użytkowników	1 pkt
Harmonogram prac projektowych	dodaj zadanie	1 pkt
	zmień status zadania	1 pkt
	zdefiniuj termin wykonania zadania	1 pkt
	dodaj opis/etykietę zadania	1 pkt
	zmień status zadania	1 pkt
	wyszukaj zadania	1 pkt
Tablica projektu	dodaj zgłoszenie	1 pkt
	dodaj flagę / etykietę	1 pkt
	przypisz do nadrzędnego zgłoszenia	1 pkt
	wyszukiwarka zadań zdefiniowanych na tablicy	1 pkt
	grupowanie zadań - filtrowanie	1 pkt
	opcja komentowania	1 pkt
	możliwość dodania załączników	1 pkt
	opcja łączenia zgłoszeń	1 pkt
	opcja dodawania niestandardowych pól do zgłoszenia	1 pkt
Powiadomienia / obserwowanie	automatyczne powiadomienia	1 pkt
	opcja obserwowania zdarzeń	1 pkt
Strony – „miejsca" – foldery pozwalające na tworzenie treści związanych z projektem	szablon pustej strony	1 pkt
	szablon strony decyzyjnej	1 pkt
	szablon strony notatka ze spotkania	1 pkt
	szablon strony wymagań produktu	1 pkt
Dostęp do opcji raportów	raporty dla wybranych zdań i scenariuszy	1 pkt
	raport	1 pkt
	całotygodniowy dla wybranego projektu	
	raport zadań zaległych	1 pkt
	raportowanie nieprzypisanych zadań	1 pkt
	raport czasu poświęconego nad pracą nad projektem	1 pkt
Lista zadań dla zalogowanego użytkownika	dodanie nowego zadania	1 pkt
	przegląd zadań na kolejny dzień, tydzień	1 pkt
	wyszukiwarka zadań	1 pkt
Chat i rozmowy w obrębie grupy roboczej	tak	1 pkt
	nie	0 pkt
Współdzielony dysk w chmurze dla użytkowników projektu	tak	1 pkt
	nie	0 pkt
Moduł CRM	zarządzanie kontaktami	1 pkt
	zarządzanie firmami	1 pkt
	zarządzanie produkt.	1 pkt

Tabela 4 przedstawia wyniki punktowe dla każdego z wybranych programów.

Tabela 4. Przydzielone punkty w zależności od kategorii i wybranych funkcji

	trello	jira	bluea nt	bitrix 24	wrike
Kategoria I - Interfejs systemu					
a) Responsywny charakter systemu	0 pkt	0 pkt	0 pkt	0 pkt	0 pkt
b) Interfejs w języku polskim	2 pkt	1 pkt	0 pkt	2 pkt	0 pkt
c) Pulpit systemowy	0 pkt	2 pkt	1 pkt	1 pkt	2 pkt
d) Wygląd pulpitu	2 pkt	3 pkt	2 pkt	3 pkt	3 pkt
e) Aplikacja mobilna	0 pkt	0 pkt	0 pkt	0 pkt	1 pkt
f) Interfejs aplikacji	1 pkt	1 pkt	1 pkt	2 pkt	1 pkt
SUMA	5 pkt	7 pkt	4 pkt	8 pkt	7 pkt
Kategoria II – podstawowa funkcjonalność					
a) Menadżer kopii zapasowych	0 pkt	1 pkt	1 pkt	1 pkt	1 pkt
b) Zewnętrzy import danych systemowych	0 pkt	1 pkt	0 pkt	0 pkt	0 pkt
c) Opcje przywracania systemu	0 pkt	4 pkt	3 pkt	2 pkt	3 pkt
d) Pomoc administracyjna	0 pkt	1 pkt	1 pkt	1 pkt	1 pkt

e) Zarządzanie załącznikami	1 pkt	1 pkt	1 pkt	1 pkt	1 pkt
SUMA	1 pkt	8 pkt	6 pkt	5 pkt	6 pkt
Kategoria III – opcje zarządzania i obsługi projektów					
a) Podstawowe dane projektu	2 pkt	3 pkt	3 pkt	3 pkt	3 pkt
b) Użytkownicy projektu	2 pkt	2 pkt	3 pkt	3 pkt	3 pkt
c) Harmonogram prac projektowych	4 pkt	6 pkt	5 pkt	5 pkt	6 pkt
d) Tablica projektu	6 pkt	9 pkt	7 pkt	8 pkt	8 pkt
e) Powiadomienia / obserwowanie	0 pkt	2 pkt	1 pkt	1 pkt	1 pkt
f) Strony – „miejsca" – foldery pozwalające na tworzenie treści związanych z projektem	0 pkt	4 pkt	2 pkt	2 pkt	3 pkt
g) Dostęp do opcji raportów	1 pkt	5 pkt	4 pkt	4 pkt	5 pkt
h) Lista zadań dla zalogowanego użytkownika	3 pkt	3 pkt	3 pkt	3 pkt	3 pkt
i) Chat i rozmowy w obrębie grupy roboczej	0 pkt	1 pkt	1 pkt	1 pkt	1 pkt
j) Współdzielony dysk w chmurze dla użytkowników projektu	0 pkt	1 pkt	0 pkt	0 pkt	1 pkt
k) Moduł CRM	0 pkt	3 pkt	2 pkt	2 pkt	3 pkt
SUMA	18 pkt	39 pkt	31 pkt	32 pkt	37 pkt
SUMA CAŁKOWITA	24 pkt	54 pkt	41 pkt	45 pkt	50 pkt

W sporządzonym autorskim rankingu wygrał program Jira. Na drugim miejscu uplasowało się narzędzie Wrike. Natomiast trzecie miejsce zajął program bitrix24. Suma punktów przyznanych programom w poszczególnych kategoriach ich analizy, poddana została wielowymiarowej analizie danych. Cechy programów charakteryzują się różnymi obszarami zmienności, dlatego też zostały zastosowane cztery metody normowania danych pozwalające na porównanie wyników i uzyskanie zmiennych transformowanych, które pozbawione są miana i zmienności. Zastosowane metody normowania to: rangowanie, przekształcenie ilorazowe, unitaryzacja oraz standaryzacja.

W ostateczności przeprowadzenie normalizacji nie przyniosło zmian w pozycjach rankingowych.

Końcowe zestawienie analizowanych programów w tworzonym rankingu wygląda następująco:

- I miejsce – **JIRA,**
- II miejsce – **WRIKE,**
- III miejsce – **BITRIX24,**
- IV miejsce – **BLUEANT,**
- V miejsce – **TRELLO.**

5. Wnioski

Na rynku istnieje duża liczba rozwiązań chmurowych do zarządzania projektem informatycznym. Osoby zainteresowane systemami zarządzania projektami często podczas ich wyboru kierują się opiniami dostępnymi w sieci Internet lub kosztem miesięcznym użytkowania wspomnianego rozwiązania. Niestety dostępne porównania aplikacji dostępne w globalnej sieci nie zawsze są obiektywne, wszystko zależy od tego, przez kogo zostały wykonane lub na jakiej stronie zamieszczone. Warto, więc wcześniej skorzystać z opcji osobistego testowania systemu. W większości przypadku jest możliwość testowania bezpłatnie aplikacji przez okres 15 dni. Użytkownik podczas takich testów sam może podjąć decyzję, które oprogramowanie cechuje się jego zdaniem największym współczynnikiem funkcjonalności i jest w stanie spełnić jego oczekiwania.

Podczas doboru cech diagnostycznych należy odrzucić te, które nie są istotne dla badanego problemu celem uzyskania jak najbardziej wiarygodnego wyniku końcowego.

Na wynik końcowy rankingu ma duży wpływ osoba dokonująca analizy oraz rodzaj wybranych cech i przedzielenia im wagi. Nie dla każdej osoby ta sama funkcjonalność aplikacji posiadać będzie taką samą rangę stąd też możliwe nieznaczne różnice w wynikach końcowych.

Podczas wyboru oprogramowania do zarzadzania projektami zdefiniować należy listę założeń funkcjonalnych i niefunkcjonalnych, jakie winien posiadać optymalny program. Na ich podstawie należy dopiero dokonać wyboru programu.

Weryfikacja fachowych publikacji, testów zamieszczonych w czasopismach IT nie pozwala tak naprawdę na wskazanie najlepszego rozwiązania chmurowego do zarządzania projektami. Producenci na własnych stronach WWW, zachwalają funkcjonalności swojego produktu, zachęcając użytkownika do jego zakupu. Osoby odpowiedzialne za proces zarządzania projektami winny same przetestować w okresie próbnym dostępne rozwiązania, podobnie jak osoby wchodzące w skład zespołu projektowego celem wyboru najlepszego rozwiązania spełniającego ich potrzeby i preferencje.

Literatura

[1] Kapusta M., Zarządzanie projektami krok po kroku, wydawnictwo Edgard, 2013

[2] Knapp J., Zeratsky J., Kowitz B., Pięciodniowy sprint. Rozwiązywanie trudnych problemów i testowanie pomysłów, wydawnictwo Helion, 2017

[3] Mingus N., Zarządzanie Projektami, wydawnictwo Helion, data wydania: 2009.

[4] Phillips J., Zarządzanie projektami IT. Wydanie III, wydawnictwo Helion, 2011

[5] Żmigrodzki M., Zarządzanie projektami dla początkujących. Jak zmienić wyzwanie w proste zadanie, wydawnictwo Onepress, data wydania: 2018.

[6] Bitrix24 Page Site, http;//bitrix24.com [2019-06-18]

[7] Website BlueAnt Application, http://proventis.net/pl/odkryj-blue-ant.html [2019-05-12]

[8] Jira SCRUM Website, http;//pl.atlassian.com/software/jira [2019-05-20]

[9] Website Trello, http;//trello.com [2019-05-18]

[10] Wrike Application Website, http;//wrike.com/pl [2019-05-12]

JOURNAL of COMPUTER SCIENCES INSTITUTE

JCSI 13 (2019) 293-297
Wysłane: 2019-06-14
Przyjęte: 2019-06-28

Analiza wydajnościowa frameworka Symfony do tworzenia nowoczesnych aplikacji webowych na podstawie wybranych wersji

Wójcik Aleksander, Wolski Mateusz*, Jakub Bartłomiej Smołka
Politechnika Lubelska, Instytut Informatyki, Nadbystrzycka 36B, 20-618 Lublin, Polska

Streszczenie. Tematyką badaną w artykule jest sprawdzenie wydajności frameworka Symfony do tworzenia aplikacji webowych. Przedstawiono przegląd literatury mówiącej o Symfony oraz jego najpopularniejsze moduły. Na podstawie stworzonych trzech identycznych aplikacji testowych napisanych we frameworkach Symfony 2.8, 3.4 oraz 4.2 porównano je ze sobą pod względem wydajnościowym. Aplikacja testowa została napisana w formacie bloga. Został wykorzystany styl architektury oprogramowania, znany jako API. Dzięki temu możliwe jest przeprowadzenie zaplanowanych testów. Badanie wydajności narzędzia Symfony dla poszczególnych wersji sporządzono na podstawie między innymi czasu ładowania danych z bazy oraz ich wyszukiwania w kolekcji danych, dodatkowymi testami było pobieranie danych z pliku csv oraz zapisywanie ich do pliku csv.

Słowa kluczowe: framework Symfony; język PHP; MVC; tworzenie aplikacji w frameworku Symfony; testy wydajnościowe

*Autor do korespondencji.

Adresy e-mail: olekwojcik1@gmail.com, maaateusz94@gmail.com

Performance analysis of the Symfony framework for creating modern web application based on selected versions

Wójcik Aleksander, Wolski Mateusz*, Jakub Bartłomiej Smołka
Institute of Computer Science, Lublin University of Technology, Nadbystrzycka 36B, 20-618 Lublin, Poland

Abstract. The subject reached in the article is to check the performance of the Symfony framework for creating web applications. An overview of the literature about Symfony and the most popular modules. Based on the created three identical test applications written in the Symfony 2.8, 3.4 and 4.2 frameworks, they were compared with each other in terms of performance. The test application was written in the blog format. The software architecture style known as the API has been used. Related to this, it is possible to conduct scheduled tests. Symfony's performance testing for individual versions was based on, the time of loading data from the database and their search in the data collection, additional tests were to download data from a csv file and save them to a csv file.

Keywords: framework Symfony; programming language PHP; MVC; creating of the web application in framework Symfony; performance tests

*Corresponding author.

E-mail addresses: olekwojcik1@gmail.com, maaateusz94@gmail.com

1. Wstęp

W dzisiejszych czasach przesyłanie informacji pomiędzy aplikacjami np. webowymi, a mobilnymi staje się coraz bardziej popularne. Większość systemów informatycznych korzysta z relacyjnych baz danych, dodatkowo znaczna ich cześć posiada bazy dane zaprojektowane w sposób obiektowy, a są to proste firmowe strony internetowe kończąc na zaawansowanych systemach obsługujących sklepy internetowe. Dzięki technologii API REST, dane oraz funkcjonalności można integrować między wieloma systemami informatycznymi.

2. Przegląd literatury

Wraz ze wzrostem zapotrzebowania na udoskonalanie aplikacji webowych coraz bardziej popularne zaczęły być frameworki [1]. Pozwalają one na zwiększenie wydajności programowania. Pierwszą główną funkcją frameworków jest wykorzystanie wbudowanych metod, które zajmują setki linii kodu oraz są bardzo czasochłonne w przypadku samodzielnego przygotowania. Drugim aspektem, w którym można znaleźć bardzo dobre zastosowanie frameworków są kwestie bezpieczeństwa, do których wykorzystywani są użytkownicy/programiści stający się testerami własnych aplikacji. W przypadku odkrycia luki w bezpieczeństwie twórcy frameworka utworzyli żywą społeczność, dającą możliwość raportowania o nieścisłościach. Kolejnym zaletą korzystania z frameworków jest ich darmowy dostęp, ponieważ frameworki pozwalają pisać programistom kod szybciej, z mniejszą ilością błędów, to ostateczny koszt wytworzenia aplikacji będzie mniejszy. Ostatnią korzyścią korzystania z frameworków jest ich rozbudowany zespół wsparcia, dokumentacja czy duże fora zwolenników danych ram programowania, dzięki temu można często znaleźć rozwiązania do różnych problemów.

3. Przegląd wybranych technologii

REST to wzór projektowy zdefiniowany w celu ułatwienia tworzenia i organizowania systemów rozproszonych. Kluczowym słowem z tej definicji powinien być styl, ponieważ ważnym aspektem REST jest to, że to styl

architektoniczny - nie jest wytyczną, nie jest standardem, oznaczałoby, że istnieje zestaw twardych reguł, które należy zastosować, aby ostatecznie uzyskać architekturę REST [2]. Główną ideą REST jest to, że system ten należy do systemów rozproszonych, zorganizowanych RESTfully. Główne cechy takiego systemu to:

- wydajność: styl komunikacji zaproponowany przez REST ma być wydajny i prosty, umożliwiając zwiększenie wydajności systemów, które go implementują,
- skalowalność: każdy system rozproszony powinien być w stanie obsłużyć ten aspekt wystarczająco dobrze, a prosta interakcja zaproponowana przez REST pozwala na to w dużym stopniu,
- prostota interfejsu: prosty interfejs umożliwia prostsze interakcje między systemami, które z kolei mogą przynosić korzyści podobne do wcześniej wspomnianych,
- modyfikowalność komponentów: rozproszony charakter systemu i oddzielenie problemów zaproponowanych przez REST umożliwia modyfikowanie komponentów niezależnie od siebie przy minimalnym koszcie i ryzyku,
- przenośność: REST może być zaimplementowana i wykorzystana przez każdy rodzaj technologii.

Framework, czyli szkielet do budowy aplikacji. Definiuje on strukturę aplikacji oraz ogólny mechanizm jej działania, a także dostarcza zestaw komponentów i bibliotek ogólnego przeznaczenia do wykonywania określonych zadań.

Główne zalety wykorzystania frameworka [7]:

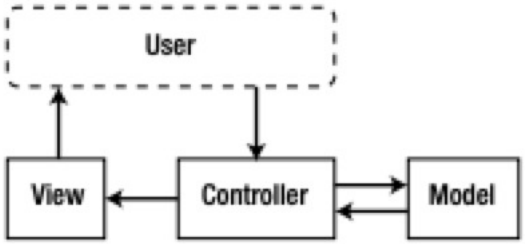
- zdefiniowana z góry struktura, która jest znana oraz przemyślana przez wiele osób,
- społeczność, która przyczynia się do ulepszenia ramowej bazy kodu
- opracowany zestaw funkcji, które są niezbędne do wykorzystania przy nowych projektach,
- moduły, biblioteki, wtyczki, które można dodać do aplikacji,
- integracja z ORM (Object-Relational Mapping),
- wstępnie ustalone wzorce projektowe w aplikacji,
- często framework narzuca dostosowanie się do pewnego wzorca projektowego, co prowadzi do lepiej zorganizowanego kodu, lepsza testowalność kodu.

MVC (Model-View-Controller) to wzorzec projektowania oprogramowania zbudowany na bazie połączenia trzech głównych typów komponentów, w języku programowania takim jak PHP, często z silnym naciskiem na paradygmaty oprogramowania obiektowego (OOP). Te trzy typy komponentów są luźno określane jako modele, widoki i kontroler [3].

Model jest miejscem, gdzie zachowywana jest cała logika biznesowa aplikacji. Logika biznesowa może być czymś specyficznym dla aplikacji przechowującej dane lub korzystającej z usług stron trzecich w celu spełnienia wymagań biznesowych. Jeśli aplikacja powinna uzyskać dostęp do informacji w bazie danych, kod odpowiedzialny za to będzie przechowywany w modelu. Jeśli jest potrzeba na przykład do pobrania danych giełdowych, kod ten będzie również przechowywany w modelu.

Widok to miejsce, gdzie przechowywane są wszystkie elementy interfejsu użytkownika naszej aplikacji. Może to obejmować kod HTML, arkusze stylów CSS lub pliki JavaScript. Wszystko, co widzi użytkownik lub elementy z którymi wchodzi w interakcje, może być przechowywane w widoku [4].

Architektura MVC składa się z następujących elementów model, widok, kontroler została przedstawiona na rys. 1. Kontroler jest komponentem łączącym modele i widoki. Izolują one logikę biznesową modelu od elementów interfejsu użytkownika w widoku oraz obsługują to w taki sposób, aby aplikacja mogła reagować na interakcję użytkownika w widoku. Kontrolery są pierwszym punktem wejścia w to trio komponentów, ponieważ żądanie jest najpierw przekazywane kontrolerowi, który następnie tworzy modele i widoki wymagane do spełnienia żądania w aplikacji [3].



Rys. 1. Architektura MVC

PHP to skryptowy język programowania - napisane w nim programy nie są kompilowane do postaci kodu maszynowego, lecz wykonywane przez interpreter PHP. PHP został stworzony do tworzenia dynamicznych stron WWW [5].

PHP jest projektem typu open-source. Każdy może pobrać za darmo jego kopię, zainstalować i używać bez żadnych ograniczeń. Do kodu źródłowego zapewniony jest pełny dostęp - jeżeli programista ma odpowiednie zdolności, może go modyfikować do woli oraz nadsyłać własne propozycje zmian do osób nadzorujących projekt. Dzięki takiej wolności PHP rozwija się bardzo dynamicznie, a w Internecie można znaleźć setki modyfikacji oraz dodatkowych modułów [6].

Symfony wspierane przez SensioLabs, zostało założone przez Fabien Potencier pod koniec 2004 roku w celu szybszego tworzenia serwisów internetowych. Wkrótce po jego utworzeniu zdecydował się na udostępnienie oprogramowania na licencji MIT. Obecnie jest jednym z najbardziej popularnych frameworków PHP [12]. Framework Symfony jest rozwiązaniem wieloplatformowym, oznacza to, że można z niego korzystać niezależnie od posiadanego systemu operacyjnego. Do głównych cech Symfony, na pewno można zaliczyć [7]:

- wzorzec projektowy MVC - opisany dokładniej w rozdziale 2.2,
- programowanie obiektowe,

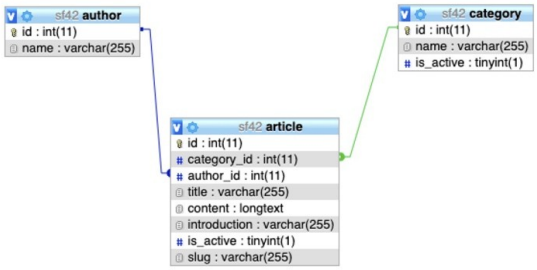
- łatwość instalacji i konfiguracji,
- zarządzanie sesjami,
- wbudowana internacjonalizacja,
- system formularzy,
- integracja z Doctrine.

4. Metoda badawcza

Praca składa się z trzech takich samych aplikacji testowych napisanych w trzech wybranych wersjach frameworka Symfony. Przygotowana aplikacja została stworzona w oparciu o popularny format bloga. Został wykorzystany styl architektury oprogramowania, znany jako API. Dzięki temu możliwe jest przeprowadzenie następujących testów:

- obciążenie wydajnościowe aplikacji wyznaczającej ciąg Fibonacciego, przy n=1000, gdzie n oznacza wygenerowane liczby ciągu Fibonacciego,
- pobranie 1000 pierwszych aktywnych artykułów z bazy danych MySQL,
- pobranie listy artykułów o ograniczeniu wyników do 50 dla każdego testu, z zastosowaniem losowych filtrów kategorii oraz autorów,
- generowanie pliku csv z danymi pobranymi z przygotowanej bazy danych,
- import danych z pliku csv do bazy danych,
- Testowa baza obejmuje następujące tabele:
- Article – informację na temat artykułów,
- Author – dane poszczególnych autorów,
- Category – informację na temat kategorii.

Na potrzeby testów została wygenerowana baza danych, zawierająca ok. 21 000 rekordów losowych danych. Schemat bazy danych na rys. 2.



Rys. 2. Schemat bazy danych

Opis środowiska testowego na którym zostały przeprowadzone testy, znajduje się w tabeli 1.

Tabela 1. Specyfikacja urządzenia fizycznego

Nazwa	Wartość
Procesor	Intel Core i5
Taktowanie	2,3 GHz
Liczba rdzeni	4
Pamięć RAM	8 GB
Pamięć flash	512 GB SDD
System operacyjny	macOS Mojave 10.14.4

Eksperyment polegał na przeprowadzeniu badań na trzech takich samych aplikacjach testowych napisanych w następujących wersjach frameworka Symfony 2.8, 3.4 i 4.2. Wersja 2.8 oraz 3.4 są wersjami LTS (Long-Term Support), czyli z najdłuższym wsparciem autora. Natomiast wersja 4.2 była w trakcie badań najnowszą stabilną wersją. Aplikacje testowe mają za zadanie zrealizować te same scenariusze . Każdy scenariusz, realizowany w powyższych frameworkach, pozwala na odmierzenie czasu w milisekundach jaki potrzebuje aplikacja na wykonanie przygotowanego badania. W zależności od testów zostały przygotowane dwie procedury testowe, pierwsza z nich dotyczyła testów eksportowania i importowania danych z pliku csv. Opis scenariusza został przedstawiony w tabeli 2.

Tabela 2. Opis pierwszej procedury testowej

Lp	Opis
1.	Uruchomienie środowiska testowego.
2.	Wczytanie niezbędnych danych do wykonania testu.
3.	Przygotowanie metody odpowiedzialnej za przeprowadzenie testu.
4.	Odczytanie aktualnego czasu systemowego i ustawienie go jako punkt start.
5.	Wywołanie przygotowanej metody.
6.	Odczytanie aktualnego czasu systemowego i ustawienie go jako punkt stop.
7.	Zapisanie wyniku testu.

Druga procedura testowa została przygotowana dla pozostałych badań. Opis tego scenariusza został przedstawiony w tabeli 3.

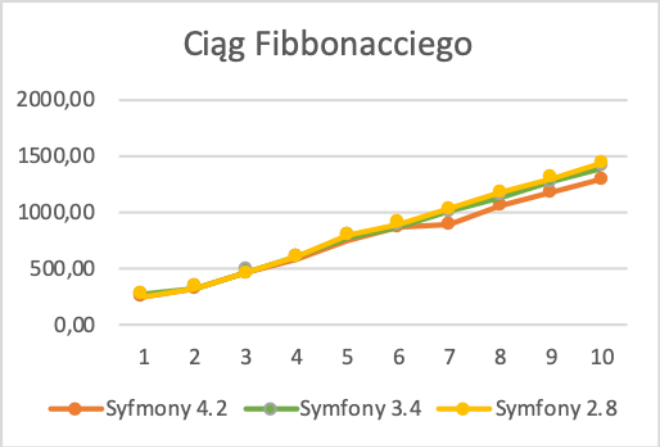
Tabela 3. Opis drugiej procedury testowej

Lp	Opis
1.	Uruchomienie środowiska testowego.
2.	Wczytanie niezbędnych danych do wykonania testu.
3.	Konfiguracja programu testowego JMeter – w zależności od przeprowadzonego testu.
4.	Wykonanie testu.
5.	Zapisanie danych.

5. Rezultaty badań

5.1. Scenariusz 1

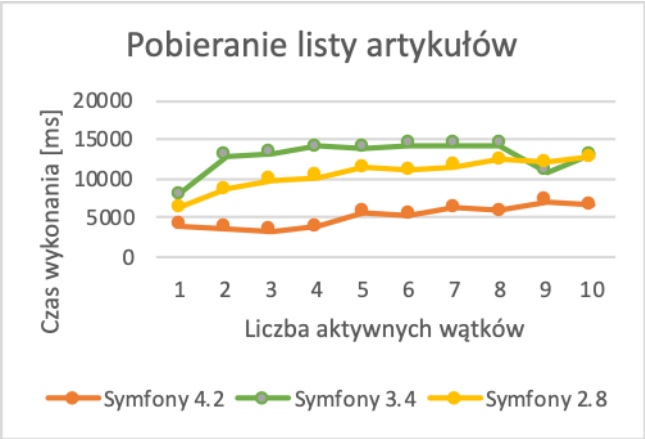
Pierwszy scenariusz polegał na obciążeniu aplikacji wyznaczającej ciąg Fibonacciego, przy n=1000, gdzie n oznacza wygenerowane liczby ciągu.



Rys. 3. Wykres czasu wykonania aplikacji obciążonej ciągiem Fibonacciego

5.2. Scenariusz 2

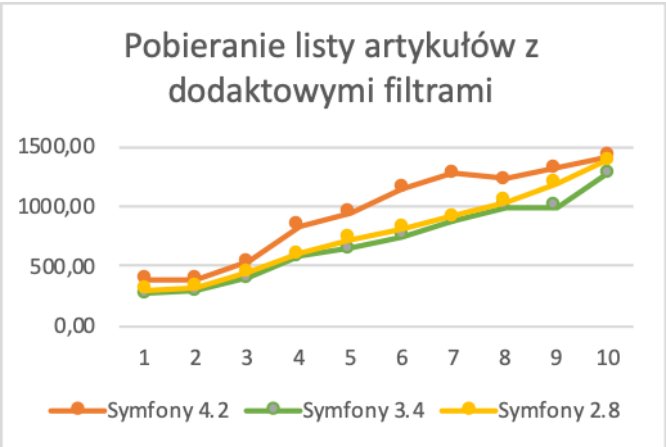
Drugi scenariusz polegał na pobraniu 1000 pierwszych aktywnych artykułów z bazy danych MySQL.



Rys. 4. Wykres wykonania żądania dla poszczególnych wersji frameworka wyświetlania listy artykułów

5.3. Scenariusz 3

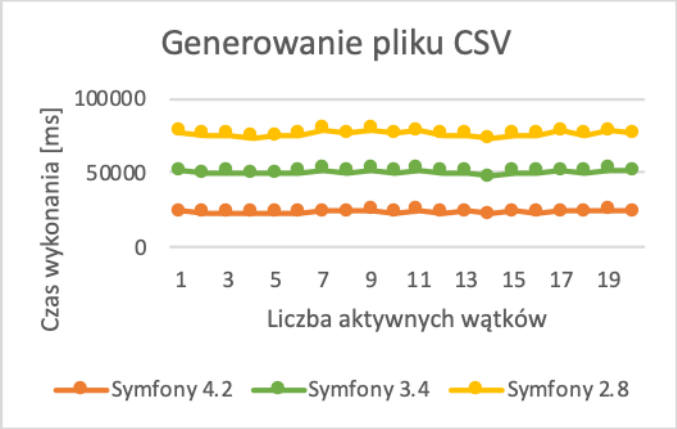
Trzeci scenariusz polegał na pobraniu listy artykułów o ograniczeniu wyników do 50 dla każdego testu, zastosowano dodatkowe filtry kategorii oraz autorów.



Rys. 5. Wykres przedstawiający pobieranie listy artykułów z dodatkowymi filtrami dla poszczególnych wersji frameworka

5.4. Scenariusz 4

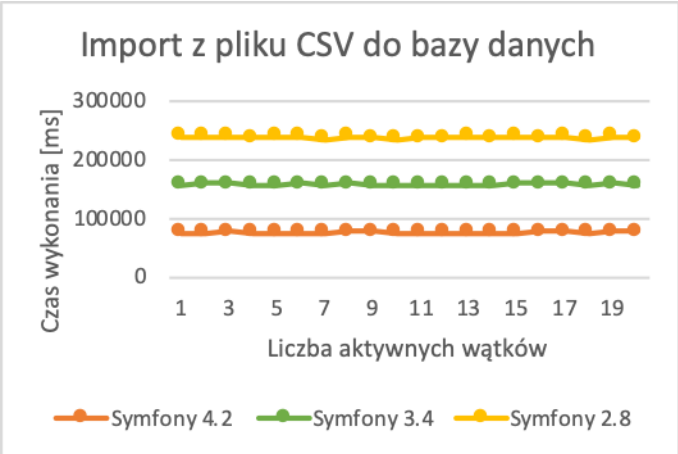
Czwarty scenariusz polegał na generowaniu pliku csv z danych pobranych z przygotowanej wcześniej bazy danych.



Rys. 6. Wykres czasów importu danych z bazy danych do pliku csv

5.5. Scenariusz 5

Piąty scenariusz polegał na odczytaniu danych z pliku csv i zapisaniu ich do bazy danych.



Rys. 7. Wykres przedstawiajacy czas importu danych z pliku csv do bazy danych

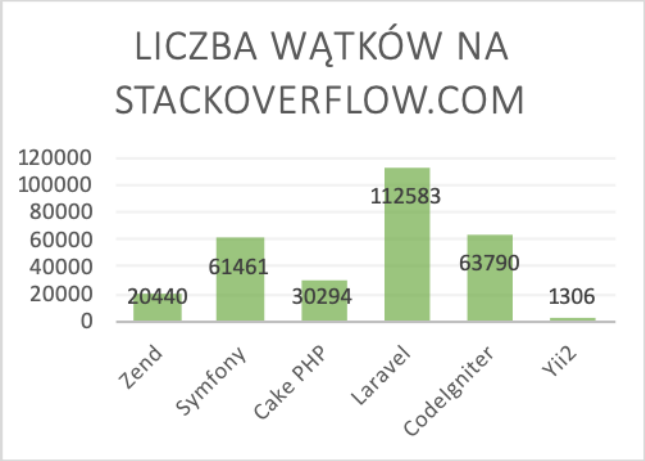
5.6. Dostępność dokumentacji

Każda wersja Symfony 2.8, 3.4 oraz 4.2 posiadają obszerne dokumentacje dostępne w języku angielskim [9]. Twórcy zadbali o to, aby w zrozumiały sposób przedstawić podstawy korzystania z nich (np. klarownie opisano wymagane narzędzia, poszczególne kroki tworzenia, instalacje oraz uruchomienia aplikacji). Poszczególne narzędzia posiadają proste , lecz dokładne materiały szkoleniowe w formie wideo. Dzięki którym można zaznajomić się z podstawowymi zasadami oraz możliwościami narzędzi. Jednakże twórcy Symfony, dodatkowo nawiązali współpracę z firmą zewnętrzną do udostępniania materiałów [10].

5.7. Wsparcie społeczności

Największą społeczność można zaleźć na platformie StackOverflow [11]. StackOverflow udostępnia statystykę z wykorzystaniem tagów – słów kluczowych. Tagi zazwyczaj związane są z technologią, której dotyczy pytanie. Serwis posiada narzędzie, które sprawdza trendy zapytań zadawanych

na StackOverflow [11]. Na Rys. 8 został przedstawiony liczba poszczególnych wątków dla wybranych technologii dotyczących najpopularniejszych frameworków PHP [11].



Rys. 8. Wykres liczby wątków na stackoverflow.com

6. Wnioski

Cel pracy, jakim było zaimplementowanie trzech jednakowych aplikacji w trzech różnych wersjach Symfony oraz przeprowadzenie wyżej opisanych testów został osiągnięty. Do badania zostały wyselekcjonowane 3 różne wersje tj. 2.8, 3.4 oraz 4.2. w każdej z tych wersji zostały napisane takie same aplikacje. Aby uwiarygodnić wyniki, wszystkie aplikacje były uruchamiane na wyizolowanym środowisku Docker.

Z otrzymanych wyników można jednoznacznie stwierdzić, że framework Symfony 4.2 wykonuję operację na dużej ilości danych znacznie szybciej od pozostałych. Jednakże w przypadku testów na próbce mniejszej ilości danych wyniki są do siebie zbliżone.

Testy zapisu i odczytu danych z i do plików csv dały jednoznaczne wyniki – Symfony 4.2 potrafi wykonać te operację trzy razy szybciej niż jego poprzednicy.

Z testów wynika, że wersją Symfony 4.2 ma znaczną przewagę przy bardziej wymagających działaniach, takich jak zwracanie dużej ilości danych z API. Natomiast, przy mniej wymagających operacjach, wszystkie badane wersję mają bardzo zbliżone wyniki.

Literatura

[1] Natalya Prokofyeva, V. B. (2017). Analysis and Practical Application of PHP Frameworks in Development of Web Information Systems. Ryga.

[2] Doglio, F. (2015). Rest 101. In: Pro REST API Development with Node.js. . Apress, Berkeley, CA .

[3] Pitt, C. (2012). *Introduction to MVC. In: Pro PHP MVC.* Apress, Berkeley, CA.

[4] Introduction to framework Symfony https://symfony.com/doc/current/create_framework/introduction.html [01.06.2019]

[5] Introducing PHP. In: Beginning PHP and MySQL. (2010). Apress.

[6] Beginning PHP an MySQL From Novice to Professional https://www.apress.com/gp/book/9781430231141 [02.06.2019]

[7] Russell, C. (2016). Frameworks. In: PHP Development Tool Essentials. Apress, Berkeley, CA.

[8] Symfony Documentation Getting Started with frameowrk https://symfony.com/doc/current/index.html#gsc.tab=0

[9] Symfony https://symfony.com/doc/ [02.06.2019]

[10] Symfony Cats Turtorial https://symfonycasts.com [31.05.2019]

[11] Stack Overflow https://stackoverflow.com [01.06.2019]

[12] Phpbenchmarks, Steevan Barboyon https://www.phpbenchmarks.com/en [06.06.2019]

JOURNAL of COMPUTER SCIENCES INSTITUTE

JCSI 13 (2019) 298-301
Wysłane: 2019-06-12
Przyjęte: 2019-06-27

Analiza porównawcza wydajności baz danych pracujących pod kontrolą systemu Windows

Serhii Stets*, Grzegorz Kozieł
Politechnika Lubelska, Instytut Informatyki, Nadbystrzycka 36B, 20-618 Lublin, Polska

Streszczenie. Celem niniejszego artykułu jest przeprowadzenie badań najpopularniejszych systemów zarządzania bazami danych takich jak MySQL, PostgreSQL oraz Firebird pod względem szybkości wykonania zapytań, obciążenia procesora oraz pamięci na dysku. Na pierwszym etapie przeprowadzono analizę popularności baz danych. W drugim etapie została napisana aplikacja umożliwiająca komunikację z wybranymi bazami danych. Na trzecim etapie zostały przeprowadzone badania za pomocą opracowanej aplikacji oraz przeanalizowano uzyskane wyniki.

Słowa kluczowe: analiza; MySQL; postgresql; firebird.

*Autor do korespondencji.

Adres e-mail: kompanybbc@gmail.com

Comparative analysis of databases working under the control of Windows system

Serhii Stets*, Grzegorz Kozieł
Institute of Computer Science, Lublin University of Technology, Nadbystrzycka 36B, 20-618 Lublin, Poland

Abstract. The aim of this paper is to conduct research on the most popular database management systems such as MySQL, PostgreSQL and Firebird in terms of query performance, CPU load and disk usage. At the first stage, the analysis of the popularity of databases was carried out. In the second stage an application was created that allows for communication with selected databases. At the third stage, tests were carried out using a written application and analysis of results.

Keywords: analysis; MySQL; postgresql; firebird.

*Corresponding author.

E-mail address: kompanybbc@gmail.com

1. Wstęp

W dzisiejszych czasach niemożliwe jest sprawne działanie bez skutecznego zarządzania danymi. Ważną kategorią są systemy przetwarzania oraz gromadzenia informacji, od których zależy wydajność każdego przedsiębiorstwa lub instytucji [1]. Taki system powinien zapewniać możliwość otrzymywania ogólnych i/lub szczegółowych raportów wyników pracy, umożliwiać łatwiejszą identyfikację trendów kluczowych wskaźników, zapewniać możliwość odbioru ważnej informacji bez znacznych opóźnień oraz wykonywać dokładną i kompletną analizę danych [2].

DBMS (ang. Database Management System) – zorganizowany zestaw oprogramowania i danych językowych, które obejmują wykresy, tabele, zapytania, raporty, widoki i inne. Zasadniczo DBMS to program, który pozwala na interakcję z bazami danych jako obiektem abstrakcyjnym (bez konieczności pisania zapytań) [3].

Obecnie znaczna część aplikacji korzysta z baz danych. Zakres stosowań baz danych jest dość szeroki: od aplikacji dla pojedynczego użytkownika, takich jak notatniki, po duże rozproszone systemy informacyjne, takie jak wyszukiwarki, sklepy internetowe, systemy bankowości elektronicznej [4].

W 1968 r. uruchomiono pierwszy przemysłowy DBMS - system IBM IMS (Information Management System). Mimo że IMS jest pierwszym ze wszystkich komercyjnych systemów zarządzania bazami danych, nadal pozostaje głównym hierarchicznym systemem zarządzania bazami danych używanym w większości komputerów głównego szeregu [7]. Pierwszy etap rozwoju DBMS wiąże się z organizacją bazy danych na dużych maszynach, takich jak IBM360/370. Bazy danych są zapisane w pamięci zewnętrznej komputera centralnego. Programy dostępu do baz danych zostały napisane w różnych językach. Interaktywny dostęp zapewniono za pomocą terminali, które nie miały własnych zasobów obliczeniowych i służyły jedynie jako urządzenia wejścia/wyjścia dla komputera centralnego.

W drugim etapie, wraz z pojawieniem się komputerów osobistych, zaczęły się rozwijać desktopowe wersje DBMS. Większość z nich miała przyjazny interfejs użytkownika, oferowała interaktywny tryb pracy z bazą danych, zarówno do tworzenia bazy danych, jak i do projektowania zapytań [5].

Trzeci etap rozwoju DBMS wiąże się z rozległym rozwojem sieci lokalnych. Praca na izolowanym komputerze z małą bazą danych staje się obecnie nietypowa dla większości aplikacji. Komputery są połączone w sieć a baza danych jest jednocześnie dostępna dla wielu użytkowników [5].

Jednym z podstawowych standardów w dziedzinie oceny DBMS są standardy opracowane przez zarząd TPC.

TPC (ang. Transaction Processing Performance Council) jest korporacją założoną w celu przeprowadzenia testów porównawczych baz danych oraz rozpowszechniania obiektywnych, weryfikowalnych danych dotyczących wydajności DBMS w branży. TPC został stworzony w celu obserwacji nie tylko poprawności testów, ale także sposobu interpretacji wyników tych testów. TPC została założona w 1988 roku przez wiodących producentów w celu obserwacji nie tylko poprawności testów, ale także sposobu interpretacji wyników tych testów, wśród których byli IBM, HP, Control Data. Każda firma może zostać członkiem TPC [8].

Testy TPC opierają się na koncepcji „transakcji", która tradycyjnie jest powiązana z relacyjnymi bazami danych, ale ma bardziej ogólne znaczenie dla zastosowań komercyjnych. Transakcja w aplikacjach biznesowych oznacza wymianę informacji o towarach, przekazywanie zleceń płatniczych czy wymianę różnego rodzaju usług. TPC definiuje i zarządza kilkoma formatami testowymi do oceny wydajności OLTP (On-Line Transaction Processing), w tym testów TPC-A, TPC-B, TPC-C, TPC-D, TPC-E i TPC-H. Jak już wspomniano, za stworzenie testu oceny odpowiedzialna jest organizacja przeprowadzająca ten test. Zespół TPC wymaga tylko spełnienia pewnych warunków. Chociaż wspomniane testy TPC nie są testami do bezpośredniej oceny wydajności bazy danych, relacyjne systemy baz danych są kluczowymi składnikami każdego systemu przetwarzania transakcji [8].

Uruchomiony w listopadzie 1989 r. Test TCP-A został zaprojektowany w celu oceny wydajności systemów działających w środowisku intensywnie aktualizowanych baz danych, typowych dla aplikacji do interaktywnego przetwarzania danych. Test TPC-A określa przepustowość systemu, mierzoną liczbą transakcji na sekundę, które system może wykonać podczas pracy z wieloma terminalami.

Pakiet testowy TPC-C pod względem rzeczywistych potrzeb użytkowników zrobił ogromny krok do przodu w odniesieniu do testów TPC-A i TPC-B. Chociaż w rzeczywistości modeluje również operacyjne przetwarzanie transakcji, jego złożoność jest przynajmniej o rząd wielkości większa niż złożoność testów A i B: wykorzystuje kilka typów transakcji, bardziej złożoną bazę danych i ogólną strukturę wykonawczą. Test TPC-C modeluje zadanie przetwarzania aplikacji. Testuje wszystkie główne elementy systemu: terminale, linie komunikacyjne, procesor, dyskowe operacje wejścia/wyjścia oraz bazę danych.

Konieczne jest również ponowne zwrócenie uwagi na fakt, że porady TPC dostarczają jedynie opisu wymagań dla testów. Sama realizacja testów jest zależna od tych, którzy bezpośrednio przeprowadzają porównanie.

W ramach niniejszej pracy zostaną wygenerowane bazy danych o różnych parametrach oraz stworzona zostanie aplikacja dla wysyłania zapytań. Za pomocą różnych narzędzi zostanie przeprowadzone badanie porównywanych baz danych oraz analiza otrzymanych wyników.

Zgodnie ze stanem na początek 2019 roku bazy danych takie jak MySQL oraz PostgreSQL znajdują się w pierwszej dziesiątce popularności DBMS [6]. Baza danych Firebird zajmuje nieco gorszą pozycję, co nie oznacza, że ten system zarządzania bazami danych nie jest warty uwagi (rys. 1).

Rank			DBMS
Jun 2019	May 2019	Jun 2018	
1.	1.	1.	Oracle
2.	2.	2.	MySQL
3.	3.	3.	Microsoft SQL Server
4.	4.	4.	PostgreSQL
5.	5.	5.	MongoDB
29.	↓ 28.	↓ 27.	Vertica
30.	30.	↓ 28.	Firebird

Rys. 1. Ranking popularności baz danych [6]

MySQL – jest to darmowy system zarządzania relacyjnymi bazami danych. MySQL jest rozwijany i wspierany przez firmę Oracle Corporation. Wcześniej MySQL był rozwijany przez szwedzką firmę „MySQL AB", która została kupiona w 2008 roku przez Sun Microsystems, a ta przez Oracle w 2010 roku. Produkt jest rozpowszechniany na licencji GNU General Public License oraz na własnej licencji komercyjnej. MySQL posiada API i wtyczki dla Delphi, C, C ++, Javy, PHP, Pythona, Ruby, Smalltalk, Component Pascal i Tcl, .NET, a także posiada obsługę ODBC za pomocą sterownika MyODBC [11].

PostgreSQL – jest jednym z najpopularniejszych otwartych systemów zarządzania bazami danych. Rozwój Postgresa, który rozpoczął się w 1986 roku, był bezpośrednio związany z Michaelem Stonebrakerem, szefem wcześniejszego projektu Ingres, w tym czasie już zakupionym przez Computer Associates. Nazwa została odszyfrowana jako „Post Ingres". Stonebraker i jego uczniowie opracowywali nowy DBMS przez osiem lat (od 1986 do 1994 roku). W tym okresie zostały wprowadzone procedury, reguły, typy zdefiniowane przez użytkownika oraz inne komponenty. W 1995 r. rozwój został ponownie podzielony: Stonebriker wykorzystując swoje doświadczenie stworzył komercyjny DBMS „Illustra", a jego uczniowie opracowali nową wersję Postgres -Postgres95, w której zastąpiono język Ingres na SQL. Następnie rozwój Postgres95 został przeprowadzony poza uniwersyteckim zespołem entuzjastów. Nowy DBMS otrzymał nazwę PostgreSQL, pod którą jest obecnie rozwijany. [10]

Firebird – jest systemem zarządzania relacyjnymi bazami danych. Działa w środowiskach systemów operacyjnych: Linux, Windows, Mac OS X i wielu innych [9].

2. Aplikacja testowa

W celu przeprowadzenia zaplanowanych badań systemów zarządzania bazami danych została napisana prosta aplikacja w języku Java w środowisku Intelij IDEA przeznaczona do

pracy na platformie Windows, pozwalająca na przesyłanie zapytań do baz danych oraz mierzenie czasu odpowiedzi na zapytanie.

3. Metodyka badań oraz stanowisko badawcze

Badania przeprowadzono na systemie Windows 10, za każdym razem uruchamiano jedynie system oraz silnik bazy danych. Każde badanie zostało powtórzone 10 razy, a jako ostateczny wynik przyjęto średnią wartość z pomiarów.

Parametry komputerów, na których zostały przeprowadzone badania.

Komputer 1:

- system operacyjny: Windows 10 Pro 64-bit,
- processor: Intel(R) Core(TM) i5-3230M,
- pamięć RAM: Kingston 8 GB,
- dysk SSD: Kingston 128 GB.

Komputer 2:

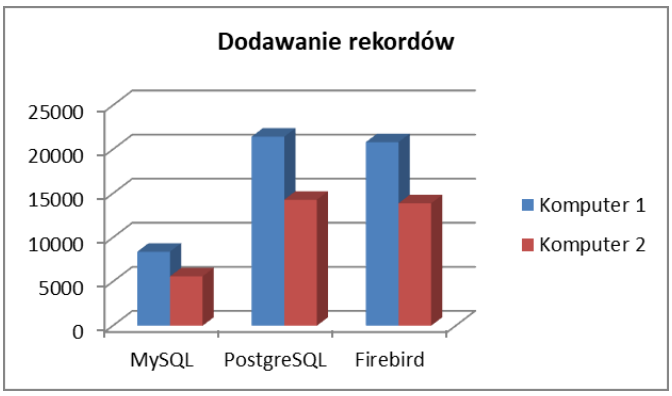
- system operacyjny: Windows 10 Pro 64-bit,
- processor : Intel(R) Core(TM) i5-8305G,
- pamięć RAM: Kingston 12 GB,
- dysk SSD: Kingston HyperX 512 GB.

4. Analiza porównawcza

Pierwszym etapem było porównanie czasów odpowiedzi baz danych podczas dodawania rekordów (rys. 2).

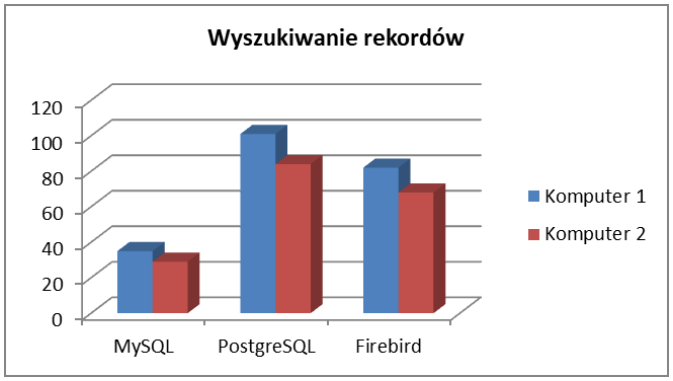
Tabela 1. Czas odpowiedzi baz danych

	Czas odpowiedzi [ms]			
	MySQL	PostgreSQL	Firebird	
Komputer 1	8400	21445	20825	Dodawanie rekordów
Komputer 2	5600	14295	13883	
Komputer 1	35	101	82	Wyszukiwanie rekordów
Komputer 2	29	84	68	
Komputer 1	20	84	47	Usuwanie rekordów
Komputer 2	30	41	39	

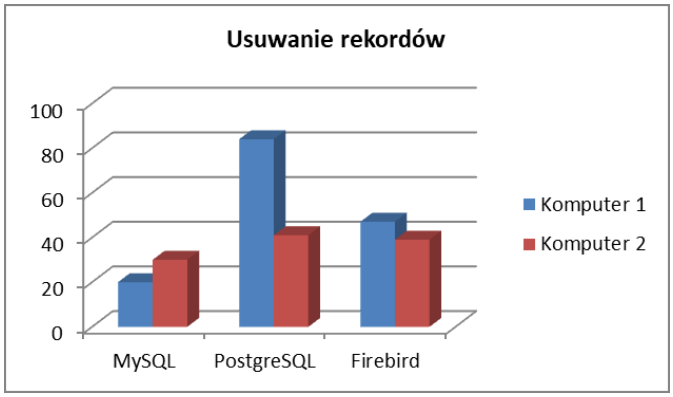


Rys. 2. Czas dodawania rekordów

Następnie analogicznie został zmierzony czas wyszukiwania rekordów oraz ich usuwania z baz danych (rys. 3, 4).



Rys. 3. Czas wyszukiwania rekordów

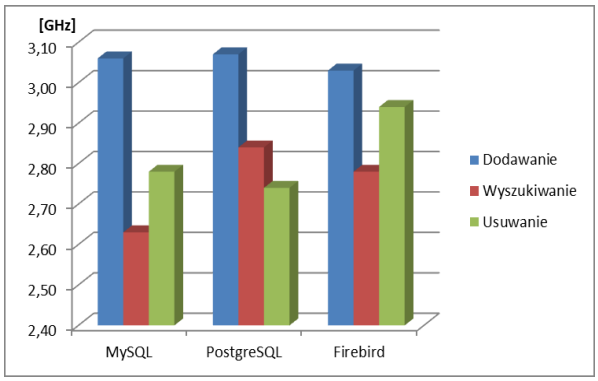


Rys. 4. Czas usuwania rekordów

Kolejnym zadaniem była analiza obciążenia procesora podczas wykonania powyższych zapytań. Wyniki są przedstawione na rysunku 5.

Tabela 2. Obciążenie procesora

	Obciążenie procesora [GHz]		
	MySQL	PostgreSQL	Firebird
Dodawanie rekordów	3,06	3,07	3,03
Wyszukiwanie rekordów	2,63	2,84	2,78
Usuwanie rekordów	2,78	2,74	2,94



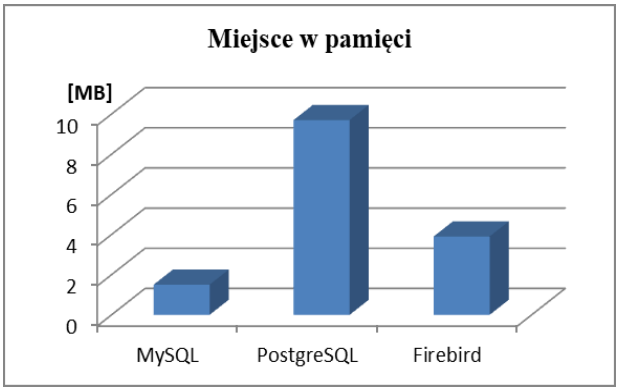
Rys. 5. Obciążenie procesora

W ostatnim etapie zostały porównane rozmiary baz danych pod względem miejsca zajmowanego na dysku

twardym. Wszystkie porównywane bazy zawierały ten sam zbiór danych.

Tabela 3. Ilość zajmowanego miejsca na dysku

Miejsce w pamięci [MB]		
MySQL	PostgreSQL	Firebird
1,5	9,7	3,9



Rys. 6. Rozmiar tabel baz danych

5. Wnioski

Podczas wszystkich przeprowadzonych badań baz danych pracujących pod kontrolą systemu Windows, pod względem szybkości wykonania zapytań pierwsze miejsce zajęła baza MySQL. Czas odpowiedzi na zapytania wysyłane do bazy MySQL był dużo krótszy od czasów uzyskanych podczas stosowania innych silników baz danych. Identycznie wygląda sytuacja pod względem ilości zajmowanego miejsca w pamięci – najlepsze wyniki uzyskała baza MySQL.

Wyniki badań obciążenia procesora wykazały zależność od rodzaju wykonywanej operacji. Podczas dodawania dużej liczby rekordów najmniejsze obciążenie procesora zaobserwowano dla silnika bazy Firebird. Podczas wyszukiwania rekordów oraz usuwania zajął on ostatnie miejsce wpośród badanych baz danych.

Na podstawie przeprowadzonych badań można stwierdzić, że system zarządzania bazami danych MySQL pod konrolą systemu Windows jest najszybszy pod względem wykonania zapytań a utworzona w nim baza danych zajmuje mniej miejsca na dysku w porównaniu do analizowanych konkurencyjnych rozwiązań, ale odbywa się to kosztem większego obciążenia procesora. Ale ponieważ powyższe testy są przeprowadzone tylko na systemie Windows nie można stwierdzić, że zachowanie testowanych baz danych na innych systemach operacyjnych będzie podobne.

Literatura

[1] Avi Silberschatz, Henry F. Korth, S. Sudarshan: Database System Concepts, McGraw-Hill Education, 2010.

[2] Baron Schwartz, Peter Zaitsev, Vadim Tkachenko: High Performance MySQL: Optimization, Backups, and Replication, O'Reilly Media, 2008.

[3] Rob Mattison: Understanding Database Management Systems, McGraw-Hill, 1997.

[4] Date, C.J: Introduction to Database Systems, Pearson, 2000.

[5] Toby J. Teorey: Database Modeling & Design: The Fundamental Principles, Morgan Kaufmann Pub, 1994.

[6] DB-Engines Ranking, https://db-engines.com/en/ranking [31.03.2019 r.]

[7] IBM Information Management System, https://en.wikipedia.org/wiki/IBM_Information_Management_System [23.06.2019]

[8] TPC-Homepage V5, http://www.tpc.org [25.02.2019 r.]

[9] Firebird, https://pl.wikipedia.org/wiki/Firebird [12.03.2019 r.]

[10] PostgreSQL, https://pl.wikipedia.org/wiki/PostgreSQL [12.03.2019 r.]

[11] MySQL, https://pl.wikipedia.org/wiki/MySQL [15.03.2019 r.]

JOURNAL of COMPUTER SCIENCES INSTITUTE

JCSI 13 (2019) 302-309
Wysłane: 2019-07-03
Przyjęte: 2019-07-10

Zastosowanie uczenia maszynowego w budowie interfejsu sterowanego głosem na przykładzie odtwarzacza muzyki

Jakub Basiakowski*

Politechnika Lubelska, Instytut Informatyki, Nadbystrzycka 36B, 20-618 Lublin, Polska

Streszczenie. Poniższy artykuł przedstawia wyniki badań wpływu zastosowania uczenia maszynowego w budowie interfejsu sterowanego głosem. Do analizy wykorzystane zostały dwa różne modele: jednokierunkowa sieć neuronowa zawierająca jedną warstwę ukrytą oraz bardziej skomplikowana konwolucyjna sieć neuronowa. Dodatkowo wykonane zostało porównanie modeli użytych w celu realizacji badań pod względem jakości oraz przebiegu treningu.

Słowa kluczowe: uczenie maszynowe; sieć neuronowa; rozpoznawanie głosu

*Autor do korespondencji.

Adres e-mail: jakub.basiakowski@pollub.edu.pl

Applying of machine learning in the construction of a voice-controlled interface on the example of a music player

Jakub Basiakowski*

Institute of Computer Science, Lublin University of Technology, Nadbystrzycka 36B, 20-618 Lublin, Poland

Abstract. The following paper presents the results of research on the impact of machine learning in the construction of a voice-controlled interface. Two different models were used for the analysys: a feedforward neural network containing one hidden layer and a more complicated convolutional neural network. What is more, a comparison of the applied models was presented. This comparison was performed in terms of quality and the course of training.

Keywords: machine learning; neural network; speech recognition

*Corresponding author.

E-mail address: jakub.basiakowski@pollub.edu.pl

1. Wprowadzenie

W dzisiejszych czasach uczenie maszynowe (ang. *machine learning*) znajduje szerokie zastosowanie w tworzonych systemach informatycznych. Różnego rodzaju aplikacje wykorzystują możliwość ciągłego rozwijania się przy użyciu dostarczanych informacji. Dzięki dużemu zainteresowaniu tą dziedziną, powstało wiele algorytmów uczenia maszynowego, które wykorzystywane są w rozwiązywaniu różnorodnych problemów. Jednym z zadań w jakim zastosowanie znajduje uczenie maszynowe jest rozpoznawanie głosu.

Celem pracy jest analiza wpływu zastosowania uczenia maszynowego w budowie interfejsu sterowanego głosem. W tym celu utworzona została aplikacja odtwarzacza muzyki, w której wybrane funkcjonalności możliwe są do wykonania poprzez wprowadzenie odpowiedniego polecenia głosowego. Proces uczenia maszynowego przeprowadzony został z wykorzystaniem dwóch różnych modeli, a następnie opracowana została analiza uzyskanych wyników oraz wykonane zostało porównanie zastosowanych w celu realizacji badań modeli.

W niniejszej pracy postawiona została teza, iż wykorzystanie sieci konwolucyjnej z odpowiednio dobranymi parametrami daje lepsze wyniki klasyfikacji niż prosta sieć jednowarstwowa.

2. Materiały i metody

2.1. Przegląd literatury

Uczenie maszynowe cieszy się dużą popularnością wśród twórców różnego rodzaju systemów. Możliwość ciągłego rozwijania aplikacji na podstawie dostarczanych do niej informacji znajduje zastosowanie w rozwiązywaniu wielu problemów, takich jak na przykład wykrywanie obiektów, rozpoznawanie twarzy czy jak w przypadku tej pracy rozpoznawanie mowy. W związku z dużym zainteresowaniem występującym wokół dziedziny uczenia maszynowego powstało wiele algorytmów, dzięki którym może zostać ono wykorzystane.

W pracy [1] uczenie maszynowe zostało wykorzystane w rozpoznawaniu mowy, a konkretnie do klasyfikacji fonemów. Autorzy przedstawiają rozwiązanie problemu z użyciem histogramów zrekonstruowanej przestrzeni fazowej. W celu realizacji badań wykorzystany został statystyczny algorytm uczenia maszynowego – naiwny klasyfikator Bayesa, który wykorzystuje prawdopodobieństwo do przyporządkowania danych wejściowych. Autorzy pracy przeprowadzili również analizę wpływu normalizowania

danych wejściowych, z której wynika iż dla danych poddanych normalizacji poprawność klasyfikacji była wyższa. Naiwny klasyfikator Bayesa wykorzystany został również w pracy [2]. W tym przypadku zastosowany został jako klasyfikator dwuklasowy w celu weryfikacji czy słowa w rozpoznanym zdaniu zostały wykryte poprawnie.

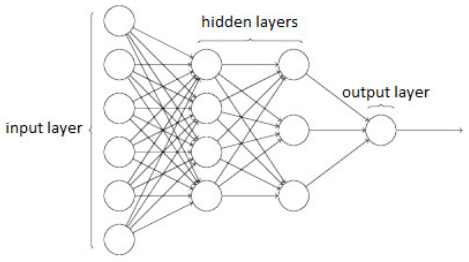
W pracach [3], [4], [5] oraz [6] w rozpoznawaniu mowy wykorzystany został algorytm maszyny wektorów nośnych. Autorzy artykułu [3] wykazali, że mechanizm uczenia maszynowego jakim jest SVM może być z powodzeniem zastosowany w klasyfikowaniu sekwencji mowy o zmiennej długości. Praca [6] przedstawia z kolei pomyślne wykorzystanie tego algorytmu w systemie ciągłego rozpoznawania głosu.

Algorytmy wykorzystane w niniejszej pracy, a mianowicie proste sieci neuronowe oraz sieci konwolucyjne również znajdują zastosowanie w rozpoznawaniu dźwięków mowy. Autorzy publikacji [7], [8] oraz [9] przedstawiają szereg zalet wykorzystania ich w rozwiązywaniu problemów klasyfikacji mowy i opisują zasadę ich działania. Praca [10] przedstawia skuteczne wykorzystanie w pełni połączonej seici jednokierunkowej w celu klasyfikacji wypowiadanych cyfr.

W pracy [11] autorzy zastosowali konwolucyjną sieć neuronową dla rozwiązania problemu rozpoznawania mowy. Klasyfikacji poddano niewielki zbiór słów kluczowych. Przedstawione zostały zalety wykorzystania modelu sieci konwolucyjnej w realizacji tego zadania. Ze względu na podobieństwo problemu zaprezentowanego w publikacji [11] z zadaniem postawionym w niniejszej pracy, zdecydowano na wykorzystanie tego ogólnodostępnego modelu dla klasyfikowania poleceń głosowych w utworzonej w celu realizacji badań aplikacji odtwarzacza muzyki. Na wykorzystanie sieci konwolucyjnej zdecydowali się również autorzy prac [12] i [13]. Publikacja [12] przedstawia wykorzystanie konwolucyjnej sieci neuronowej do klasyfikacji odgłosów występujących w środowisku, takich jak na przykład szczekanie psa czy klakson samochodowy.

2.2. Jednokierunkowa sieć neuronowa

Powstanie sztucznych sieci neuronowych (ang. *artificial neural network*) zainspirowane zostało możliwościami biologicznych neuronów oraz sieci neuronowych występujących w mózgu. Przetwarzają one informacje w sposób imitujący w pewnym stopniu działanie swoich naturalnych odpowiedników [14].



Rys. 1. Schemat sieci neuronowej typu perceptron wielowarstwowy [15]

Jednym z rodzajów sztucznych sieci neuronowych jest sieć jednokierunkowa. W tym podrozdziale przedstawiona zostanie sieć typu perceptron wielowarstwowy (ang. *multilayer perceptron, MLP*). Schemat takiej sieci przedstawiony został na rysunku 1. Składa się ona zazwyczaj z warstwy wejściowej, warstw ukrytych oraz warstwy wyjściowej.

- Warstwa wejściowa (ang. *input layer)* – zawiera dane dostarczone do sieci, a liczba znajdujących się w niej neuronów odpowiada ilości tych danych.
- Warstwy ukryte (ang. *hidden layers*) – zawierają neurony przetwarzające dane. W sieci neuronowej może występować jedna lub więcej warstw ukrytych.
- Warstwa wyjściowa (ang. *output layer*) – zawiera przewidywane odpowiedzi. W przypadku klasyfikacji liczba neuronów w tej warstwie jest równa liczbie klas.

Neurony, z których zbudowana jest sieć mogą posiadać wiele wejść i jedno wyjście. Każde połączenie występujące pomiędzy neuronami posiada przyporządkowaną wagę, przez którą pomnożona zostaje informacja wyjściowa jednego neuronu przekazywana do wejścia neuronu kolejnej warstwy. W docelowym neuronie, sumowane są wszystkie wejściowe dane pomnożone przez odpowiadające im wagi, a następnie do tej sumy dodawana jest pewna wartość ustalająca próg, przy którym neuron powinien się uaktywnić. Wartość ta to tak zwany bias. Otrzymany wynik jest podstawiany do funkcji aktywacji, gdzie zostaje przekształcony we właściwą wartość wyjściową. Równanie 1 przedstawia opisany wyżej schemat działania:

$$y=f\left(\sum_{i=1}^{N} w_i x_i + b\right) \tag{1}$$

gdzie:

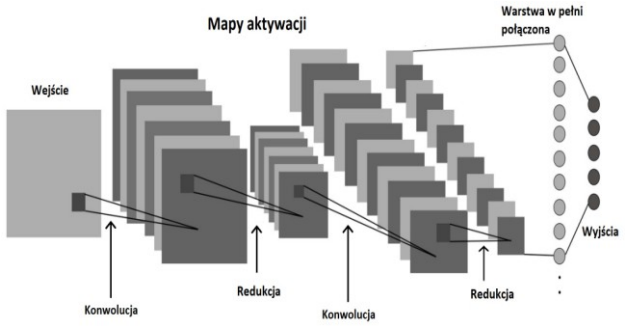
f – funkcja aktywacji,
x – informacja wejściowa,
w – waga informacji,
b – bias.

2.3. Konwolucyjna sieć neuronowa

Zamiast ogólnego mnożenia wektorów wykorzystywanego w zwykłych sieciach neuronowych, sieci konwolucyjne wykorzystują operację splotu (konwolucji) z wykorzystaniem zestawów pewnych filtrów. Operacja konwolucji najczęściej występuje w postaci korelacji krzyżowej [16]. Dzięki jej wykorzystaniu możliwe jest powstanie tak zwanych map aktywacji.

Splotowe sieci neuronowe wykorzystując zdefiniowane filtry, wykrywają pewne wzorce występujące na niewielkiej części danych wejściowych, a następnie są w stanie zauważyć, że te wykryte cechy powtarzają się na przestrzeni całej informacji dostarczonej na wejście. W rezultacie splotu danych wejściowych oraz tychże filtrów powstają wcześniej wspomniane mapy aktywacji, które są w kolejnym kroku zmniejszane (redukowane), a następnie po raz kolejny

poddawane operacji konwolucji. Architektura sieci konwolucyjnych została przedstawiona na rysunku 2.



Rys. 2. Architektura konwolucyjnej sieci neuronowej [17]

Typowa architektura sieci konwolucyjnych pozwala na wyróżnienie trzech głównych warstw:

- Warstwa konwolucyjna (ang. *convolutional layer*) – odpowiada za ekstrakcję cech. W tej warstwie wykonywana jest operacja splotu, w wyniku jej działania powstają mapy aktywacji.
- Warstwa łącząca (ang. *pooling layer*) – redukuje ilość informacji (wymiar map aktywacji). Najczęściej odbywa się to przez obliczenie średniej, lub wybór największej wartości spośród analizowanej niewielkiej części mapy aktywacji.
- Warstwa w pełni połączona (ang. *fully connected layer*) – jest to tradycyjna warstwa występująca również w innych, niekonwolucyjnych typach sieci neuronowych. Dane w niej przetwarzane są w postaci wektorów, dlatego też dostarczane do niej macierze map cech powinny zostać „spłaszczone" do postaci pojedynczego wektora, który zostaje podany na wejście tej warstwy.

3. Aplikacja testowa

Na potrzeby realizacji badania wpływu zastosowania uczenia maszynowego w budowie interfejsu sterowanego głosem utworzona została aplikacja odtwarzacza muzyki. Docelową platformą są urządzenia mobilne z systemem Android. Do prawidłowego działania aplikacji niezbędne są dwa zezwolenia: na nagrywanie dźwięku przez mikrofon urządzenia – w celu wprowadzania poleceń głosowych oraz na odczyt danych z pamięci urządzenia – w tym przypadku konieczny jest dostęp do plików utworów muzycznych, które będą wczytywane, a następnie odtwarzane.

Funkcje aplikacji nie różnią się od możliwości standardowych odtwarzaczy muzyki, jednak niektóre z nich możliwe są do wykonania za pomocą odpowiedniej komendy głosowej. W związku z tym zdefiniowane zostały wymagania funkcjonalne stworzonej aplikacji:

- stworzenie listy odtwarzania,
- usunięcie listy odtwarzania,
- wybór listy odtwarzania za pomocą wypowiedzenia jej nazwy,
- dodawanie utworu do listy odtwarzania,
- usunięcie utworu z listy odtwarzania,
- wyświetlenie listy wszystkich utworów,
- zatrzymanie odtwarzania,
- wznowienie odtwarzania,
- przyspieszenie odtwarzanego utworu o 10 sekund,
- cofnięcie odtwarzanego utworu o 10 sekund,
- przejście do następnego utworu,
- przejście do poprzedniego utworu,
- wyświetlenie listy rozpoznawanych poleceń głosowych,
- wyłączenie aplikacji za pomocą odpowiedniej komendy.

Funkcje aplikacji, które możliwe są do wykonania w sposób nie tylko standardowy (dotknięcie odpowiedniego przycisku na ekranie), ale również za pomocą głosu przedstawione zostały w tabeli 1 wraz z wywołującymi je poleceniami. To właśnie te komendy będą wykorzystane w analizie zastosowania uczenia maszynowego.

Tabela 1. Funkcje aplikacji wywoływane poleceniami głosowymi

Funkcja aplikacji	Polecenie wywołujące funkcję
Zatrzymanie odtwarzania	stop
Wznowienie odtwarzania	go
Przyspieszenie odtwarzanego utworu o 10 sekund	right
Cofnięcie odtwarzanego utworu o 10 sekund	left
Przejście do następnego utworu	forward
Przejście do poprzedniego utworu	backward
Wyłączenie aplikacji	off

4. Plan badań

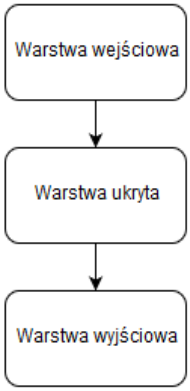
4.1. Wykorzystane modele

Zastosowane w utworzonej aplikacji testowej modele wybrane zostały z ogólnodostępnej biblioteki języka Python – *Tensorflow*. Ta otwartoźródłowa platforma umożliwia nie tylko tworzenie własnych modeli, ale również oferuje szeroki zakres gotowych rozwiązań w wykorzystaniu uczenia maszynowego dla różnych problemów, takich jak na przykład klasyfikowanie obrazów, wykrywanie obiektów czy jak w przypadku tej pracy rozpoznawania głosu [18].

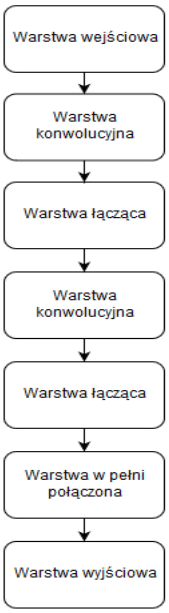
W celu przeprowadzenia badań wykorzystane zostały dwa algorytmy uczenia maszynowego oparte na sztucznych sieciach neuronowych. Pierwszym z nich jest prosta, jednokierunkowa oraz w pełni połączona sieć neuronowa, której poglądowy schemat przedstawiony został na rysunku 3.

Wykorzystana sztuczna sieć neuronowa posiada jedną warstwę ukrytą, w której występuje przetwarzanie danych wejściowych. Do neuronów tej warstwy dodawana jest zmienna wyznaczająca próg aktywacji (bias). Funkcja aktywacji, jaka została zastosowana w tym modelu to funkcja ReLU.

Drugi algorytm, który został wykorzystany w ramach badań to bardziej zaawansowany model – konwolucyjna sieć neuronowa. Poglądowy schemat tejże sieci przedstawia rysunek 4.



Rys. 3. Uproszczony schemat wykorzystanego modelu sieci jednokierunkowej



Rys. 4. Uproszczony schemat wykorzystanego modelu sieci konwolucyjnej

Model sieci konwolucyjnej, który został zastosowany w pracy posiada dwie warstwy konwolucyjne, dwie warstwy łączące, oraz warstwę, która występuje w prostych sieciach neuronowych – w pełni połączoną. W tym przypadku również zastosowana została zmienna biasu, która dodawana jest w każdej warstwie konwolucyjnej sieci oraz w warstwie w pełni połączonej. Funkcją aktywacji neuronów, tak jak w poprzednim modelu jest funkcja ReLU. W przypadku sieci konwolucyjnej należy również wspomnieć o metodzie redukowania informacji (wymiaru map aktywacji utworzonych przez operację splotu) wykonywanej przez warstwy łączące. W przypadku wykorzystanego modelu jest to wybór maksymalnej wartości spośród analizowanego obszaru danych.

4.2. Proces uczenia maszynowego

W celu przeprowadzenia procesu uczenia maszynowego wybranych modeli niezbędne było pozyskanie odpowiednich danych treningowych. W związku z tym wykorzystany został ogólnodostępny zbiór jednosekundowych [19] plików dźwiękowych, z których każdy zawiera jedno słowo (z trzydziestu pięciu występujących w zbiorze) wypowiedziane przez ludzi w różnym wieku oraz różnej płci. Dane te zebrane zostały przez Google aby pomagać w trenowaniu i doskonaleniu systemów wykrywających słowa kluczowe. Jest to ponad dwu-gigabajtowe archiwum danych zawierających ponad 105 tysięcy plików [19].

W niniejszej pracy w procesie uczenia maszynowego wybranych modeli wykorzystane zostały otwartoźródłowe skrypty napisane w języku Python udostępnione przez zespół Tensorflow na platformie GitHub [20]. Trening sieci neuronowych wykonany został przy użyciu pliku *train.py*, którego uruchomienie rozpoczyna cały proces.

Zastosowane sieci przyjmują dane wejściowe w postaci dwuwymiarowej, dlatego też sygnał dźwiękowy przekształcany zostaje do postaci obrazu, a w tym konkretnie przypadku spektrogramu. Odbywa się to poprzez podział całego sygnału na krótkie, trwające kilka milisekund próbki, dla których obliczone zostają amplitudy składowych harmonicznych. Oś pionowa spektrogramu odpowiada częstotliwości, natomiast oś pozioma określa czas. To właśnie spektrogram jest poddawany analizie oraz przetwarzany zostaje w kolejnych warstwach sieci. Przygotowanie danych wejściowych, a mianowicie przekształcenie sygnałów dźwiękowych z opisanego wcześniej zbioru utworzonego przez zespół Google [19] do postaci dwuwymiarowej zrealizowane zostało przez skrypt train.py.

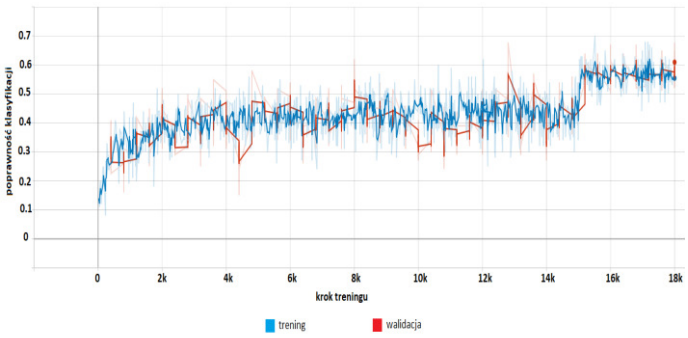
Plik train.py wymagał zastosowania kilku modyfikacji. Niezbędne było określenie modelu, który miał zostać poddany uczeniu maszynowemu. Zmienione zostały również etykiety, które reprezentują rozpoznawane przez model słowa, na te, które zostały wykorzystane w aplikacji testowej. To do nich będą klasyfikowane wprowadzane sygnały dźwiękowe. Kolejnym etapem było określenie liczby kroków oraz szybkości uczenia (ang. *learning rate*), jednak w tym przypadku wybrane zostały wartości domyślne – wykonanych zostało 18 000 kroków dla każdego modelu, z szybkością uczenia wynoszącą 0.001 dla pierwszych 15 000 kroków oraz 0.0001 dla kolejnych 3 000.

Na potrzeby pracy, dla każdego z zastosowanych algorytmów uczenia maszynowego, zapisane zostały po trzy modele, które będą badane – po 6 tysiącach kroków, po 12 tysiącach kroków oraz po zakończeniu całego treningu (po 18 tysiącach kroków). Dodatkowo analizie poddane zostaną same przebiegi uczenia maszynowego tych modeli.

5. Rezultaty badań

5.1. Sieć neuronowa jednokierunkowa

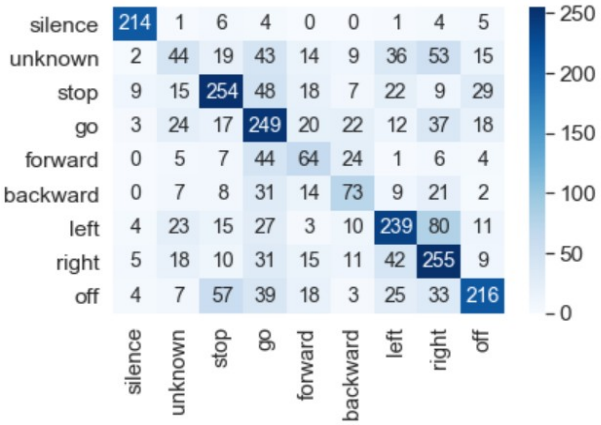
Pierwszym analizowanym modelem jest prosta sieć neuronowa. Przebieg jej uczenia przedstawiony został na rysunku 5. Czas trwania treningu opisywanej sieci nie był długi ze względu na niski stopień jej skomplikowania, trwał on blisko 6 godzin.



Rys.5. Przebieg treningu sieci jednokierunowej

W początkowym etapie treningu (pierwsze 2 tysiące kroków) poprawność klasyfikacji słów wzrastała wraz z kolejnymi krokami. W dalszej części (do 15 tysięcy kroków) zauważyć można, iż sieć przyporządkowuje słowa ze zgodnością wahającą się w granicach od 35% do 50%, bez widocznego progresu. Skok procentowej poprawności klasyfikowania słów wystąpił dopiero w momencie zmniejszenia szybkości uczenia sieci (zmiana wartości długości kroku uczenia z 0.001 do 0.0001) na ostatnie 3 tysiące kroków, jednak wzrost ten nastąpił gwałtownie po czym podobnie jak we wcześniejszej fazie treningu sieć przestała osiągać postępy w poprawnym klasyfikowaniu. Ostatecznie uczony model zakończył trening z prawidłowością rozpoznawania słów wynoszącą około 60%.

W celu dokładniejszej analizy uczenia maszynowego wykorzystanych modeli, po każdych 6 tysiącach kroków wygenerowana została macierz pomyłek (ang. confusion matrix), która jest podstawową metodą oceny poprawności klasyfikacji. Każdy wiersz macierzy reprezentuje instancję przewidywanej klasy, natomiast każda kolumna odpowiada instancji klasy rzeczywistej. Warto również zaznaczyć, iż do testów każdego modelu wykorzystana została walidacja krzyżowa (ang. *cross-validation*), której współczynnik N oznaczający liczebność zbioru testowego w przypadku modelu sieci jednokierunkowej był równy 2531 słów. W celu generowania macierzy wykorzystana została biblioteka Tensorflow, natomiast dla lepszej czytelności zostały one zwizualizowane przy użyciu biblioteki *seaborn* języka Python [21]. Ze względu na brak znaczących postępów w trakcie treningu, dla modelu prostej sieci jednokierunkowej przedstawiona została jedynie macierz wygenerowana po zakończeniu treningu (rysunek 6).



Rys. 6. Macierz błędu dla sieci jednokierunkowej po wykonaniu 18 tysięcy kroków

Macierz błędu wygenerowana po zakończeniu procesu uczenia maszynowego prostej sieci jednokierunkowej wykazała najlepszą poprawność klasyfikowania spośród wszystkich utworzonych macierzy dla tej sieci. Walidacja wykazała niespełna 60% zgodności. Zauważyć można, że na głównej przekątnej znajdują się największe liczby występujące w danych kolumnach. Oznacza to, ze polecenia najczęściej były rozpoznawane prawidłowo, a nie przypisywane do innych klas. Należy jednak zwrócić uwagę, że w macierzy występują wysokie wartości poza główną przekątną, co świadczy o tym, że model w dalszym ciągu często popełnia błędy klasyfikując komendy głosowe. Najlepiej rozpoznawana była klasa reprezentująca ciszę, po czym można wywnioskować, że brak sygnału dźwiękowego jest najłatwiejszy dla rozpoznania przez model. W aplikacji testowej użyty został tylko model, który wykonał wszystkie zaplanowane kroki. Wykorzystanie modelu w aplikacji zostało przetestowane w następujących aspektach:

- prawidłowe rozpoznawanie poleceń z wyciszoną muzyką,
- prawidłowe rozpoznawanie poleceń z muzyką w tle odtwarzaną na niewielkim poziomie głośności (25%),
- prawidłowe rozpoznawanie poleceń z muzyką w tle odtwarzaną na średnim poziomie głośności (50%),
- prawidłowe rozpoznawanie poleceń z muzyką w tle odtwarzaną na wysokim poziomie głośności (100%),
- nie wykonywanie operacji przez aplikację przypisanych do poleceń głosowych podczas odtwarzania muzyki samoczynnie, bez wypowiadania poleceń przez osobę testującą,
- rozpoznawanie poleceń z podłączonym zestawem słuchawkowym.

Tabela 2 prezentuje wyniki przeprowadzonego testu. Wskazują one, które przypadki wystąpiły podczas korzystania z wytrenowanego modelu jednowarstwowej sztucznej sieci jednokierunkowej.

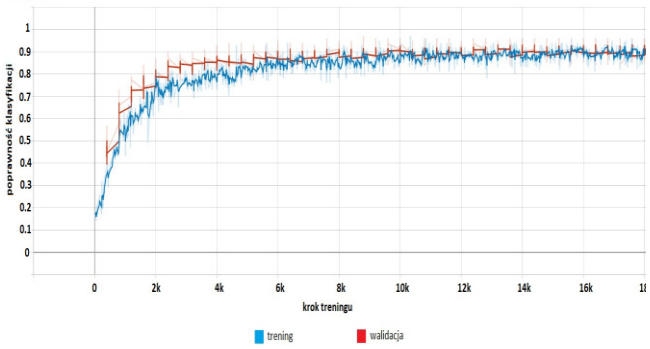
Tabela 2. Wyniki testów dla modelu jednowarstwowej sieci jednokierunkowej

Test	Rezultat
Rozpoznawanie poleceń z wyciszoną muzyką	-
Rozpoznawanie poleceń z muzyką na poziomie głośności 25%	-
Rozpoznawanie poleceń z muzyką na poziomie głośności 50%	-
Rozpoznawanie poleceń z muzyką na poziomie głośności 100%	-
Brak samoczynnego wykonywania operacji	+
Rozpoznawanie poleceń z podłączonym zestawem słuchawkowym	-

Zastosowany model sieci jednokierunkowej nie radzi sobie z poprawną klasyfikacją poleceń. Poziom głośności odtwarzania dźwięku nie ma znaczenia, gdyż żadna operacja nie została wykonana.

5.2. Konwolucyjna sieć nueronowa

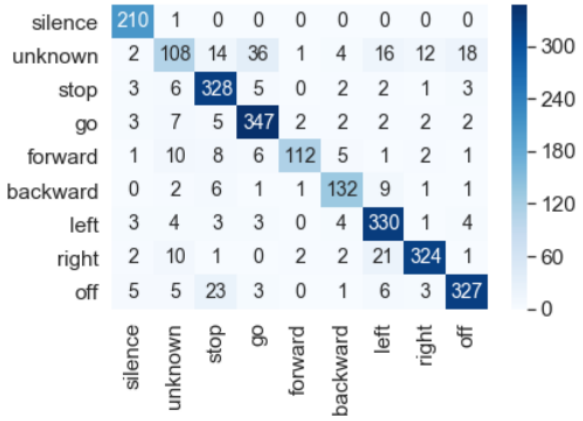
Kolejnym algorytmem, który został wykorzystany do badań jest konwolucyjna sieć neuronowa. Jest ona bardziej skomplikowana od wcześniej opisywanej jednowarstwowej sieci jednokierunkowej, dlatego też trening tej sieci był bardziej czasochłonny, trwał około 30 godzin. Rysunek 7 przedstawia przebieg procesu uczenia maszynowego sieci konwolucyjnej.



Rys. 7. Przebieg treningu sieci konwolucyjnej

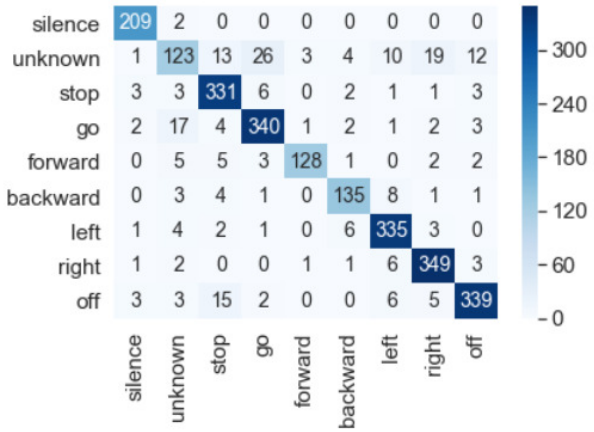
Trening opisywanej sieci przebiegał bardzo efektywnie w początkowej części procesu uczenia maszynowego. Do około 3 tysiąca kroków widoczny jest znaczny wzrost poprawności klasyfikacji wraz z postępem treningu. W kolejnych krokach sieć w dalszym ciągu poprawia wynik zgodności w rozpoznawaniu słów, jednak wzrost ten postępuje już znacznie wolniej. Po przekroczeniu 8 tysięcy kroków treningowych, sieć utrzymuje poprawność klasyfikacji na poziomie w okolicach 85%, a nawet 90%, aż do końca treningu. Zmiana szybkości uczenia modelu w końcowej fazie (po 15 tysiącach kroków treningu) powoduje zmniejszenie wahań poprawności sieci w klasyfikacji poleceń występujących pomiędzy krokami procesu uczenia.

W przypadku modelu konwolucyjnej sieci neuronowej, również zostały wykonane trzy macierze błędu, aby lepiej zobrazować proces uczenia maszynowego. Ponownie w celach testu wykorzystana została walidacja krzyżowa, a współczynnik oznaczający liczebność próbki wynosił jak poprzednio 2531 słów. Pierwsza z macierzy pomyłek przedstawiona została na rysunku 8.



Rys. 8. Macierz błędu dla konwolucyjnej sieci neuronowej po wykonaniu 6 tysięcy kroków treningu

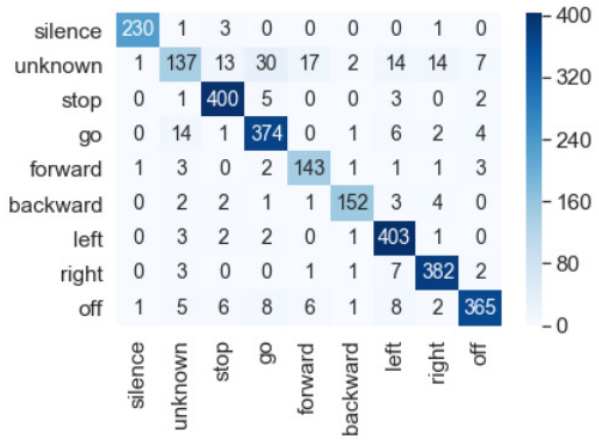
Zdecydowana większość poleceń została poprawnie sklasyfikowana. Na tle całej macierzy najwięcej błędów dostrzec można w klasie unknown oznaczającej nieznane polecenie, do której to najczęściej błędnie przypisywane były komendy, które model powinien znać. Najmniej pomyłek wystąpiło w przypadku, gdy rzeczywistą instancją klasy było słowo forward. Walidacja wykazała, że całkowita poprawność klasyfikacji modelu na tym etapie treningu wynosi już aż 87,6%. Macierz błędu wykonana po 12 tysiącach kroków (rysunek 9) treningu jest bardzo podobna do poprzednio analizowanej macierzy (po 6 tysiącach kroków).



Rys. 9. Macierz błędu dla konwolucyjnej sieci neuronowej po wykonaniu 12 tysięcy kroków treningu

Wszystkie z rozpoznawanych słów są w znaczącym stopniu dobrze klasyfikowane. W dalszym ciągu najwięcej błędów pojawia się w wierszu unknown, gdzie jako nieznane polecenia rozpoznawane są te, które model powinien przydzielać do odpowiadających im klas. Ich liczba jednak nie jest znacząca. Wskaźnik walidacji w tym przypadku wyniósł 90,4%.

W przypadku macierzy wygenerowanej po wykonaniu wszystkich 18 tysięcy kroków (rysunek 10) wskaźnik walidacji jest największy i wynosi 91,9%. Liczby występujące poza główną przekątną są małe, co oznacza, że klasyfikacja odbywa się na zadowalającym poziomie, występuje niewiele błędnie sklasyfikowanych poleceń.



Rys. 10. Macierz błędu dla konwolucyjnej sieci neuronowej po wykonaniu 18 tysięcy kroków treningu

W aplikacji testowej wykorzystane zostały wszystkie trzy modele (zapisane na trzech różnych etapach treningu) konwolucyjnej sieci neuronowej. Testy odbyły się według tych samych aspektów, co w przypadku prostej sieci jednokierunkowej. Wyniki dla modeli tej sieci przedstawia tabela 3.

Sieć konwolucyjna po 6000 kroków treningu bezproblemowo radziła sobie z prawidłowym rozpoznawaniem poleceń w przypadku, gdy odtwarzanie nie było zbyt głośne. W przypadku gdy poziom głośności mediów ustawiony został na 50% maksymalnej wartości, rozpoznawanie poleceń w dalszym ciągu było wykonywane, aczkolwiek konieczne było wypowiadanie komend głośniej niż w poprzednich przypadkach. W momencie, gdy głośność odtwarzania ustawiona została na maksymalną wartość, ponad 50% poleceń było rozpoznawanych nieprawidłowo. Dodatkowo przy wysokim poziomie dźwięku, wynoszącym już około 70% aplikacja wykonywała losowe operacje, pomimo braku wypowiadania rozpoznawanych słów.

Model, który wykonał 12000 kroków, podobnie jak poprzedni nie miał żadnych problemów z rozpoznawaniem poleceń głosowych w przypadku gdy poziom głośności muzyki odtwarzanej w tle nie przekraczał 25%. Głośniejsze wypowiadanie poleceń było niezbędne w celu ich rozpoznania przy maksymalnym poziomie głośności urządzenia. Dodatkowo przy wysokim poziomie głośności aplikacja wykonywała niezamierzone operacje.

Ostatni z testowanych modeli, który wykonał cały trening sprawdził się w aplikacji podobnie jak model po 12 tysiącach kroków treningowych. Rożnica w poprawności klasyfikacji tych modeli była niewielka, stąd też ich działanie w aplikacji testowej jest niemal identyczne. Komendy z łatwością są rozpoznawane prawidłowo do momentu, gdy poziom głośności zbliża się do maksimum.

Tabela 3. Wyniki testów dla modelu konwolucyjnej sieci neuronowej

	Rezultat		
Test	**Sieć konwolucyjna po 6000 krokachów**	**Sieć konwolucyjna po 12000 kroków**	**Sieć konwolucyjna po 18000 kroków**
Rozpoznawanie poleceń z wyciszoną muzyką	+	+	+
Rozpoznawanie poleceń z muzyką na poziomie głośności 25%	+	+	+
Rozpoznawanie poleceń z muzyką na poziomie głośności 50%	+	+	+
Rozpoznawanie poleceń z muzyką na poziomie głośności 100%	-	-/+	-/+
Brak samoczynnego wykonywania operacji	-	-	-
Rozpoznawanie poleceń z podłączonym zestawem słuchawkowym	+	+	+

Wszystkie z testowanych modeli konwolucyjnej sieci neuronowej działają prawidłowo z wykorzystaniem zestawu słuchawkowego, gdyż w tym przypadku mikrofon nie odbiera zakłóceń w postaci odtwarzanej muzyki podczas wprowadzania poleceń głosowych.

6. Dyskusja wyników i wnioski

Trening jednowarstwowej sieci jednokierunkowej wykonany został znacznie szybciej niż sieci konwolucyjnej. Jak jednak wynika z badań, zastosowanie modelu prostej sieci daje znacznie gorsze wyniki w obsługiwaniu aplikacji testowej za pomocą głosu w porównaniu do sieci splotowej. Z rysunków 5 i 7 wynika, że bardziej skomplikowana sieć konwolucyjna już po około 2 tysiącach kroków treningu osiągnęła lepsze rezultaty w klasyfikacji słów niż w stanie była osiągnąć zastosowana zwykła sieć jednowarstwowa. Wykorzystany model konwolucyjnej sieci neuronowej już po wykonaniu 6 tysięcy kroków dał możliwość głosowego sterowania interfejsem. Im więcej kroków treningu zostało wykonanych, tym lepiej spisywał się model sieci splotowej.

Uzyskany wynik jednokierunkowej sieci w pełni połączonej nie jest zadowalający. W przypadku publikacji [10] autorzy wykorzystując podobny algorytm uzyskali znacznie lepszy rezultat (poprawność klasyfikacji około 94%), jednak parametry użytej przez nich sieci różniły się od tych

zastosowanych w niniejszej pracy, co jak można wywnioskować ma znaczący wpływ na efektywność treningu. Sieć konwolucyjna dobrze poradziła sobie z klasyfikacją poleceń, wykorzystany model uzyskał poprawność klasyfikacji na poziomie powyżej 90%. W publikacji [12] gdzie wykorzystany został ten sam algorytm zgodność rozpoznawania odgłosów występujących w ludzkim otoczeniu wyniosła od 60% do 85% w zależności od wykorzystanego zbioru treningowego. Można więc wywnioskować że dane wejściowe dostarczane do sieci mają wpływ na proces treningu.

Analizując uzyskane wyniki można stwierdzić, że postawiona we wprowadzeniu pracy teza jest prawdziwa. Konwolucyjna sieć neuronowa z odpowiednio dobranymi parametrami pozwaliła uzyskać lepsze wyniki klasyfikacji niż jednowarstwowa sieć jednokierunkowa.

Literatura

[1] J. Ye, R. J. Povinelli, M. T. Johnson: „Phenome classification using naive Bayes classifier in reconstructed phase space", IEEE Digital Signal Processing Workshop, 2002

[2] A. Sanchis, A. Juan, E. Vidal: „A Word-Based Naive Bayes Classifier for Confidence Estimation in Speech Recognition", IEEE Transactions on audio, speech and language processing, vol. 20, NO. 2, 2012

[3] N. Smith, M. Gales: „Speech Recognition using SVMs", Cambridge University Engineering Dept, 2002

[4] C. Ittichaichareon, S. Suksri, T. Yingthawornsuk: „Speech Recognition using MFCC", International Conference on Computer Graphics, Simulation and Modeling , 2012

[5] W. M. Campbell, J. P. Campbell, D. A. Reynolds, E. Singer, P. A. Torres-Carrasquillo: „Support vector machines for speaker and language recognition", Computer Speech and Language 20 210–229, 2006

[6] A. Ganapathiraju, J. E. Hamaker, J. Picone: „Applications of Support Vector Machines to speech recognition", IEEE Transactions on signal processing, vol 52, NO. 8, 2004

[7] K. Al Smadi, I. Trrad, T. Al Smadi: „ Artificial Intelligence for Speech Recognition Based on Neural Networks", Journal of Signal and Information Processing, 2015, 6, 66-72, 2015

[8] W. Gevaert, G. Tsenov, V. Mladenov: „Neural networks used for speech recognition",Journal of automatic control, University of Belgrade, vol. 20:1-7 , 2010

[9] M. Tunckanat, R. Kurban S. Sagiroglu: „Voice Recognition Based On Neural Networks", IJCI Proceedings of International Conference on Signal Processing, ISSN 1304-2386, Volume:1, Number:2, 2003

[10] A. Ahad, A. Fayyaz, T. Mehmood: „Speech Recognition using Multilayer Perceptron", Students Conference, ISCON apos:02. IEEE Volume 1, Issue, 16-17, 2002

[11] T. N. Sainath, C. Parada: „Convolutional Neural Networks for Small-footprint Keyword Spotting", Interspeech, 2015

[12] K. J. Piczak: „Environmental Sound Classification With Convolutional Neural Networks", IEEE International Workshop on Machine Learning For Signal Processing, 2015

[13] O. Abdel-Hamid, A. Mohamed, H. Jiang, L. Deng, G. Penn, D. Yu: „Convolutional Neural Networks for Speech Recognition", IEEE/ACM Transactions on audio, speech and language processing, vol. 22, NO. 10, 2014

[14] Tadeusiewicz R.: Sieci neuronowe. Akademicka Oficyna Wydawnicza RM, 1993

[15] Nielsen M.: Neural Networks and Deep Learning. Determination Press, 2015

[16] Goodfellow I., Bengio Y. i Courville A., Deep Learning, 2016

[17] Wprowadzenie do kowolucyjnych sieci neuronowych https://towardsdatascience.com/simple-introduction-to-convolutional-neural-networks-cdf8d3077bac, czerwiec 2019

[18] Strona projektu Tensorflow https://www.tensorflow.org/, czerwiec 2019

[19] Pete Warden: Speech Commands: A Dataset for Limited-Vocabulary Speech Recognition, 2018

[20] Repozytorium biblioteki Tensorflow https://github.com/tensorflow/tensorflow, czerwiec 2019

[21] Dokumentacja biblioteki Seaborn https://seaborn.pydata.org/index.html, czerwiec 2019

JOURNAL of COMPUTER SCIENCES INSTITUTE

JCSI 13 (2019) 310-314
Wysłane: 2019-07-09
Przyjęte: 2019-07-15

Zastosowanie sieci neuronowych do analizy opinii konsumenckich

Roman Mysan*, Ivan Loichuk, Małgorzata Plechawska-Wójcik
Politechnika Lubelska, Instytut Informatyki, Nadbystrzycka 36B, 20-618 Lublin, Polska

Streszczenie. Niniejszy artykuł przedstawia analizę możliwości zastosowania sieci neuronowych do klasyfikacji danych tekstowych w postaci komentarzy. Ponadto przedstawiono wyniki badania dwóch metod optymalizacji sieci neuronowej: Adam i Gradientu. Celem pracy jest przeprowadzenie badań zachowania się sieci neuronowej w zależności od zmiany parametrów oraz ilości danych użytych do nauczania sieci neuronowej. Na potrzeby realizacji tego celu utworzona została aplikacja testowa korzystająca z sieci neuronowej w celu wyświetlenia ogólnej oceny obiektu noclegowego na podstawie dodanych opinii użytkowników.

Słowa kluczowe: sieć neuronowa; TensorFlow; sztuczna inteligencja

*Autor do korespondencji.

Adresy e-mail: roman.mysan@pollub.edu.pl, ivan.loichuk@pollub.edu.pl

Application of neural networks to the analysis of consumer opinions

Roman Mysan*, Ivan Loichuk, Małgorzata Plechawska-Wójcik
Institute of Computer Science, Lublin University of Technology, Nadbystrzycka 36B, 20-618 Lublin, Poland

Abstract. This paper presents an analysis of the possibilities of using neural networks to classify text data in the form of comments. Moreover, results of research of two neural network optimization methods: Adam and Gradient are presented. The aim of the work is to conduct research on the behavior of the neural network depending on the change of parameters and the amount of data used to teach the neural network. To achieve the goal, a test application was created. It uses a neural network to display the overall assessment of the accommodation facility based on the added user feedback.

Keywords: neural network; TensorFlow; artificial intelligence

*Corresponding author.

E-mail addresses: roman.mysan@pollub.edu.pl, ivan.loichuk@pollub.edu.pl

1. Wprowadzenie

Sztuczna inteligencja oraz uczenie maszynowe nie są nowymi technologiami. Te technologie od dawna się rozwijają. Niestety dotychczas nie było możliwości tak szerokiego stosowania mechanizmów uczenia maszynowego, ponieważ komputery nie miały odpowiedniej mocy obliczeniowej. Wraz ze wzrostem mocy obliczeniowej komputerów, ten kierunek zdobywa coraz większą popularność. Dzisiaj każdy kto ma komputer i dostęp do Internetu może stworzyć sieć neuronową i testować ją na danych wejściowych. Można również skorzystać z serwisów takich jak na przykład Google lub Amazon, które umożliwiają uruchomienie sieci neuronowych na swoich serwerach. Oprócz mocy obliczeniowej, nie mniej ważnym apektem jest zestaw danych, na których sieć neuronowa będzie się uczyła. Zestaw danych musi być odpowiednio duży, aby uzyskać wysoką efektywność uczenia.

Organizacje na całym świecie gromadzą duże ilości danych do rozwiązania różnych problemów biznesowych. Sztuczną inteligencję można zdefiniować jako system, który wykonuje określonego rodzaju zadania przy wykorzystaniu wiedzy, otrzymanej na podstawie interpretowania danych wejściowych. Żeby prawidłowo wykonywać zadania, system uczy się na danych wejściowych, przez interpretowanie danych wejściowych uzyskując wiedzę, którą jest w stanie wykorzystać do wykonania zadań.

Celem badania jest analiza efektywności wykorzystania algorytmów sieci neuronowych w nowoczesnych aplikacjach webowych oraz porównanie dwóch metod optymizacji sieci neuronowej: Adam i Gradientu. Zadanie zrealizowano w języku programowania Python oraz biblioteki sieci neuronowej TensorFlow.

2. Przegląd literatury

Wraz ze wzrostem dostępności dokumentów elektronicznych i szybkim rozwojem sieci Internet, automatyczna klasyfikacja danych stała się kluczową metodą organizacji danych i odkrywania wiedzy. Właściwa klasyfikacja dokumentów elektronicznych, wiadomości online, blogów i maili wymaga użycia metod wyszukiwania tekstu, uczenia maszynowego oraz technik przetwarzania języka naturalnego w celu uzyskania znaczącej wiedzy. W artykule *A Review of Machine Learning Algorithms for Text-Documents Classification* [4] autorzy poruszają problematykę reprezentacji tekstu oraz omawiają techniki i metody stosowane w klasyfikacji danych tekstowych. Jedną z takich technik jest reprezentacja tekstu (*ang. text representation*), która służy do redukcji złożoności dokumentów i ułatwiania ich automatycznej obsługi. Co do technik nauczania maszynowego to autorzy opisują takie metody nauczania jak: Rocchio's Algorithm, Decision Tree, Decision Rules Classification, Artificial Neural Network,

Genetic Algorithm, Support Vector Machine (SVM), K nearest neighbor (k-NN).

Obecnie istnieje wiele metod optymizacji sieci neuronowych. Należą do nich: Adagrad, RMSProp, Adam, SGDNesterov, AdaDelta, metoda Gradientu. Użycie metody Adam zostało opisane w artykule *Adam: a method for stochastic optimization* [8]. Autorzy artykułu analizują teoretyczne właściwości konwergencji algorytmu oraz porównują współczynnik zbieżności z najlepszymi znanymi wynikami optymizacji wypukłej. Wyniki pokazały, że w praktyce Adam działa lepiej i jest korzystniejszy w porównaniu z innymi metodami optymizacji stochastycznej.

Każda z wymienionych metod jest wspierana przez bibliotekę TensorFlow. Jednym z przykładów zastosowania biblioteki jest użycie sieci neuronowej do przewidywania przepływu ruchu drogowego. Zastosowanie algorytmu zostało opisane przez autorów artykułu *Deep Neural Networks for traffic flow prediction* [5]. Opisana w artykule głęboka sieć neuronowa korzysta z danych o ruchu drogowym w czasie rzeczywistym. Jest to pionierski przykład takich badań. Sugerowany model wykorzystuje rzeczywiste dane ruchu agregowane co pięć minut. Wyniki pokazują, że współczynnik dokładności modelu wynosi około 99%.

Po przeanalizowaniu publikacji naukowych które dotyczą tematu sztucznej inteligencji można stwierdzić, że istnieją publikacje [4, 8] prezentujące wyniki badań wydajności sieci neuronowych przy użyciu różnych metod lub przy zmianie parametrów konfiguracyjnych sieci. W niektórych artykułach naukowych [5] udało się znaleźć przykłady użycia biblioteki TensorFlow w różnych dziedzinach życia.

3. Opis przebiegu badań

Głównym zadaniem aplikacji stworzonej na potrzeby badań jest zbieranie opinii internautów o określonych obiektach noclegowych i generowanie zestawień według określonych kategorii na podstawie analizy przeprowadzanej za pomocą sieci neuronowej. W celu realizacji badań i implementacji sieci neuronowej, zostaną wykorzystane biblioteki: TensorFlow, TensorBoard, NLTK. TensorFlow pozwala wykonywać różnego rodzaju operacje uczenia maszynowego z dużą wydajnością. Wytrenowane modele sieci neuronowej będą zastosowane w aplikacji webowej.

Sieć neuronowa przygotowana na potrzeby badań powinna być tak skonfigurowana i nauczona, aby mogła rozróżnić kategorie (lub kilka kategorii) do której należy komentarz i rodzaj komentarza (pozytywny lub negatywny). Analizując opinie turystów z portalu **booking.com** wybrano cztery kategorie opisu noclegów które najbardziej interesują turystów:

- lokalizacja obiektu noclegowego,
- czystość i komfort,
- personel,
- wyżywienie.

Jednym z najkosztowniejszych, z punktu widzenia nakładu pracy, etapów badania jest zbiór danych do uczenia się. Podczas przygotowania danych ważna była nie tylko ilość danych, ale i ich jakość. Aby system był gotowy do pracy z dużą ilością danych w postaci opinii użytkowników portalu, informacje były zebrane z największych, dostępnych portali noclegowych. Pozwoliło to na nauczenie sieci na podstawie prawdziwych opinii.

Dane przygotowane na potrzeby uczenia się sieci neuronowej zostały podzielone na 8 kategorii (uwzględniając pozytywne i negatywne opinie dla każdej z czterech kategorii). Pierwszy zestaw danych zawiera około 1700 rekordów, natomiast drugi zestaw zawiera już około 12 tysięcy rekordów.

Aby zbadać wpływ ilości danych na efekt uczenia, sieć neuronowa była trenowana na dwóch zestawach danych. Poza tym, aby sprawdzić jaki wpływ na efekt uczenia ma liczba epok, sieć neuronowa była uczona na 10, 20, 50 oraz 100 epokach.

Zostaną również przeprowadzone badania efektywności metod optymalizacji Adam oraz gradientu w zadaniu klasyfikacji tekstu. Zestaw danych do uczenia będzie obejmował 12 tysięcy komentarzy, zestaw danych testowych będzie obejmował około jednego tysiąca komentarzy.

Wszystkie badania zostaną przeprowadzone z użyciem sieci neuronowej, która posiada jedną warstwę ukrytą z szybkością uczenia 0.001.

Nauczona sieć neuronowa była testowana za pomocą zbioru danych walidacyjnych, niezależnych od zbioru uczącego. Ilość danych testowych jest dosyć duża, aby uzyskać dokładne wyniki testowania.

Podczas analizy efektywności uczenia sieci neuronowych zostaną uwzględnione: wartość odchylenia w procesie uczenia, wartość odchylenia w wyniku końcowym oraz dokładność otrzymaną po przetestowaniu nauczonej sieci neuronowej.

4. Metody optymalizacji gradientu oraz Adam

Do optymalizacji wartości odchylenia w sieciach neuronowych najczęściej wykorzystuje się metodę gradientu, która przy każdej iteracji algorytmu, szuka kierunku, w którym wartość odchylenia maleje. Wadą klasycznej metody gradientu jest to, że podczas procesu uczenia sieci neuronowej (optymalizacji wartości odchylenia) wykorzystuje stałą wartość parametru szybkości uczenia przez cały proces uczenia, co przy małej liczbie epok może spowodować nie satysfakcjonujący wynik.

Aby rozwiązać problem metody gradientu, Diederik P. Kingma oraz Jimmy Lei Ba opracowali metodę optymalizacji o nazwie Adam i opisali ją w swojej pracy: „Adam: a method for stochastic optimization" [2]. Metoda optymalizacji Adam ma wysoką wydajność i jest bardzo efektywna w optymalizacji. Główną zaletą tej metody jest to, że oblicza ona wartość parametru szybkości uczenia do każdej wagi sygnału, który łączy się z neuronem i podczas uczenia sieci Adam może odpowiednio dopasowywać tą wartość do każdego sygnału.

5. Prezentacja rezultatów badań

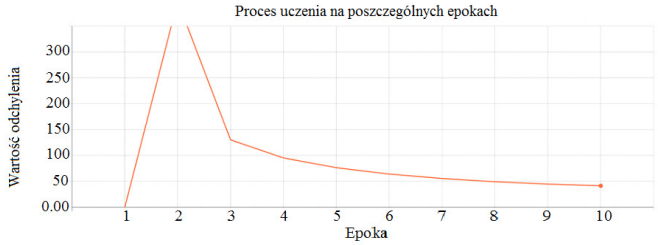
5.1. Badanie 1

Z zestawem danych, który obejmuje 1700 komentarzy, otrzymano dokładność 42%, a wartość odchylenia udało się zoptymalizować do 38. Wyniki uczenia przedstawiono w tabeli 1. Patrząc na otrzymaną dokładność można stwierdzić, że wynik nie jest zadowalający, ale z takim małym zestawem danych trudno otrzymać lepszy. Dane muszą być wysokiej jakości i posiadać dużo cech (ang. *features*), na których sieć będzie się uczyła klasyfikować komentarze. Poza tym, analizując otrzymaną wartość odchylenia (38), można stwierdzić, że 10 epok uczenia było wartością zbyt małą.

Tabela 1. Wyniki procesu uczenia w Badaniu 1

Wartość odchylenia	Dokładność (%)
38,88	42

Analizując proces uczenia, który przedstawiono na rysunku 1, można zauważyć duży skok w wartości odchylenia pomiędzy pierwszą a drugą epoką, ale po 2 epoce następuje duży spadek wartości odchylenia. Podczas całego procesu uczenia sieć rozwijała się, ale po 9 epoce widać, że progres w uczeniu zmniejszył się.



Rys. 1. Proces uczenia na poszczególnych epokach

Na potrzeby analizy wpływu liczby epok na efekt uczenia, liczbę epok było zwiększono kilka razy: do 20 epok, do 50 epok i do 100 epok. Wynik uczenia przedstawiono w tabeli 2.

Tabela 2. Wynik uczenia sieci dla poszczególnej liczby epok

Liczba epok	Wartość odchylenia	Dokładność(%)
10	38,80	42
20	32,35	42
50	31,45	47
100	31,37	47

Analizując wyniki z tabeli 2 otrzymane po zwiększeniu liczby epok, można stwierdzić, że zwiększenie liczby epok miało pozytywny wpływ na efekt uczenia. Porównując wynik dokładności, który otrzymano po 10 epokach uczenia z wynikiem otrzymanym po 100 epokach widać, że dokładność wzrosła o 5%.Wartość odchylenia udało się zoptymalizować z 38 do 31. Poza tym, analizując pozostałe wyniki, można zauważyć, że po uczeniu, które trwało 50 epok otrzymano prawie taki sam wynik, jak po uczeniu przy uczeniu po 100 epokach. W tym przypadku wartość odchylenia zmniejszyła się o bardzo małą wartość, wzrost dokładności jest niewielki.

Podsumowując można stwierdzić, że liczba epok w tym badaniu nie miała dużego wpływu na wynik końcowy, a optymalną liczbą epok do uczenia jest 50. 100 epok jest natomiast zbyt dużą liczbą, bo wynik końcowy prawie się nie różni od wyniku uczenia po 50 epokach.

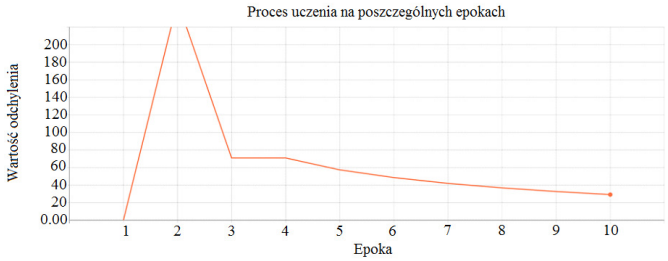
5.2. Badanie 2

Aby zbadać wpływ ilości danych na efekt uczenia, następne badania były przeprowadzone z drugim zestawem danych (który obejmuje 12 tysięcy komentarzy). Wynik uczenia z tym zestawem danych przedstawiono w tabeli 3. Analizując wynik widać, że dokładność po 10 epokach uczenia stanowi 58%, a wartość odchylenia udało się zoptymalizować do 26. Porównując otrzymany wynik z wynikiem badania 1 widać, że wraz z większym zestawem danych otrzymano o wiele lepszy efekt uczenia. Po zwiększeniu ilości danych dokładność wzrosła o 16%, a wartość odchylenia zmniejszyła się o 12.

Tabela 3. Wynik uczenia sieci w badaniu 2

Wartość odchylenia	Dokładność(%)
26,19	58

Analizując proces uczenia, który przedstawiono na rysunku 2, można zauważyć, że wykres jest bardzo podobny do wykresu dla badania 1 (rysunek 1). Jednak w badaniu 1 zauważono, że po 9 epokach progresu w uczeniu prawie nie było, natomiast w tym przypadku dobrze widać, że podczas całego procesu uczenia sieć neuronowa ma progres w optymalizacji wartości odchylenia. Aby uzyskać jeszcze lepszy efekt uczenia, liczba epok musi być zwiększona.



Rys. 2. Proces uczenia na poszczególnych epokach

Poprzednio stwierdzono, że efekt uczenia z większym zestawem danych może być lepszy po zwiększeniu liczby epok. Aby zweryfikować tą potencjalną zależność, liczbę epok zwiększono kilka razy. Wyniki przedstawiono w tabeli 4.

Tabela 4. Wynik uczenia sieci dla poszczególnej liczby epok

Liczba epok	Wartość odchylenia	Dokładność(%)
10	26,19	58
20	9,77	61
50	3,07	64
100	2,29	64

Analizując wyniki można stwierdzić, że w tym przypadku zwiększenie liczby epok również miało pozytywny wpływ na efekt uczenia jak i w badaniu 1. Zwiększenie liczby epok spowodowało wzrost dokładności o 6% i duży spadek wartości odchylenia. Patrząc na wyniki w tabeli 4 można

zauważyć, że maksymalnie wysoką dokładność udało się otrzymać po 100 epokach uczenia, minimalną wartość odchylenia też otrzymano po 100 epokach uczenia. Poza tym, analizując pozostałe wyniki można zauważyć, że otrzymana dokładność przy uczeniu, które trwało 50 epok nie różni od dokładonści przy uczeniu, które trwało 100 epok. Sytuacja jest bardzo podobna jak w badaniu 1, gdy po 50 epokach otrzymane wyniki były podobne jak te uzyskane przy 50 epokach.

Z danego badania wynika, że zwiększenie liczby epok miało pozytywny wpływ na efekt uczenia, a optymalną liczbą epok jest 50.

5.3. Badanie 3

Wynik uczenia sieci neuronowej z wykorzystaniem metody gradientu przedstawiono w tabeli 5.

Tabela 5. Wynik uczenia z użyciem metody gradientu dla poszczególnej liczby epok

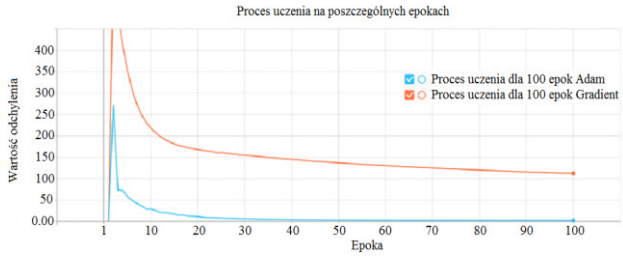
Liczba epok	Wartość odchylenia	Dokładność(%)
10	209,21	15
20	157,49	27
50	145,12	29
100	112,08	38

Analizując wynik można stwierdzić, że sieć w procesie uczenia cały czas się doskonaliła, bo wyniki pokazują, że dokładność cały czas rosła, a wartość odchylenia zmniejszała się. W porównaniu z wynikami, które otrzymano przy uczeniu sieci z wykorzystaniem metody Adam, które przedstawiono w tabeli 6, można zauważyć, że podczas uczenia z wykorzystaniem metody Adam uzyskano o wiele lepszy wynik klasyfikacji.

Tabela 6. Wynik uczenia z użyciem metody Adam dla poszczególnej liczby epok

Liczba epok	Wartość odchylenia	Dokładność(%)
10	26,19	58
20	9,77	61
50	3,07	64
100	2,29	64

Dokładność, którą uzyskano przy uczeniu sieci neuronowej za pomocą algorytmu Adam jest o wiele większa w porównaniu z dokładnością uzyskaną podczas uczenia z wykorzystaniem metody gradientu. Dokładność, po 10 epokach z wykorzystaniem metody gradientu stanowi tylko 15%, w porównaniu do dokładności otrzymanej metodą Adam, która stanowi 58%. Duża różnica jest nie tylko w dokładności, ale również w wartości odchylenia, którą otrzymano w wyniku uczenia sieci. Wartość odchylenia po 10 epokach z wykorzystaniem metody gradientu stanowi 209, natomiast metoda Adam zoptymalizowała wartość odchylenia do 26. Analizując kolejne wyniki uczenia, po zwiększeniu liczby epok z 10 do 20, widać, że dokładność przy uczeniu z wykorzystaniem metody Adam wzrosła o 3%, ale patrząc na wynik, który uzyskano przy pomocą metody gradientu, widać duży wzrost dokładności, z 15% do 27%. Dokładność nadal jest o wiele niższa w porównaniu do otrzymanej przy pomocy metody Adam. Natomiast, analizując wartość odchylenia po zwiększeniu liczby epok, można zauważyć duży spadek wartości odchylenia w obu przypadkach. Warto zauważyć, że po 50 epokach metoda Adam straciła progres w optymalizacji wartości odchylenia, wówczas progresu w dokładności też nie było. Porównując te wyniki do tych uzyskanych z wykorzystaniem metody gradientu, można zauważyć, że metoda gradientu cały czas miała progres.



Rys. 3. Porównanie wartości odchylenia

Analizując wykres na rysunku 3, który ilustruje proces uczenia dla obu metod, widać, że metoda Adam uzyskała optymalną wartość odchylenia dużo wcześniej niż metoda gradientu. Po 30 epoce, metoda Adam prawie nie optymalizowała wartość odchylenia. W przypadku metody gradientu, cały czas widać było progres, ale wartość odchylenia była o wiele większa niż dla metody Adam. W wyniku uczenia sieci metoda Adam uzyskała optymalną wartość odchylenia, ale dla metody gradientu epok było za mało, aby maksymalnie optymalizować wartość odchylenia.

Podsumowując, metoda Adam okazała się o wiele lepsza niż metoda gradientu. Metoda gradientu uzyskała o wiele gorsze wyniki uczenia niż metoda Adam przy takiej samej liczbie epok. Oczywistym jest fakt, że dla metody gradientu epok było za mało. Natomiast, dla metody Adam liczba epok była wystarczająca, aby uzyskać optymalną wartość odchylenia oraz dokładność. Poza tym, metoda Adam pokazała wysoką dokładność oraz niską wartość odchylenia już po 10 epokach uczenia, gdy metoda gradientu dawała o wiele gorsze wyniki.

6. Wnioski

Po przeanalizowaniu wyników badań 1 oraz 2 można stwierdzić, że zestaw danych ma duży wpływ na efekt uczenia. Poza tym, na efekt uczenia ma wpływ nie tylko ilość danych do uczenia, ale i jakość tych danych. Porównując wyniki badania 1 oraz 2 było widać, że wraz ze zwiększeniem danych do uczenia dokładność wzrosła, natomiast 64% dokładności nie jest wynikiem satysfakcjonującym. Takie zachowanie sieci jest spowodowane słabą jakością danych w drugim zestawie danych (który obejmuje 12 tysięcy komentarzy). Poza tym badania pokazały, że liczba epok ma duży wpływ na wyniki uczenia, dlatego, liczbę epok trzeba dopasowywać, aby uzyskać jak najlepsze wyniki uczenia.

Badanie 3 miało na celu porównanie metod optymalizacji Adam oraz gradientu. Z otrzymanych wyników badania 3 widać, że metoda optymalizacji Adam daje wysoką wydajność i jest o wiele lepsza od klasycznej metody gradientu. Przy uczeniu z wykorzystaniem metody

optymalizacji Adam udało się uzyskać optymalną wartość odchylenia o wiele wcześniej w porównaniu do uczenia z wykorzystaniem metody gradientu. Dla metody gradientu epok było za mało, natomiast dla metody Adam epok było wystarczająco (nawet za dużo), aby uzyskać optymalną wartość odchylenia.

Literatura

[1] M. Fábio, M. F. Alan, Neural network programming with Java, Packt Publishing, 2016

[2] P. D. Kingma, J. Lei Ba: Adam: a method for stochastic optimization, Published as a conference paper at ICLR 2015

[3] B. Yoshua, Practical Recommendations for Gradient-Based Training of Deep Architectures, Version 2, Sept. 16th, 2012

[4] K. Aurangzeb, B. Baharum, L. Lam Hong*, K. Khairullah, A Review of Machine Learning Algorithms for Text-Documents Classification, Journal of advances in information technology, vol. 1, no. 1, february 2010

[5] Y. Hongsuk, J. HeeJin, B. Sanghoon, Deep Neural Networks for traffic flow prediction, Published in: 2017 IEEE International Conference on Big Data and Smart Computing (BigComp)

[6] S. Ruder, An overview of gradient descent optimization algorithms∗, Insight Centre for Data Analytics, NUI Galway Aylien Ltd., Dublin, 15 Jun 2017

[7] M. Kaut, S. W. Wallace, Evaluation of scenario-generation methods for stochastic programming, March 2007

[8] https://www.wired.com/2016/06/how-google-is-remaking-itself-as-a-machine-learning-first-company/ [22.06.2019]

[9] P. Lula, Text-mining jaką narzędzie pozyskiwania informacji z dokumentów tekstowych, Akademia Ekonomiczna w Krakowie, Katefra Informatyki, 2005

[10] A. Joulin, E. Grave, P. Bojanowski, T. Mikolov, Bag of Tricks for Efficient Text Classification, 6 Jul 2016

[11] Z. Min-Ling, Z. Zhi-Hua, Multilabel Neural Networks with Applications to Functional Genomics and Text Categorization, 28 August 2006

[12] S. Fabrizio, Machine learning in automated text categorization, March 2002

JOURNAL of COMPUTER SCIENCES INSTITUTE

JCSI 13 (2019) 315-322
Wysłane: 2019-06-14
Przyjęte: 2019-06-29

Analiza porównawcza szkieletów dedykowanych projektowaniu aplikacji korporacyjnych

Katarzyna Curyła*, Karolina Habernal
Politechnika Lubelska, Instytut Informatyki, Nadbystrzycka 36B, 20-618 Lublin, Polska

Streszczenie. Celem artykułu jest przedstawienie analizy porównawczej szkieletów do projektowania aplikacji korporacyjnych. Na potrzeby analizy zaprojektowana została aplikacja do obiegu dokumentów w kancelarii prawnej. W analizie porównawczej przedstawiono szkielet TOGAF oraz siatkę Zachmana. Szkielety zostały poddane badaniu na grupie badawczej na podstawie określonych kryteriów podzielonych na pięć grup. Zostały zdefiniowane następujące grupy kryteriów: implementacji, testowania, czasochłonności, dokumentacyjne oraz planowania harmonogramu. Z wyników badań wynika, że siatka Zachmana umożliwia przedstawienie procesu projektowania architektury biznesowej z różnych perspektyw: uwzględniających zarówno wizje klienta jak i projektanta oraz programisty, natomiast szkielet TOGAF prezentuje głównie podejście biznesowe uwzględniające cele i strategie przedsiębiorstwa. Przeprowadzona analiza została zrealizowana na grupie badawczej dwóch ekspertów dziedzinowych.

Słowa kluczowe: architektura korporacyjna; szkielet TOGAF; siatka Zachmana

*Autor do korespondencji.

Adresy e-mail: khabernal@gmail.com, kasiacuryla@gmail.com

Comparative analysis of frameworks dedicated to enterprise designing

Katarzyna Curyła*, Karolina Habernal
Institute of Computer Science, Lublin University of Technology, Nadbystrzycka 36B, 20-618 Lublin, Poland

Abstract. The aim of this paper is a comparative analysis of frameworks for designing corporate applications. For analytic reasons the application for document circulation within a law firm was designed. The presented comparative analysis includes TOGAF framework as well as Zachman framework. The analysis was conducted on group of experts based on evaluation criteria divided on several groups. Five groups of criteria were defined: implementation, testing, time-consuming, documentation and schedule planning. As the research shows, Zachman framework presents a versatile approach towards designing corporate architecture, because it presents not only the client's vision but also the visions of a designer and a programmer, whereas TOGAF framework focuses mainly on business approach regarding aims and strategy of an enterprise. The aforementioned research was carried out on a group on a two domain experts.

Keywords: enterprise architecture; TOGAF; Zachman

*Corresponding author.

E-mail addresses: khabernal@gmail.com, kasiacuryla@gmail.com

1. Wprowadzenie

Systemy informatyczne stały się nieodłączną częścią przedsiębiorstw. Dzięki swojej złożoności ułatwiają pracę m.in. poprzez przechowywanie i obieg dokumentów. Systemy informatyczne oparte na architekturze korporacyjnej odpowiadają potrzebom przedsiębiorstwa z perspektywy funkcji i celów biznesowych.

Architektura korporacyjna odnosi się nie tylko do systemów informatycznych w przedsiębiorstwie, ale także do całego przedsiębiorstwa. Zawiera logikę organizacji w połączeniu z infrastrukturą IT (ang. Information Technology). Architektura korporacyjna przedstawia plan koncepcyjny firmy. Pozwala jasno określić, jak organizacja może się rozwijać i osiągnąć wyznaczone cele [11]. Jedną z korzyści wdrożenia opisywanej architektury jest wprowadzenie zmian strategicznych w przedsiębiorstwie. Dzięki temu możliwe jest skuteczne wykorzystanie i zarządzanie informacją w przedsiębiorstwie. Projektowanie aplikacji w oparciu o architekturę korporacyjną sprawia, że systemy projektowane dla przedsiębiorstw spełniają potrzeby biznesowe co pośrednio wpływa na ich rozwój [11].

Do projektowania aplikacji lub systemów przy pomocy architektury korporacyjnej wykorzystywane są szkielety, które ułatwiają proces projektowanie systemów. Wybór szkieletu jest równie ważny jak faza projektowa, ponieważ to właśnie na nim będzie bazował system oraz rozwój przedsiębiorstwa.

Do znanych narzędzi używanych do projektowania aplikacji korporacyjnych należą szkielety TOGAF (ang. The Open Group Architecture Framework) oraz Zachman. Szkielet TOGAF pokazuje odwzorowanie celu i strategii przedsiębiorstwa na projektowaną aplikację. Struktura siatki Zachmana umożliwia klasyfikację architektury organizacji.

Po zapoznaniu się z dokumentacją wymienionych szkieletów nasuwa się pytanie o ich możliwości zastosowania w praktyce. Niniejszy artykuł przedstawia wyniki analizy porównawczej szkieletów TOGAF i Zachman. Porównanie

szkieletów zostało przedstawione na podstawie zaprojektowanej aplikacji dla przedsiębiorstwa o zdefiniowanych celach i strategiach biznesowych.

Hipoteza badawczą zdefiniowano następująco: szkielet TOGAF oraz siatka Zachmana są równie dobre do projektowania architektury aplikacji korporacyjnej.

W literaturze można znaleźć kilka opracowań dedykowanych projektowaniu aplikacji korporacyjnych za pomocą szkieletu TOGAF lub siatki Zachmana.

W artykule [8] poddano badaniu jakościowemu metodykę projektowania architektury korporacyjnej. Została opisana architektura zaprojektowana na podstawie szkieletu TOGAF. Przeanalizowano jego wykorzystanie w przedsiębiorstwie. Na podstawie badań autor wywnioskował, że użyte elementy badanego szkieletu nie były przydatne w badanej organizacji. Zaprezentowane badania kwestionują użycie szkieletu TOGAF podczas tworzenia architektury korporacyjnej.

Kolejnym artykułem, który został przeanalizowany była praca badawcza [10]. W tej pracy autorzy zbadali szkielety do projektowania architektury korporacyjnej TOGAF oraz DoDA (ang. The Department of Defense Architecture Framework). Natomiast DoDAF zawiera mapę pięćdziesięciu dwóch modeli. Autor łączy mapę tworzoną w szkielecie DoDAF oraz ramy architektury, które obejmuje TOGAF. Autor proponuje proces projektowania aplikacji odwzorowującej potrzeby przedsiębiorstwa. W podsumowaniu przedstawione zostały wyniki wdrożenia ram dostosowanych i wykorzystanych w projekcie dla przedsiębiorstwa budowlanego.

W artykule [1] autor zaproponował modelowanie oparte na procesach biznesowych do analizy wymagań integracji przedsiębiorstw przy użyciu struktury Zachman. Celem tego artykułu było zaprojektowanie tzw. modelu onboarding, aby zapewnić rozwiązanie zgodne ze strukturą i działaniami przedsiębiorstwa. Onboarding jest to uczenie pracownika podstaw i umiejętności wymaganych w przedsiębiorstwie. Następnie przedstawiono opis przedsiębiorstwa oraz przeprowadzone badania.

W artykule [14] przeprowadzono badania wykorzystując grę HayDay. Jest to gra biznesowa związana z rolnictwem i rozwojem przedsiębiorstw rolnych. Ramy Zachmana zastosowane w tym badaniu obejmują zakres, przedsiębiorstwo, model i system modelowy. W badanej grze zastosowano cztery kryteria biznesowe, które zostały przedstawione jako procesy: zamawiania, przechowywania, produkcji oraz dostaw. W artykule znajduje się wprowadzenie, które zawiera opis architektury Zachmana oraz jak wygląda jej struktura. Następnie autor przedstawia utworzone diagramy. Dzięki zaprojektowaniu architektury korporacyjnej gry Hayday, autor otrzymał szczegółowe informacje na temat aspektów organizacyjnych, procesów i strategii biznesowych w grze Hayday.

Pozostała część niniejszego artykułu obejmuje część teoretyczną oraz praktyczną. Część teoretyczna przedstawia omawiane w artykule szkielety. Natomiast część praktyczna prezentuje zastosowaną metodę badawczą, opis przykładowych rozwiązań oraz wyniki analizy porównawczej wyżej wymienionych szkieletów zrealizowanej w oparciu o projekty wykonane przy ich pomocy.

2. Materiały

W tej części artykułu zostały przedstawione najważniejsze aspekty teoretyczne badanych szkieletów. Jako materiały zostały zdefiniowane zastosowane szkielety czyli siatka Zachmana oraz szkielet TOGAF.

2.1. TOGAF

TOGAF to szkielet wykorzystywany do tworzenia architektury aplikacji korporacyjnych. Zawiera formalny lub szczegółowy opis systemu wdrażanego w przedsiębiorstwie, który przedstawia różne poziomy komponentów. TOGAF zawiera również przewodnik opisujący wdrożenie systemu. Odzwierciedla strukturę i możliwości przedsiębiorstwa. Bazuje na iteracyjnym modelu procesu wspieranego przez najlepsze praktyki i zestaw architektury wielokrotnego użytku. TOGAF pokazuje strukturę komponentów, ich relacje, zasady, proces wdrożenia oraz opis ewolucji systemu w czasie [7]. Uzupełnia standardy organizacji i może zintegrować je w szeroki i holistyczny sposób [13]. Szkielet TOGAF definiuje metody rozwoju architektury, w tym metodę ADM (ang. Architecture Development Method), narzędzia do tworzenia architektury korporacyjnej oraz zbiór najlepszych praktyk projektowych, które pozwalają na rozwiązywanie złożonych problemów [11]. Dzięki metodzie ADM (ang. Architecture Development Method) oraz narzędziom przedsiębiorstwa mogą zdefiniować sposób, w jaki będą usprawniać swoje działania. Pozwalają na zaprojektowanie procesów i infrastruktury krytycznej dla obecnych i przyszłych operacji [13]. Cykl ADM to inkrementacyjna i iteratywna metoda architektury biznesowej, danych, aplikacji i technologii [13]. Przedstawia jak krok po kroku zaprojektować i zrealizować architekturę korporacyjną dla organizacji, spełniającą określone wymagania biznesowe [6]. Decyzje podejmowane w metodzie ADM bazują na zasobach [11]. Pozwala to na zaprojektowanie architektury strategicznej lub całego potencjału. Efektem działań w ramach metody ADM jest wprowadzenie zmian na poziomie biznesowym oraz IT [6]. Architecture Development Method składa się z ośmiu głównych faz odnoszących się zarówno do architektury organizacji :

- *Faza wstępna* – obejmuje przygotowanie, określenie zakresu architektury oraz rozpoczęcie przygotowania projektu. Zawiera: poznanie środowiska, w którym będzie tworzony projekt, określenie zakresu, struktury i zasad projektu [11].
- *Wizja architektury* – polega na utworzeniu wizji architektury. Inicjuje pierwszą iteracje procesu tworzenia architektury systemu [11]. Określa cele, potrzeby, ograniczenia, definicje zakresu oraz ryzyka [6]. Sprawdza kontekst biznesowy projektu.
- *Architektura biznesowa* – bazuje na opisie organizacji przedsiębiorstwa. Zawiera opis procesów biznesowych. Jest to przygotowanie modeli referencyjnych oraz

narzędzi, które będą wykorzystywane do zaprojektowania architektury korporacyjnej. Na tym etapie opracowana zostaje również bazowa i docelowa architektura dla organizacji [6]. W skład tej fazy wchodzą: opis struktury projektowanego systemu, cele, funkcje, usługi, procesy i role biznesowe oraz korelacja pomiędzy funkcjami organizacji .

- *Architektura systemów informatycznych* ta faza przedstawia podstawową organizację systemu IT, która jest używana w przedsiębiorstwie [11]. Polega na rozbiciu architektury biznesowej na dwa podprocesy: architekturę aplikacji oraz architekturę danych [6]. Nie zawsze konieczne jest określanie dwóch rodzajów architektury, zwykle zależy to od zakresu projektu. Architektura aplikacji może być rozwijana niezależnie od architektury danych, mogą też być rozwijane równolegle [11].
- *Architektura technologii* – podstawowy element każdej organizacji, która wykorzystuje systemy IT. Faza ta opisuje sprzęt oraz oprogramowanie używane w przedsiębiorstwie [11].
- *Możliwości i rozwiązania* – ten etap polega na opracowaniu planu wdrożenia głównych części implementacji [11].
- *Planowanie migracji* etap ten polega na zatwierdzeniu architektury, która została zidentyfikowana w fazie „Możliwości rozwiązania". Opracowana zostaje analiza kosztów i korzyści oraz ocena ryzyka. [6].
- *Zarządzanie implementacją* – ten etap obejmuje nadzór nad implementacją systemu. Monitoruje przebieg implementacji. W tej fazie następuje realizacja projektu architektury korporacyjnej [11].
- *Zarządzanie zmianami architektury* to faza, która zapewnia monitorowanie i zarządzanie procesem zmian. Etap ten zapewnia wprowadzanie szybkich zmian do projektowanej architektury [11].
- *Zarządzanie wymaganiami* produkty w poszczególnych fazach są produktami roboczymi, ściśle określonymi w opracowanych dokumentach, następnie zostają formalnie sprawdzone, uzgodnione oraz podpisane przez zainteresowane strony projektu [11].

2.2. Siatka Zachmana

Siatka Zachmana (ang. Zachman Framework) to dwuwymiarowy schemat, który opisuje przedsiębiorstwo [4]. Jest to narzędzie biznesowe, które można wykorzystać do modelowania istniejących funkcji, elementów i procesów w organizacji. Pomaga również w zarządzaniu zmianami biznesowymi [12]. Siatka Zachmana zapewnia sposób przeglądania systemu z różnych perspektyw i pokazuje powiązania między nimi [9]. Wiersze reprezentują różne perspektywy (widoki) i role przedsiębiorstwa [1]:

- *Zakres, kontekst (Planujący)* określa cel, strategię biznesową i charakter firmy z widoku planistów. Zakres zawiera listę wszystkich ważnych informacji, które zarządzają i wpływają na politykę biznesową przedsiębiorstwa i jej wyniki [3]. Widok ten nazywany jest modelem czarnej skrzynki, ponieważ przedstawia tylko wejścia i wyjścia, a nie wewnętrzne funkcjonowanie [1]. Służy jako kontekst, w którym będzie można zarządzać innymi widokami [2].
- *Pojęcia biznesowe (Właściciel)* opisuje organizację, w której musi funkcjonować system informacyjny [2]. Ten model definiuje interakcję między jednostką, a procesami biznesowymi [3]. Definiuje cele, strategie, strukturę oraz procesy wykorzystywane do wspierania misji przedsiębiorstwa [1].
- *Logika systemu (Analityk*) zawiera wymagania systemowe, obiekty, działania i funkcje, które implementują model biznesowy. Określa, w jaki sposób system powinien wykonywać swoje funkcje. Nazywany jest modelem białego pudełka, ponieważ prezentuje jego wewnętrzne funkcjonowanie [1].
- *Istniejąca technologia (Projektant)* uwzględnia ograniczenia ludzi, narzędzi, technologii i materiałów [1]. Projektant uwzględnia specyfikacje z perspektywy planisty i właścicieli w celu wdrożenia technologii do przetwarzania informacji [3].
- *Komponent, składniki (Budowniczy)* prezentuje poszczególne, niezależne komponenty, które mogą zostać przydzielone wykonawcom do wdrożenia. Wiersz ten jest bardziej szczegółowy od wierszy znajdujących się nad nim [1].
- *Klasy zarządzania* odwzorowuje rzeczywiste procesy w organizacji, które będą odwzorowane w rozważanym systemie operacyjnym [1]. Jest efektem planowania, projektowania i opracowywania w poprzednich etapach. Wytwarza produkt końcowy z perspektywy użytkownika [4].

Siatka Zachmana podkreśla, że każdy widok jest odrębny i ma unikalny charakter i cel. Do każdego poziomu widoku można dostosować odpowiedni poziom szczegółowości. To co jest użyteczne dla właściciela może, nie być przydatne dla projektanta bez względu na to jak bardzo jest szczegółowe. Natomiast właściciel może nie być w stanie zrozumieć modeli projektanta niezależnie od poziomu abstrakcji ponieważ opisują one zupełnie inny widok systemu [2].

Kolumny (ramy) w strukturze siatki Zachmana odnoszą się do pytań zadawanych przez projektującego system przedsiębiorstwa: co, jak, gdzie, kto, kiedy i dlaczego [9]. Ramy koncentrują się na zapewnieniu spójności wszystkich widoków, zapewniając kompletny system niezależnie od kolejności w jakiej zostały utworzone diagramy. Odpowiedzi na powyższe pytania można uzyskać bez zagłębiania się w szczegóły. Odpowiedzi w dużym stopniu zależą od perspektywy [3]. Ramy dla siatki Zachmana:

- *Dane (co)* opisuje podmioty, które są uważane za ważne dla przedsiębiorstwa uwzględniając każdą perspektywę [1]. Odpowiada na pytanie z czego składa się przedsiębiorstwo. Zawiera listę informacji określających zasoby przedsiębiorstwa, mające wpływ na jego kierunek i cel. Opisuje także relację podmiotów, uwzględniając każdą perspektywę przedsiębiorstwa [3]. Przedstawia zawartość systemu lub dane w przypadku systemów informatycznych [2].
- *Funkcja (jak)* określa funkcje, procesy, operacje i czynności, których dotyczy przedsiębiorstwo w odniesieniu do każdej z perspektyw. W tej kolumnie uwzględniane są również wejścia i wyjścia aplikacji [1]. Opisuje w jaki sposób proces jest przekształcany

przyjmując cele przedsiębiorstwa jako dane wejściowe [3].

- *Sieć (gdzie)* pokazuje powiązania między działaniami w przedsiębiorstwie [1]. Każdy z wierszy zawiera połączenia między węzłami przedsiębiorstwa. Węzły mogą obejmować biznes, dostawców usług i połączenia między dostawcami [3].
- *Ludzie (kto)* przedstawia pracowników i jednostki organizacyjne współpracujące z systemem informacyjnym [2]. Reprezentuje informacje do oceny możliwości i wydajności ludzi. Projekt przedsiębiorstwa wiąże się z podziałem pracy i strukturą władzy oraz odpowiedzialności. Dotyczy również interfejsów człowiek-maszyna oraz relacji między pracownikami i wykonywaną przez nich pracą [1]. Odpowiada na pytanie, kto jest za co odpowiedzialny [3].
- *Czas (kiedy)* reprezentuje czas lub relacje zdarzeń, które określają kryteria wydajności. Jest to potrzebne do projektowania harmonogramów, architektury przetwarzania, architektury sterowania i systemów pomiaru czasu [1]. Pionowa oś infrastruktury to oś kontrolna, a oś pozioma to oś czasu [3].
- *Motywacja (dlaczego)* opisuje motywacje pracowników i przedsiębiorstwa. Zawiera cele przedsiębiorstwa, biznesplan, architekturę wiedzy oraz przyczyny podejmowania decyzji oraz podejmowania działań [1]. Określa działania, które ograniczają przedsiębiorstwa do zewnętrznych wymagań organizacji [3]. Reguły ograniczające będą najczęściej stosowane do opisów *Dlaczego i Jak* [2].

Siatka Zachmana ma kształt matrycy 6x6, a każda komórka zawiera zestaw diagramów [2]. Każda komórka w schemacie może mieć dwa wymiary – zakres (szerokość) i poziom szczegółowości (głębokość) [1].

3. Zastosowane metody

Ten rozdział przedstawia metody badawcze, które zostały zastosowane do przeprowadzenia analizy. Analiza bazuje na opracowanej specyfikacji aplikacji do obiegu dokumentów w kancelarii prawnej.

3.1. Specyfikacja funkcjonalna zaprojektowanej aplikacji

Opracowany system powinien obsługiwać następujące grupy użytkowników: adwokaci, aplikanci, sekretariat oraz administrator. Główne wymagania funkcjonalne zostały zrealizowane jako następujące moduły: baza spraw, ewidencja spraw, zasoby, opłaty, ochrona dostępu do aplikacji, pliki, archiwum, korespondencja, kalendarz, pisma oraz kontakty.

3.2. Wyniki badań

W celu przeprowadzenia analizy porównawczej przedstawionej w niniejszym artykule konieczne było określenie odpowiednich kryteriów. Dzięki nim było możliwe wybranie najlepszego szkieletu do projektowania aplikacji korporacyjnych. Zastosowane kryteria porównawcze:

- *Implementacji* w tym kryterium został porównany poziom szczegółowości wykonanych diagramów,
- *Dokumentacyjne* w tym kryterium zostało przeprowadzone porównanie dostępności dokumentacji oraz narzędzi projektowych zastosowanych szkieletów.
- *Testowania* kryterium to opisuje realną możliwość zaprogramowania systemu na podstawie przedstawionych diagramów,
- *Planowania harmonogramu* to kryterium ocenia możliwość ułożenia harmonogramu przy projektowaniu aplikacji, wykorzystując badane szkielety.
- *Czasochłonności* w tym kryterium została zawarta ocena czasu pracy poświęconego na zaprojektowanie aplikacji systemu za pomocą badanych szkieletów.

4. Realizacja procesu badawczego

W celu realizacji analizy porównawczej zostały zaprojektowane diagramy przedstawiające system do obiegu dokumentów w kancelarii prawnej. Zostały użyte szkielety architektury korporacyjnej TOGAF i Zachman.

Projekt zrealizowano przy pomocy siatki Zachmana wykorzystując dziesięć typów diagramów: Package Diagram, Activity Diagram, Class Diagram, Use Case Diagram, Component Diagram, Deployment Diagram, State Diagram, ERD Diagram, Sequence Diagram, Communication Diagram.

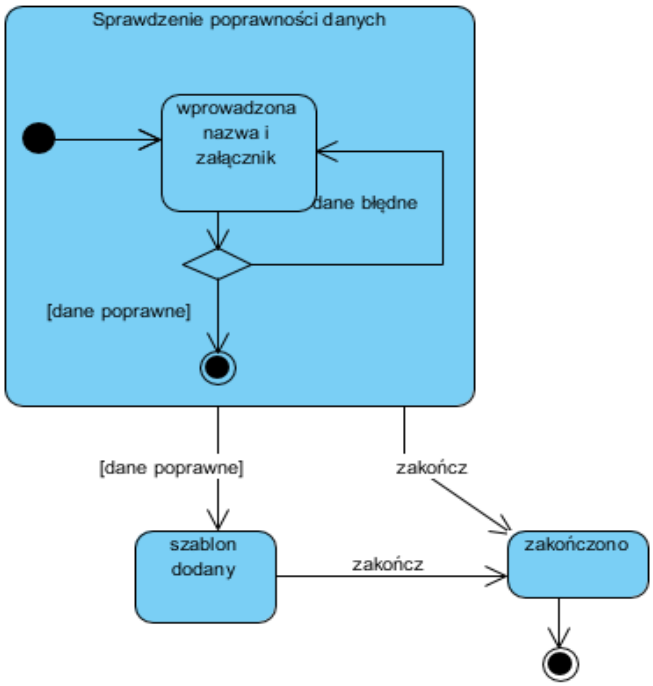
Za pomocą szkieletu TOGAF zaprojektowano czternaście typów diagramów: Solution concept diagram, Functional decomposition diagram, Product lifecycle diagram, Business use-case diagrams, Event diagram, Data entity diagram, Class diagram, Data security diagram, Enterprise manageability diagram, Process/system realization diagram, Software engineering diagram, Environments and locations diagram, Communications engineering diagram, Context diagram.

4.1. Diagramy dla siatki Zachmana

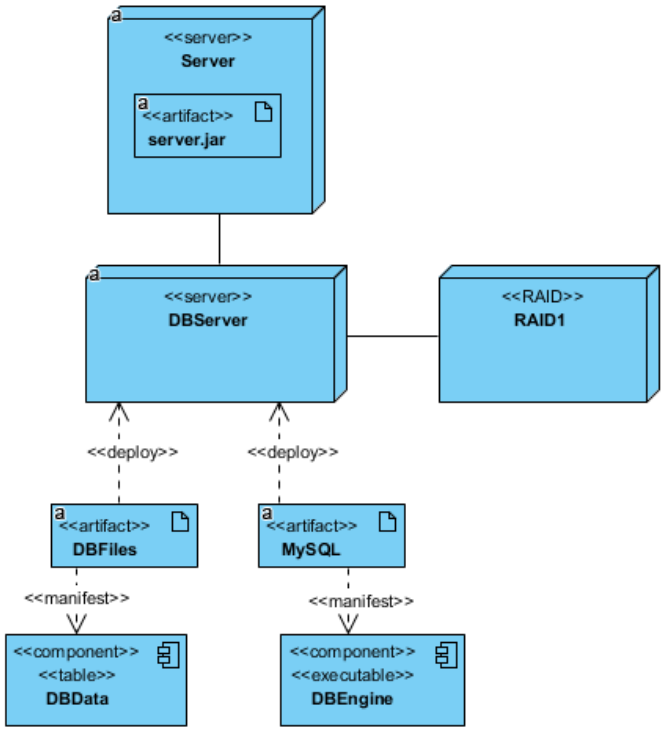
W tej części artykułu zostały przedstawione przykładowe diagramy zrealizowane w oparciu o siatkę Zachmana.

Diagram stanów został zrealizowany dla modułu zawierającego dodawanie pism (rys. 1). Na początku sprawdzana jest poprawność wprowadzonych przez użytkownika danych, jeżeli dane są poprawne to szablon pisma zostaje dodany, w przeciwnym przypadku użytkownik musi ponownie wprowadzić dane.

Na rysunku 2 przedstawiony został diagram wdrożenia, prezentuje on strukturę systemu. Do węzła DBServer połączone są artefakty DBFiles oraz MYSQL. Artefakty te są dodatkowo rozszerzone o komponenty DBData oraz DBEngine. Węzeł DBSerwer jest połączony asocjacją z RAID1 oraz Server. Server ma plik wykonywalny.



Rys. 1. Diagramu stanów dla siatki Zachmana

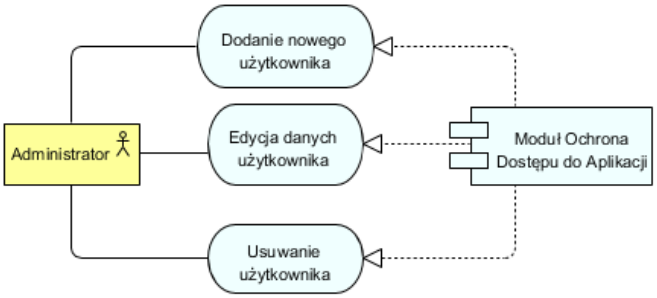


Rys. 2. Diagram wdrożenia dla siatki Zachmana

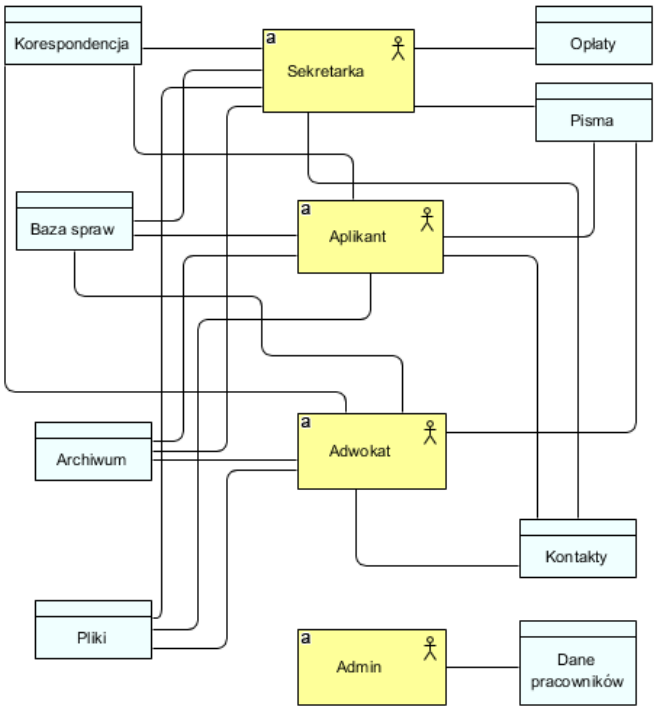
4.2. Diagramy dla szkieletu TOGAF

W tej części zostały zaprezentowane zaprojektowane diagramy zrealizowane za pomocą szkieletu TOGAF.

Na rys. 3 został przedstawiony diagram use case zaprojektowany dla użytkownika administrator, który ma dostęp do modułu Ochrona Dostępu do aplikacji. Zawiera on takie funkcjonalności jak dodanie nowego użytkownika, edycja danych użytkownika oraz usuwanie użytkownika.



Rys. 3. Use-case Diagram dla szkieletu TOGAF



Rys. 4. Security Diagram dla szkieletu TOGAF.

Na diagramie rys. 4 zostały przedstawione najważniejsze dane w projektowanym systemie: dane pracowników, kontakty, pisma, opłaty, korespondencja, baza spraw, archiwum oraz pliki. Zostali do nich przyporządkowani aktorzy.

5. Prezentacja rezulatów badań

W tej części został opisany proces analizy porównawczej, który przeprowadzono na podstawie zaprojektowanych diagramów przy pomocy badanych szkieletów. Wykorzystując szkielet TOGAF zostało wykonanych czternaście typów diagramów. Do zrealizowania projektu za pomocą siatki Zachmana wykorzystano dziesięć rodzajów diagramów.

Zdefiniowano pięć grup kryteriów: implementacji, dokumentacyjne, planowania harmonogramu, testowania oraz czasochłonności. Dla każdego szkieletu, w każdej z grup na podstawie oceny ekspertów, została wystawiona ocena w skali

1-5. Uśrednione oceny zostały zaokrąglone do pełnych wartości. Analiza została przeprowadzona na grupie badawczej składającej się z dwóch ekspertów dziedzinowych.

5.1. Grupa kryteriów implementacji

Grupa kryteriów implementacji obejmuje kwestie związane z implementacją projektowanej aplikacji.

Podczas projektowania przy pomocy szkieletu TOGAF zostało wykorzystanych sześć faz :

- wizja architektury,
- architektura biznesu,
- architektura danych,
- architektura aplikacji,
- architektura technologii,
- możliwości i rozwiązania.

Natomiast przy realizacji aplikacji za pomocą siatki Zachmana zostało użytych pięć perspektyw:

- Zakres, kontekst reprezentuję diagramy od strony planisty,
- Modele biznesowe – diagramy przedstawione z punktu właściciela,
- Logika systemu – przedstawiają diagramu według projektanta,
- Istniejąca technologia – utworzone diagramy ze strony budowniczego,
- Komponent, składniki – diagramy według poddostawcy.

Dodatkowo każda perspektywa została rozpatrzona na podstawie sześciu pytań: co, jak, gdzie, kto, kiedy, dlaczego. Diagramy nie muszą być realizowane według podanej kolejności pytań. W przypadku gdy diagram powtarza się w danej perspektywie w kilku pytaniach można zrealizować go jako jeden.

Na podstawie zaprojektowanych diagramów przy pomocy szkieletu TOGAF zauważono, że przedstawiają one głównie podejście biznesowe do projektowanego systemu. Wyróżnia się w nich aktorów biznesowych, funkcje i procesy biznesowe. Diagramy te skupiają się na tym co ma robić projektowany system, równocześnie nie wchodząc w szczegóły jak ma to być zrealizowane. Dla tego kryterium uzyskano średnią ocenę 3, ponieważ, tak jak zostało wcześniej wspomniane, diagramy nie są szczegółowe.

Dzięki dwuwymiarowemu podziałowi zastosowanemu w siatce Zachmana projektujący w tym narzędziu nie ma możliwości pominięcia któregokolwiek z etapu realizacji projektu. Zrealizowane diagramy zawierają podział na ramy, dzięki temu można dla każdej perspektywy wybrać odpowiedni poziom szczegółowości. Dla tego kryterium średnia ocena to 4, ponieważ nie wszystkie diagramy opisane w dokumentacji zostały wykorzystane.

5.2. Grupa kryteriów dokumentacyjnych

Kolejnym kryterium porównawczym, które zostało zastosowane była grupa kryteriów dokumentacyjnych. Dostępność dokumentacji dla szkieletu TOGAF jest bardzo duża i prosta w znalezieniu. Szkielet ten ciągle się rozwija, wprowadzane są nowe wersje. Jedynym minusem dokumentacji jest to, że łączy ona w sobie część dotyczącą strony biznesowej i informatycznej, przez co czasami trudno jest się w niej odnaleźć. Dostęp do narzędzi darmowy i łatwo dostępny. Projekty można wykonywać m.in. w Modelio lub Visual Paradigm. Średnia ocena uzyskana dla szkieletu TOGAF dla tego kryterium to 4.

Dostęp do dokumentacji dla siatki Zachmana jest bardzo utrudniony. Dokumentacja została znaleziona na stronach dwóch narzędzi (Visual Paradigm oraz Enterprise Architect), które zostały wykorzystane do zrealizowania projektu. Nie jest to jednak dokładna dokumentacja, ale wystarczyła do zrealizowania projektu. Jest niewiele narzędzi do projektowania siatki Zachmana, ale są one darmowe i proste w obsłudze. Dzięki temu użytkownik, który z nich korzysta nie musi poświęcić dużo czasu, aby zrozumieć jak ma je obsługiwać. Brak oficjalnej dokumentacji dla tego szkieletu oraz niewielka dostępność narzędzi spowodowała wystawienie uśrednionej oceny 2.

5.3. Grupa kryteriów testowania

W tabeli 1 zostały umieszczone kryteria, na podstawie których została oceniona grupa kryteriów testowania dla szkieletu TOGAF oraz siatki Zachmana. Podsumowanie zawiera uśrednione wyniki dla poszczególnych szkieletów.

Tabela 1. Grupy kryteriów testowania dla badanych szkieletów

Kryterium	TOGAF	Siatka Zachmana
Spójność diagramów	5	5
Szczegółowość diagramów	3	5
Uwzględnienie aktorów	5	5
Jasno sprecyzowane role aktorów	5	5
Jasno sprecyzowany wygląd ekranów dla poszczególnych funkcjonalności z modułów	4	4
Zastosowana architektura programistyczna	3	3
Zastosowane technologie	2	3
Zastosowane urządzenia	2	1
Scenariusze pracy z poszczególnymi dokumentami w aplikacji	4	4
Czytelny przepływ danych	5	5
Uwzględnienie ochrony danych	5	2
Współpraca między zewnętrznymi aplikacjami	5	2
Jasno sprecyzowane funkcjonalności w module Baza spraw	5	5
Jasno sprecyzowane funkcjonalności w module Ewidencja spraw	5	5
Jasno sprecyzowane funkcjonalności w module Zasoby	5	5
Jasno sprecyzowane funkcjonalności w module Pliki	5	5
Jasno sprecyzowane funkcjonalności w module Opłaty	5	5
Jasno sprecyzowane funkcjonalności w module Archiwum	5	5
Jasno sprecyzowane funkcjonalności	5	5

w module Korespondencja		
Jasno sprecyzowane funkcjonalności w module Kalendarz	5	5
Jasno sprecyzowane funkcjonalności w module Pisma	5	5
Jasno sprecyzowane funkcjonalności w module Kontakty	5	5
Jasno sprecyzowana obsługa ekranów	4	5
Jasno sprecyzowane typy danych w bazie	5	5
Relacje w projekcie bazy danych	5	5
Relacje na diagramie klas	5	5
Jasno sprecyzowana współpraca między modułami	5	5
Podsumowanie	4.5	4.4

Szkielet TOGAF otrzymał w przybliżeniu ocenę 4.5, spełnia on większość kryteriów testowania. Niskie oceny otrzymał w kryteriach: zastosowane technologie i urządzenia, ponieważ nie zostały one jasno sprecyzowane na diagramach.

Natomiast siatka Zachmana otrzymała niewiele mniejszą ocenę 4.4, najniższą ocenę otrzymała w kryterium zastosowane urządzenia.

5.4. Grupa kryteriów planowania harmonogramu

W tabeli 2 zostały przedstawione dwa kryteria do planowania harmonogramu oraz ich uśrednione oceny. Kryteria te to:

- Kolejność wykonywanych zadań – określa czy jest możliwość określenia kolejności wykonywanych zadań na podstawie zaprojektowanych diagramów.
- Zdefiniowanie technologii – określa to czy na diagramach są zdefiniowane potrzebne technologie, które powinny zostać użyte przy realizacji wcześniej określonych zadań.

Tabela 2. Grupy kryteriów harmonogramu dla badanych szkieletów

Kryterium	TOGAF	Siatka Zachmana
Kolejność wykonywanych zadań	4	4
Zdefiniowanie technologii	2	4
Podsumowanie	**3**	**4**

Szkielet TOGAF w tym kryterium uzyskał ocenę 3, ponieważ szkielet można rozdzielić na zadania, a diagramy nie prezentują technologii potrzebnych do zrealizowania projektu. Siatka Zachmana otrzymała ocenę 4, ponieważ można łatwo podzielić projekt na określone zadania oraz na diagramach są zdefiniowane technologie potrzebne do zrealizowania projektu.

5.5. Grupa kryteriów czasochłonności

Grupa kryteriów czasochłonności została wyznaczona na podstawie czasu, jaki eksperci poświęcili na realizacje pełnego procesu projektowania z uwzględnieniem określonych wcześniej założeń (wymagania funkcjonalne, wybrane diagramy).

Tabela 3. Grupa kryterium czasochłonności dla badanych szkieletów

Rodzaj oceny	Szkielet TOGAF	Siatka Zachmana
Uśredniony czas	**17h**	**14h**

Tabela 3 przedstawia uśredniony czasy dla oceny eksperckiej. Liczba godzin które były potrzebne do zaprojektowania diagramów za pomocą szkieletu TOGAF wynosi w przybliżeniu siedemnaście godzin. Ocena to 4, ponieważ do zrealizowania projektu za pomocą tego szkieletu była potrzebna większa liczba diagramów do zaprojektowania.

Na wykonanie projektu za pomocą siatki Zachmana potrzeba było około czternastu godzin. Czas jest krótszy niż dla szkieletu TOGAF ponieważ część diagramów została pominięta. Ocena przyznana w tym kryterium dla siatki Zachmana to 5, ponieważ projektujący nie musi korzystać ze wszystkich diagramów.

6. Dyskusja wyników i wnioski

W przeprowadzonej analizie porównawczej zostały osiągnięte początkowe założenia. W części teoretycznej zapoznano się z dokumentacją poszczególnych szkieletów. Następnie opracowano specyfikację aplikacji do obiegu dokumentów w kancelarii prawnej. Na jej podstawie zostały zaprojektowane diagramy za pomocą szkieletu TOGAF oraz siatki Zachmana. W celu wykonania analizy porównawczej zostały zastosowane następujące grupy kryteriów porównawczych: implementacji, dokumentacyjne, testowania, planowania harmonogramu oraz czasochłonności.

Porównując oba szkielety w grupie kryteriów implementacyjnych wywnioskowano, że lepsza była siatka Zachmana. Diagramy zrealizowane na jej podstawie przedstawiają podejście do systemu z różnych perspektyw: klienta, projektanta, programisty. Natomiast w szkielecie TOGAF jest przedstawione głównie podejście, które przedstawia funkcjonalności, jakie ma wykonywać projektowana aplikacja.

W zastosowanej grupie kryteriów dokumentacyjnych zauważono, że dokumentacja dla szkieletu TOGAF ciągle się rozwija i jest ogólnodostępna, a narzędzia do projektowania są darmowe i intuicyjne w użyciu. Dokumentacja dla siatki Zachmana jest mała i trudno ją znaleźć. Wspomniana dokumentacja nie jest rozwijana, natomiast narzędzi do projektowania jest niewiele jednak są one darmowe i łatwe w obsłudze.

W grupie kryteriów testowania zostało zastosowanych dwadzieścia sześć kryteriów. Szkielet TOGAF okazał się w niewielkim stopniu lepszy w tym kryterium, spełniał on większość kryteriów. Minusem było to, że projektowane diagramy nie zawierały dokładnego opisu technologii, natomiast opis urządzeń był za mało rozwinięty. Siatka Zachmana otrzymała niższą ocenę, ponieważ nie zostały w niej jasno sprecyzowane użyte urządzenia.

Grupa kryteriów planowania harmonogramu badało czy w szkieletach jest jasno określona kolejność wykonywanych

zadań oraz czy na diagramach zostały jasno zdefiniowane technologie. W tym kryterium lepsza okazała się siatka Zachmana, ponieważ zostały w niej jasno przedstawione technologie, które będą użyte przy implementacji systemu.

Ostatnią grupą kryteriów, które została zastosowana w przedstawionej analizie porównawczej była czasochłonności. Na podstawie obliczonego uśrednionego czasu widać, że więcej czasu poświęcono na zaprojektowanie diagramów przy użyciu szkieletu TOGAF. Różnica pomiędzy wykonaniem diagramów za pomocą szkieletu TOGAF i siatki Zachmana wynosiła 3 godziny. Wpływ na to miała ilość i dokładność projektowanych diagramów.

Na podstawie przeprowadzonej analizy porównawczej, szkielet TOGAF uzyskał ocenę 3,7 natomiast siatka Zachmana 3,88. Wynika z tego, że siatka Zachmana w niewielkim stopniu, jest lepsza do projektowania architektury aplikacji korporacyjnej. Postawiona na początku artykułu hipoteza została potwierdzona. Oba szkielety do projektowania architektury aplikacji korporacyjnych są porównywalne.

Literatura

[1] Abbas W. F., Ismail S. H., Haron H., Hariri W. N.: Enterprise Integration of Employee Onboarding Process Using Zachman Framework, International Journal of Engineering & Technology, 7(4.31), 2018, s. 46-51.

[2] De Villiers D. J.: Using the Zachman Framework to assess the rational unified process. The Rational Edge, 2001.

[3] Ertaul L., Sudarsanam R.: Security Planning Using Zachman Framework for Enterprises, 2005.

[4] Ertaul L., Vandana S., Gulati K., Saldamli G.: Enterprise Security Planning using the Zachman Framework-Builder's Perspective. In Proceedings of the International Conference on Security and Management (SAM). The Steering Committee of The World Congress in Computer Science, Computer Engineering and Applied Computing (WorldComp), 2011.

[5] Goethals F.: An overview of enterprise architecture framework deliverables. 2005.

[6] Governica, hasło: TOGAF, https://www.governica.com/TOGAF, (09.03.2019).

[7] Josey A.: TOGAF Version 9.1-A Pocket Guide. Van Haren, 2016

[8] Kotusev S.: The TOGAF-Based Enterprise Architecture Practice: An Exploratory Case Study. Communications of the Association for Information Systems, 2018, 43(1), 20.

[9] Sowa J. F., Zachman J. A.: Extending and formalizing the framework for information systems architecture.IBM systems journal, 1992

[10] Tao Z. G., Luo Y. F., Chen C. X., Wang M. Z., Ni F.: Enterprise application architecture development based on DoDAF and TOGAF.Enterprise Information Systems, 2017,11(5), s. 627-651.

[11] TOGAF Version 9.1 Enterprise Architecture, Module I Management Overview, http://www.togaf.info/togafSlides91/TOGAF-V91-M1-Management-Overview.pdf, (03.01.2019)

[12] Visual Paradigm, hasło: „Zachman", https://www.visual-paradigm.com/guide/enterprise-architecture/what-is-zachman-framework/, (20.03.2019).

[13] Visual Paradigm, hasło: The Best TOGAF Software, https://www.visual-paradigm.com/guide/togaf/best-togaf-software/, (09.03.2019).

[14] Yaqin M. A., Prayoga F. A., Ihsan A. N., Pulungan F.: Arsitektur Enterprise pada Permainan Hay Day Menggunakan Metode Zachman Framework. Prosiding SENIATI, 5(3), 2019, s. 50-58.

JOURNAL of COMPUTER SCIENCES INSTITUTE

JCSI 13 (2019) 323-331
Wysłane: 2019-09-07
Przyjęte: 2019-09-17

Ekstrakcja parametrów z próbek danych biometrycznych

Paweł Danek*, Krzysztof Ćwirta, Piotr Kopniak
Politechnika Lubelska, Instytut Informatyki, Nadbystrzycka 36B, 20-618 Lublin, Polska

Streszczenie. W artykule opisano możliwe sposoby ekstrakcji parametrów z próbek danych biometrycznych, takich jak odcisk palca czy nagranie głosu. Zweryfikowano wpływ konkretnych sposobów obróbki na skuteczność algorytmów obróbki próbek biometrycznych oraz ich porównania. Wykonano badania polegające na przetworzeniu dużej liczby próbek z użyciem wybranych algorytmów. W przypadku odcisku palca wykorzystano normalizację obrazu, filtr Gabora i porównanie z użyciem deskryptorów. Dla autoryzacji głosowej analizowano algorytmy LPC i MFCC. W przypadku obu rodzajów autoryzacji uzyskano zadowalającą skuteczność rzędu 60-80%.

Słowa kluczowe: biomteria; odcisk; głos; autoryzacja; normalizacja; gabor; deskryptor; lpc; mfcc

*Autor do korespondencji.

Adresy e-mail: pawel.danek@pollub.edu.pl, krzysztof.cwirta@pollub.edu.pl, p.kopniak@pollub.pl

Extraction of parameters from biometric data samples

Paweł Danek*, Krzysztof Ćwirta*, Piotr Kopniak
Institute of Computer Science, Lublin University of Technology, Nadbystrzycka 36B, 20-618 Lublin, Poland

Abstract. This article describes possible ways to extract parameters from biometric data samples, such as fingerprint or voice recording. Influence of particular approaches to biometric sample preparation and comparision algorithms accuracy was verified. Experiment involving processing big ammount of samples with usage of particular algorithms was performed. In fingerprint detection case the image normalization, Gabor filtering and comparision method based on descriptors were used. For voice authorization LPC and MFCC alghoritms were used. In both cases satisfying accuracy (60-80%) was the result of the surveys.

Keywords: biometrics; fingerprint; voice; authorization; normalization; gabor; descriptor; lpc; mfcc

*Corresponding author.

E-mail addresses: pawel.danek@pollub.edu.pl, krzysztof.cwirta@pollub.edu.pl p.kopniak@pollub.pl

1. Wstęp

W ramach pracy przeanalizowano działanie algorytmów autoryzacji biometrycznej i ekstrakcji danych z próbek pod kątem odcisku palca oraz głosu. Nie ulega wątpliwości fakt, że w dzisiejszych czasach spora część codziennego życia przenosi się do Internetu. Codziennością jest kontakt ze znajomymi czy współpracownikami z wykorzystaniem *mediów społecznościowych*, robienie zakupów *on-line*, praca czy rozrywka z wykorzystaniem sieci. W tym, ogólnodostępnym medium znajduje się coraz więcej danych dotyczących każdego człowieka oraz dostęp do wielu, wrażliwych funkcjonalności - takich jak np. bankowość *on-line*. W dobie miniaturyzacji oraz przenoszenia tych funkcjonalności na ekrany *smartfonów* - coraz większy nacisk kładzie się na zabezpieczenia.

Jednym z najdokładniejszych i najtrudniejszych do złamania mechanizmów zabezpieczeń jest obecnie biometryka. Dzieje się tak dlatego, że ciało człowieka, jego budowa i cechy charakterystyczne dają świetne modele autoryzacyjne, które dla każdego człowieka są unikalne i niepowtarzalne. Nie da się ich *pożyczyć* lub *ukraść*, a sklonowanie ich jest dużo trudniejsze niż odgadnięcie hasła do serwisu. Każda próbka danych biometrycznych charakteryzuje się swoją budową i listą cech, które można z nich ekstrahować. Proces wyodrębniania poszczególnych parametrów jest jednym z najważniejszych, które budują skuteczne systemy autoryzacji biometrycznej.

W celu analizy sposobów obróbki próbek stosowanych w takich mechanizmach rozpoznawania użytkownika wybrano dwa, najbardziej popularne ze sposobów – autoryzacja z wykorzystaniem odcisku palca i głosu. W ramach badań porównano po dwa różne sposoby ekstrakcji próbek – z wykorzystaniem detektora Harrisa (wykrywającego duże zmiany częstotliwości) oraz algorytmu wykrywającego minucje na podstawie liczby przecięć linii papilarnych.. W przypadku weryfikacji odcisku palca – wykorzystano mechanizm porównania oparty o *deskryptory obrazu* – natywny mechanizm bibliotek do przetwarzania obrazu, z użyciem któregoś można zweryfikować podobieństwo dwóch grafik. Wejściem dla deskryptora jest lista punktów kluczowych, na podstawie której zbudowany będzie deskryptor. Od określenia źródła tych punktów zależy skuteczność i szybkość działania algorytmu – w jednym algorytmie wykorzystano punkty wykryte przez detektor Harrisa, natomiast w drugim wyznaczono *minucje* – punkty charakterystyczne odcisku. W ramach badań wykorzystano bazę odcisków międzynarodowego konkursu FVC (publicznie dostępną w Internecie). Skrypt, napisany w języku Python, kolejno porównywał odciski na zasadzie *każdy z każdym* z wykorzystaniem obu algorytmów. Każdy z przypadków miał swój oczekiwany rezultat, na bazie którego generowano

końcowy wynik. Z wykorzystaniem tych danych obliczono procentowe skuteczności algorytmów (ogólne, przypadku pozytywnej autoryzacji i pozytywnej odmowy dostępu).

W przypadku autoryzacji z wykorzystaniem próbki głosowej porównano skuteczność dwóch algorytmów ekstrakcji próbek – MFCC oraz LPC. Algorytm trenował się z wykorzystaniem *n* próbek dla każdego mówcy – w tym procesie wyznaczał oba rodzaje cech dla każdej z próbek treningowych mówcy. Na tej podstawie powstała książka kodowa. Następnie – dla *m* innych próbek każdego z mówców wyszukiwano największy stopień podobieństwa (najmniejszy dystans) do próbek zapisanych w książkach kodowych. Jeśli nazwy mówców się zgadzały – porównanie zaliczano jako udaną próbę. Na podstawie tych wyników określono procentową skuteczność autoryzacji z wykorzystaniem obu sposobów ekstrakcji cech z próbek.

2. Autoryzacja z wykorzystaniem odcisku palca

Jednym z najpopularniejszych rozwiązań w temacie biometryki jest wykorzystanie linii papilarnych. Obróbka skanu odcisku palca, choć jest procesem bardzo skomplikowanym, to na wysokim poziomie abstrakcji składa się z kilku prostych kroków. Wśród nich można wyróżnić [1]:

- przygotowanie obrazu
- uwydatnienie linii
- odchudzenie linii
- ekstrakcję cech
- porównanie modeli

Każdy z tych punktów realizowany jest z użyciem różnych, charakterystycznych dla swojego przebiegu algorytmów. Największą różnorodność wśród zastosowanych rozwiązań można spotkać w pierwszym z punktów. Spotyka się on bowiem z najtrudniejszym zadaniem. Skany odcisków mogą pochodzić z różnych skanerów (niekoniecznie optycznych) lub nawet być wygenerowane przez odpowiednie oprogramowanie. Skutkuje to dużą rozbieżnością w jakości analizowanych próbek - różne mogą być nie tylko takie parametry jak rozdzielczość obrazów, ale także szczegółowe informacje - na przykład jasność pikseli w miejscu nacisku lub pustym polu.

Z tego powodu algorytmy wykorzystane przy przygotowaniu odcisku są często kluczem do stworzenia dobrej jakości oprogramowania. Celem tego punktu jest obróbka obrazu w takich sposób, aby otrzymać wyraźny obraz, bez szumów, z odpowiednim kontrastem pomiędzy liniami papilarnymi a pustymi polami. Jednym z ciekawych i skutecznych rozwiązań do osiągnięcia tego celu jest wykorzystanie mechanizmu *Contrast Limited Adaptive Histogram Equalization* - w skrócie *CLAHE* [2-3]. Jest to zoptymalizowany proces normalizacji histogramu. Jego założeniem jest analiza natężenia jasności punktów obrazu oraz wyrównanie i *wygładzenie* dużych skoków w histogramie. Efektem tego procesu jest eliminacja prześwietleń obrazu (zmniejszenie liczby zbyt jasnych punktów) oraz uwydatnienie kontrastu (poprzez wzmocnienie ciemnych obszarów). W przypadku obrazów, w których kontrast nie jest *globalny* - czyli nie powinien być zbliżony dla każdego regionu obrazu, zaczęto wykorzystywać mechanizmy adaptacyjnej normalizacji – *AHE* [4]. Zakładają one podział obrazu na bloki o określony rozmiarze oraz uwydatnieniu kontrastu w danym regionie. Jednak nie zawsze skutkuje to oczekiwanym rezultatem - małe bloki składające się z samych jasnych punktów mogą być po takim procesie prześwietlone, a bloki zawierające szum - wzmocnione. Stąd mechanizm *CLAHE*, którego przedrostek - *Contrast Limited* - zapewnia redukcję punktów, które przekraczają zadany limit kontrastu. Są one przesuwane do innych miejsc histogramu. Dopiero po tym procesie - w każdym bloku przeprowadzana jest normalizacja histogramu [2].

Innym, prostszym podejściem jest wykorzystanie progu jasności [5]. Zakładając, że przetwarzany jest obraz w skali szarości analizie poddawana jest jasność z zakresu od 0 do 255. Punktem zainteresowania są najciemniejsze piksele - czyli miejsca, w których przyciśnięta została linia papilarna. Warto wyznaczyć zakres analizy - czyli odrzucić wszystkie punkty o jasności wyższej niż - przykładowo - 30. Następnie należy poznać najczęściej występujące wartości jasności w tym zakresie - idealnie sprawdzi się tutaj mediana. Po jej poznaniu oraz wykorzystaniu odpowiedniej tolerancji - 5% lub 10% wartości - wszystkie punkty niższe od określonej wartości uznaje się za fragment linii papilarnej. Wszystkim punktom odcisku przypisuje się minimalną wartość jasności - 0, natomiast wszystkie inne powinny być tłem - czyli otrzymać maksymalną wartość 255. To podejście sprawdzi się dobrze w przypadku dobrej jakości skanów. Podobny algorytm można zastosować w oparciu o odchylenie standardowe parametru jasności.

W przypadku uwydatnienia linii odcisku najczęściej wykorzystywany jest filtr Gabora [5,6]. Proces zaproponowany przez Dennis'a Gabor'a [7] znalazł szerokie zastosowanie w medycznym przetwarzaniu obrazu [8]. Wykorzystywany jest przy obróbce obrazów uzyskanych z aparatury medycznej. Jest on również stosowany w większości projektów zajmujących się autoryzacją z wykorzystaniem odcisku palca. Głównym założeniem wykorzystania tego filtru jest ekstrakcja fragmentów obrazu o konkretnej częstotliwości, skierowanych w konkretnym kierunku [7]. Takie fragmenty obrazu oznaczone będą wysoką wartością jasności. Do zastosowania tego filtra trzeba znać kierunki linii papilarnych na obrazie oraz częstotliwości tych fragmentów. Do określenia kierunków linii można skorzystać z gradientów funkcji Gaussa (których wartości narastają odpowiednio z położeniem w danym kierunku) lub transformat Sobela w oparciu o matematyczne metody wyznaczania kierunku wektorów oparte o *arcus tangens* [9]. Częstotliwość bloków obrazu należy rozumieć jako ilość wysokich zmian poziomów jasności na konkretnym obszarze. Można liczyć ją poprzez analizę każdego z pikseli lub ponownie wykorzystać wyliczenie odchylenia standardowego. W większości jednak będą to wartości na tyle zbliżone, że można również przyjąć ich średnią za wystarczającą. Znając częstotliwość obrazu oraz orientację każdego z małych bloków obrazu można przystąpić do wyliczenia wartości filtra Gabora. Te z kolei wylicza się z użyciem wzoru matematycznego (1), według którego oryginalnie zdefiniowany został ten filtr [7].

$$g(x,y;\lambda,\theta,\phi,\gamma)=\exp\left(-\left(\frac{x'^2+\gamma^2 y'^2}{2\sigma^2}\right)\right)\cos\left(2\pi\frac{x'}{\lambda}+\phi\right) \quad (1)$$

gdzie [7]: λ - długość fali, θ - kąt fali, ϕ - przesunięcie falowe, γ - współczynnik proporcjonalności, σ- odchylenie standardowe obwiedni fali.

Na każdy z bloków obrazu należy nakładać jego odpowiednią wersję [5,6] - zgodną z kierunkiem linii w danym bloku. Efektem wykonania tego procesu będzie skan odcisku z *uwydatnionymi* liniami papilarnymi. Efekt tego procesu widoczny jest na rysunku [1]. Dzięki temu procesowi są one proste i jednolite - znikają poszarpania oraz puste pola, które pozostawiają po sobie pory w skórze. Wykonanie tego procesu jest jednym z najważniejszych kroków prowadzących do prawidłowego wyznaczenia cech charakterystycznych odcisku.

Rys. 1. Skan odcisku po zastosowaniu filtra Gabora [10]

Kolejnym z ważnych procesów przygotowujących obraz do wykrywania punktów charakterystycznych odcisku jest *odchudzanie* odcisku palca [6]. Proces ten polega na cieniowaniu pogrubionych linii [11] - do momentu kiedy ich grubość będzie możliwie najniższa. Jednym z rozwiązań tego problemu jest wykorzystanie jąder przekształceń, których nałożenie na obraz cieniuje jego krawędzie [11]. Każdy blok obrazu (o rozmiarach równych macierzy) jest przekształcony zgodnie z wagą każdego z pikseli jądra. W celu wykonania procesu cieniowania odpowiednio są to jądra zwiększające wagi zewnętrznych krawędzi linii. Przykładowo, taka macierz [6] zmniejsza jasność dolnych krawędzi bloku.

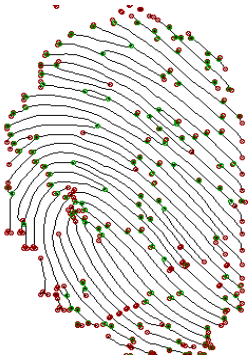
$$\begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 0 \\ 0.1 & 0.1 & 0.1 \end{bmatrix} \tag{2}$$

Po zdefiniowaniu odpowiedniej liczby operatorów należy wykorzystywać je jako jądro przekształcenia do momentu, aż wykonanie przekształceń przestanie dawać jakiekolwiek efekty [6]. Wynikiem przekształcenia obrazu poprzez operatory *cieniowania* jest skan odcisku, w którym każda linia ma grubość rzędu jednego piksela, jak widać na rysunku [2].

Dzieje się tak ze względu na rozmiar analizowanego obszaru – w przypadku ograniczenia się do najprostszych typów (przy założeniu że oczekiwana grubość linii to 1 piksel) wystarczy analiza drobnych bloków o rozmiarze 3x3. Takie podejście jest znane jako *Crossing number concept* i powszechnie stosowane w algorytmach autoryzacji biometrycznej z użyciem odcisku palca [6,14].

Rys. 2. Skan odcisku po procesie cieniowania [10]

Jeśli w takim obszarze znajdują się co najmniej dwa ciemne punkty obok siebie, których najbliższy sąsiad jest jasnym punktem - jest to zakończenie lub rozpoczęcie linii. Jeśli w tym samym obszarze znajduje się punkt, który ma dwóch, bezpośrednich sąsiadów, to jest to rozwidlenie linii [6,14]. Już przy tak prostym algorytmie, analizując kolejne bloki obrazów, można wykryć bardzo dużą liczbę punktów kluczowych w pojedynczym skanie odcisku. Ich położenie (w obrazie oraz względem siebie) oraz rodzaj są kluczowe dla algorytmów porównawczych. Efekt działania tego algorytmu widać na rysunku [3].



Rys. 3. Zakończenia linii oraz rozwidlenia wykryte w skanie odcisku [10]

Ostatnim i najważniejszym etapem procesu przetwarzania odcisków jest porównanie dwóch zestawów minucji względem siebie. Jednym z ciekawych sposobów porównania dwóch odcisków jest wykorzystanie deskryptorów obrazów [5] oraz ich komparatorów. Mechanizmy (takie jak *ORB - Oriented Rotated Brief* czy *bruteforce matcher*) są wbudowane w popularne biblioteki (np. *OpenCV*) służące do przetwarzania obrazu. Posiadają one implementacje w wielu językach programowania. Deskryptor budowany jest na podstawie punktów kluczowych obrazu (którymi mogą być minucje oraz ich okolice), a ich porównanie odbywa się na zasadzie odnalezienia *trafień*, czyli korespondujących punktów w drugim obrazie. Pomiędzy korespondującymi punktami kluczowymi wyznaczona jest odległość - im mniejsza, tym trafienie jest dokładniejsze [3]. Dzięki wykorzystaniu takiego mechanizmu - procesy zaimplementowane w popularnych bibliotekach automatycznie odnajdą korespondujące układy współrzędnych dla dwóch odcisków. Wśród innych, ciekawych i skuteczniejszych rozwiązań prym wiodą te, które analizują kierunek oraz typ minucji [13]. Jednym z ciekawszych podejść jest wstępne obrócenie obu

skanów (tak aby znajdowały się w jednym układzie odniesienia), odnalezienie minucji, po czym odnalezienie korespondujących punktów poprzez założenie że dzieli je najmniejsza odległość [15]. Po wydobyciu takich informacji należy porównać typy minucji oraz ich kierunki. Ostatnim z ciekawych i skutecznych rozwiązań jest ekstrakcja z obrazu wszystkich fragmentów poza okolicami minucji [13]. Po wykonywaniu obróceń i przesunięć obu obrazów, nakładając je na siebie, w pewnym punkcie powinno otrzymać się dużą liczbę pokryć. Świadczy to o tym, że dane odciski są zbieżne. Wszystkie z opisanych powyżej rozwiązań spotykane są w amatorskich projektach zajmujących się autoryzacją biometryczną z użyciem odcisku palca.

3. Badania skuteczności

W celu zbadania skuteczności algorytmów przygotowano skrypt wykorzystujący projekty oparte o wykrywanie punktów kluczowych na bazie detektora Harrisa oraz ręczne wykrywanie minucji. Zbiór punktów wykorzystywano do zbudowania deskryptora obrazu. Następnie, z wykorzystaniem rodzimych mechanizmów biblioteki *OpenCV* porównano podobieństwo obu obrazów. Badanie skuteczności wykonano w oparciu o publicznie dostępną bazę skanów ogólnoświatowego konkursu FVC z 2004 roku [10]. Zbiór plików – w strukturze dziesięć różnych odcisków, po osiem próbek każdy – porównano na zasadzie *każdy z każdym*. W przypadku porównania różnych próbek tego samego odcisku – oczekiwano poprawnej autoryzacji. Jeśli wykorzystane zostały skany innych odcisków – algorytm powinien odmówić dostępu. Jako parametry wyjściowe określono procentową skuteczność autoryzacji (jako procentowy stosunek liczby udanych autoryzacji do liczby prób, w których porównano skany tego samego odcisku) oraz procentową skuteczność odmowy dostępu (analogicznie do poprzedniego przypadku – procent przypadków w których uzyskano poprawny wynik negatywny). Obliczenia wykonano na maszynie VPS. Sumarycznie przypadków porównania było nieco ponad 3100, z czego zdecydowana większość to przypadki odmowy dostępu. W tabeli wynikowej projekt działający na bazie detektorów Harrisa oznaczony został jako *A*, natomiast implementacja z wykrywaniem minucji jako *B*.

Tabela 1. Wynik badania skuteczności algorytmu porównującego odciski palców

Parametr	Projekt A	Projekt B
Ogólna skuteczność procentowa	86%	87%
Skuteczność przypadków autoryzacji	44%	60%
Skuteczność przypadków odmowy dostępu	90%	90%
Średni czas porównania	13s	57s

Jako próg referencji, przy dostosowywaniu minimalnego progu autoryzacji, wykorzystano utrzymanie skuteczności odmowy dostępu na poziomie 90%. Po dostosowaniu algorytmów do tego parametru – oba mają wysoką ogólną skuteczność procentową (procent wszystkich przypadków, w których oczekiwany wynik zgadzał się z faktycznym). Wysoka wartość tego wyniku spowodowana jest faktem, że w całym badaniu przypadków odmowy dostępu było dużo więcej niż przypadków autoryzacji. Dużo bardziej interesującym parametrem jest procentowa skuteczność w przypadku porównania dwóch, tych samych odcisków. Projekt definiujący punkty kluczowe jako wynik działania detektora Harrisa autoryzował poprawnie tylko 44% przypadków. Wynik jest bardzo niski i rozwiązanie zdecydowanie nie nadaje się na produkcyjne rozwiązanie. Duży wzrost – do 60% - odnotowano w przypadku zdefiniowania punktów kluczowych deskryptora jako minucji odcisku. Pomimo sporego skoku – procent autoryzacji jest wciąż niski. Winnym jest sposób porównania – deskryptory obrazu, chociaż jest to bardzo kreatywne rozwiązania, nie dają dobrej skuteczności w przypadku porównania tak szczegółowych *obrazów*. W celu poprawienia tego wyniku warto zmienić sposób porównania danych otrzymanych na podstawie obróbki skanu. Wśród najpopularniejszych rozwiązań często definiuje się sposób oparty o odnalezienie najbliższej, sąsiadującej minucji (wraz z zapamiętaniem ich typów) czy założenie, że w odpowiednio obróconych skanach odcisków odpowiadające sobie minucje, to te które dzieli najmniejsza odległość. Sposoby te wymieniono w poprzednim rozdziale.

W badaniach określono także średni czas porównania dwóch skanów odcisków. W projekcie B, wykrywającym minucje, widać duży (blisko pięciokrotny) wzrost czasu wykonania. Jest to spowodowane procesem odchudzania odcisku, który wykonuje się kilka razy dla każdego ze skanów (aż przestanie dawać jakikolwiek efekt). Warto brać pod uwagę jakość kodu przy implementacji takich projektów, ponieważ nieoptymalne zaprojektowanie nawet jednego, prostego procesu może przynieść duże straty w czasie wykonania algorytmu. Odchudzanie obrazu, chociaż konieczne do wykrycia minucji, znacznie spowolniło pojedyncze wykonanie. Konsekwencją zysku skuteczności był duży wzrost czasu wykonania.

Ogólna skuteczność procentowa rzędu 80% to zadowalający wynik, należy pamiętać jednak, że zdecydowana większość przypadków testowych to przypadki odmowy dostępu (jest dużo więcej danych do porównania). Przy utrzymaniu tego współczynnika na poziomie 90% udało się uzyskać maksymalnie 60% skuteczności w przypadku prawidłowej autoryzacji. Wynik – jak na prosty projekt opublikowany w Internecie – jest zadowalający, jednak dla komercyjnego rozwiązania z pewnością potrzebna byłaby dużo większa skuteczność. Kolejnym etapem badań w tym kierunku oraz sposobem na ulepszenie algorytmu powinna być modyfikacja sposobu porównania. Wykorzystanie prostego mechanizmu na bazie deskryptorów obrazów jest bardzo czytelne i optymalne, jednak nie daje stuprocentowej skuteczności. Przegląd literatury dowiódł, że w tej dziedzinie częściej stosowane są inne rozwiązania, które opisano we wcześniejszych fragmentach artykułu.

4. Rozpoznanie mówcy

Innym, coraz popularniejszym sposobem autoryzacji z wykorzystaniem cech budowy organizmu

człowieka jest rozpoznanie osoby mówiącej - biometria głosowa. Metoda automatycznego rozpoznawania mówcy bazuje przede wszystkim na wykorzystaniu cech behawioralnych mówcy, które z biegiem lat zostały nabyte przez mówcę (zostały świadomie wyuczone lub nabyte nieświadomie) [16].

Główną zasadą obowiązująca w biometryce jest obowiązek, aby charakterystyka behawioralna była unikatowa i uniwersalna, trwała i prosta do ilościowej oceny- tylko w takim przypadku wspomniana charakterystyka będzie zapewniała wymagany stopień złożoności konieczny do uwidocznienia wymaganych zróżnicowań cech indywidualnych mówcy [16].

Głos każdej osoby jest odmienny, ponieważ głos jest zależny od fundamentalnych cech fizycznych, które mają ogromny wpływ na rodzaj powstałych sygnałów dźwiękowych, generowanych przez człowieka podczas mówienia [17].

Każdy mówca ma wpływ na wiele czynników, które z kolei stanowią podstawę do wygenerowania głosu. Są to m.in atrybuty [17]:

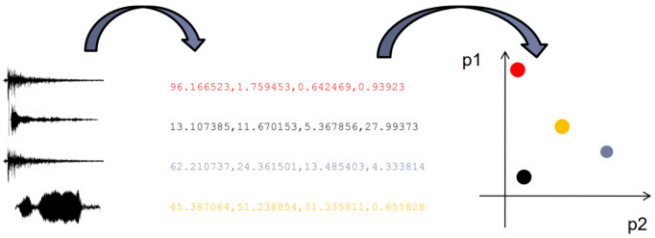
- częstotliwość podstawowa (zależy m.in od długości kanału głosowego) – zwana wysokością tonu. Dla mężczyzn zakres częstotliwości podstawowej wynosi 100-120Hz, dla kobiet 200-250Hz [18],
- dźwięk nosowy – dźwięk mowy, w którym strumień powietrza przechodzi przez nos wskutek obniżenia podniebienia miękkiego z tyłu jamy ustnej. Podczas wymawiania spółgłosek nosowych, usta są w pewnym momencie zamknięte (przez wargi lub język), a strumień powietrza jest całkowicie wydalany przez nos [19],
- kadencja (ton opadający) – następstwo dźwięków nadające frazie charakter zakończenia [20],
- przegięcie – sposób mówienia, w którym głośność lub wysokość tonu jest modyfikowana [21].

Poziom skuteczności systemu, który zajmuje się rozpoznawaniem mówcy jest podporządkowany temu, w jakim stopniu badane parametry fizyczne sygnału mowy odpowiadały będą za przenoszenie cech osobniczych mówcy [22].

W celu rozpoznania osoby, w biometryce głosowej wykorzystywane są następujące parametry i wielkości [22]:

- parametry wyznaczone bezpośrednio z przebiegu czasowego: względne długości czasu wypowiedzi poszczególnych elementów fonetycznych; obwiednia czasowa amplitudy dźwięku; parametry analizy przejść przez zero sygnału mowy; rozkład interwałów czasowych;
- parametry wyznaczone z widma sygnału mowy: uśrednione widmo amplitudowe; widmo mocy; częstotliwość podstawowa tonu krtaniowego; częstotliwości, stosunki amplitudowe oraz szerokości pasm formantów; widmo krótkoterminowe; momenty widmowe
- parametry liniowego kodowania predykcyjnego;
- inne, jak np. charakterystyki prozodyczne, parametry cepstralne.

Proces parametryzacji jest jednym z podstawowych etapów rozpoznawania mowy [23]. Dzięki zastosowaniu procesu parametryzacji, a więc wykorzystaniu parametrów i ich przeanalizowaniu system pozwala na zauważenie różnic, o istnieniu, których człowiek mógł nie zdawać sobie sprawy [24].

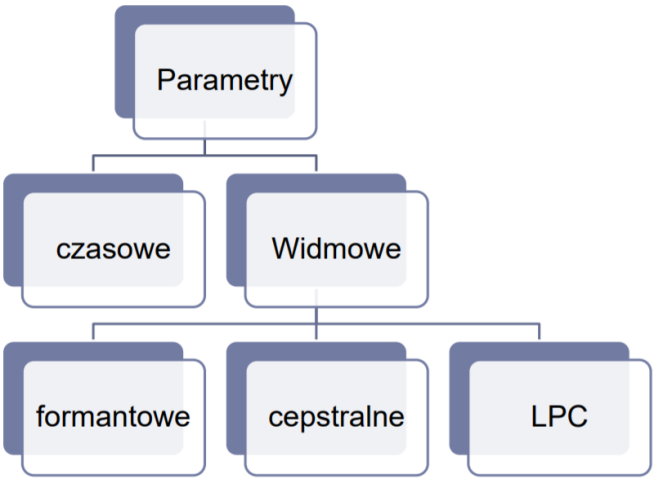


Rys. 4. Ogólny schemat parametryzacji [25]

Celem parametryzacji jest [24]:

- umożliwienie rozróżniania obiektów różnych klas,
- rozpoznanie obiektu nieznanej klasy,
- zweryfikowanie czy dany obiekt należy do danej klasy.

Dokonano następującej klasyfikacji parametrów [24]:



Rys. 5. Schemat klasyfikacji parametrów [26]

Metody czasowe z powodu zbyt wysokiego rozproszenia informacji użytecznych nie są wykorzystywane jako efektywny sposób opisu mowy. Z kolei najczęściej wykorzystywanymi metodami są metody widmowe [27].

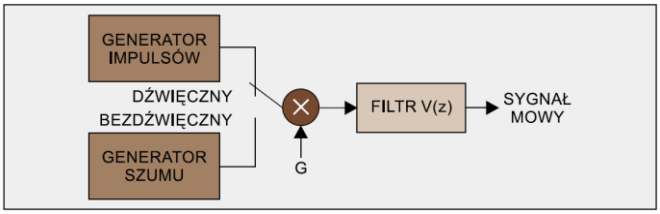
Najczęściej stosowanymi metodami parametryzacji sygnału mowy są [16,23,24,27]:

- współczynniki predykcji liniowej (LPC)
- współczynniki analizy cepstralnej w skali mel (MFCC)

Zaletami LPC jest wysoka precyzja estymacji parametrów mowy i szybkie działanie [28].
Pierwsze wykorzystanie LPC notowane jest na 1966 rok. Metody tej użyli Saito i Itakura [29].
MFCC pierwszy raz opisano w artykule autorstwa Davisa i Mermelsteina. Artykuł powstał w 1980 roku, a więc 14 lat później niż pierwsze wykorzystanie LPC [30].
MFCC opiera się na metodzie parametryzacji mowy, która wykorzystuje analizę podpasmową sygnału poprzez filtry pasmowo-przepustowe, które są rozłożone równomiernie na melowej skali częstotliwości [30].

Współczynniki predykcji liniowej (z ang. Linear Prediction Coefficients - LPC). W metodzie, która

wykorzystuje współczynniki predykcji liniowej dokonywane jest założenie, iż mowa jest to sygnał, który powstał w wyniku splotu danego pobudzenia i wolnozmiennego filtru, który został skojarzony z transmitancją toru głosowego człowieka [22].



Rys. 6. Uogólniony schemat powstawania sygnału mowy [31]

Na przedstawionym schemacie symbolem G oznaczono pobudzenie generowane przez krtań oraz płuca. Filtr V(z) odpowiedzialny jest za modelowanie traktu głosowego człowieka (dotyczy jamy ustnej, głosowej i gardłowej).

Podczas mówienia człowiek generuje głoski. Wytwarzane przez człowieka głoski podzielono na dźwięczne i bezdźwięczne. W przypadku głosek bezdźwięcznych filtr pobudzany będzie sygnałem szumowym, natomiast dla głosek dźwięcznych filtr będzie pobudzany ciągiem impulsów.

Ciąg impulsów generuje pewną częstotliwość, którą nazwano *częstotliwością tonu krtaniowego*. Częstotliwość tonu krtaniowego definiuje częstotliwość drgania strun głosowych człowieka. Z powodu odmienności wspominanego parametru dla każdego mówcy, parametr ten może być wykorzystywany do rozpoznawania głosu [22].

Jeśli zostanie ustalone założenie upraszczające (dla głosek nosowych) to filtr, który jest odpowiedzialny za modelowanie traktu głosowego może zostać przedstawiony w postaci modelu 'same bieguny'. Przy takim założeniu wspominany filtr będzie filtrem o nieskończonej odpowiedzi impulsowej.

Finalnie, aby obliczyć transmitancję H(z) omawianego modelu należy skorzystać ze wzoru [22]:

$$H(z)=G(z)V(z)\frac{G}{A(z)}=\frac{G}{1-\sum_{k=1}^{p}a_k z^{-k}} \tag{3}$$

gdzie: H(z) – transmitancja modelu, G(z) – pobudzenie, V(z) – filtr, a_k – współczynniki predykcji, p – rząd predykcji, G – intensywność pobudzenia traktu głosowego.

Przedstawiony powyżej wzór opisuje otrzymany sygnał mowy. Aby opisany sygnał mowy stanowił wiarygodny model mówcy, należy nieustannie aktualizować utworzony model. Głos człowieka jest zmienny, natomiast sygnał mowy człowieka w pewnych, krótkich przedziałach 10-20 [ms] jest stacjonarny. Stacjonarność sygnału mowy człowieka powoduje, iż właśnie w krótkich przedziałach współczynniki omawianego modelu H(z) nie podlegają zmianie.

W celu aktualizacji modelu mówcy korzysta się z metody liniowej predykcji sygnału mowy. Metoda ta pozwala na obliczenie parametrów konkretnego modelu dla kolejnych ramek sygnału (bloków dźwięku o określonej długości).

We wspominanej metodzie zakłada się, iż próbkę sygnału mowy można uzyskać wskutek kombinacji liniowej p poprzednich próbek. Można rzec, iż metoda LPC pozwala na 'przewidzenie' wartości sygnału mowy na podstawie p poprzednich wartości sygnału mowy [22].

Wartość rzeczywista, która opisuje próbkę sygnału mowy jest inna niż wartość próbki, ponieważ w metodzie LPC nie jest brany pod uwagę wpływ sygnału pobudzenia. Wskutek pominięcia wpływu sygnału pobudzenia pojawia się 'błąd', który stanowi różnicę pomiędzy dwiema wartościami: rzeczywistą oraz przewidzianą.

Ideą LPC jest znalezienie zbioru współczynników predykcji. Współczynniki predykcji można otrzymać podczas minimalizowania średniokwadratowej wartości błędu E dla całej ramki. Do obliczenia błędu można użyć metody autokorelacji [22].

Po dokonaniu obliczeń pochodnych błędu względem kolejnych współczynników predykcji, a następnie po przyrównaniu ich do zera otrzymujemy ciąg równań danych przedstawiony w postaci iloczynu macierzy [22]. Następnym krokiem jest rozwiązanie układu równań liniowych, gdzie wynikiem jest wyznaczona wartość a_k

Gdy wszystkie parametry LPC dla wszystkich ramek zostaną obliczone to efektem jest ciąg wielowymiarowych wektorów (każdy wektor to p współczynników a_k) [22].

Otrzymane w ten sposób wektory finalnie wykorzystywane są do obliczenia podobieństwa pomiędzy analizowaną wypowiedzią mówcy, a zarejestrowanym modelem mówcy. Analizowana wypowiedź mówcy zostaje opisana za pomocą sekwencji wektorów testowych [22].

Wpółczynniki mel-cepstralne (z ang. Mel-frequency Cepstral Coefficents - MFCC) są parametrami, które wykorzystuje się w akustyce mowy i kompresji sygnałów fonicznych. Parametry mel-cepstralne powstają z cepstrum sygnału prezentowanego w skali melowej [24].

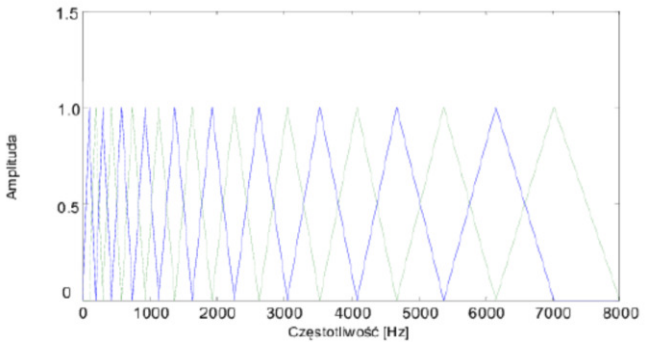
Podstawowa różnica pomiędzy MFCC, a rzeczywistym cepstrum sygnału jest taka, iż w MFCC wykorzystywane jest nieliniowane skalowanie częstotliwości. Do wspomnianego skalowania częstotliwości wykorzystywana jest skala mel [23].

Dla porównania poniżej przedstawiono zależność występującą pomiędzy częstotliwością w skali mel, a częstotliwością opisaną przy pomocy Hz [23]

$$f_{mel}(f_{Hz})=2595\log_{10}\left(1+\frac{f_{Hz}}{700}\right) \tag{4}$$

gdzie: f_{mel} – częstotliwość w skali mel, f_{Hz} – częstotliwość w skali Hz.

Skala melowa uzyskiwana jest wskutek filtracji sygnału bankiem filtrów o charakterystyce trójkątnej. K-ty współczynnik mel-cepstralny odpowiada zawartości k-tego pasma. Liczba pasm mieści się w przedziale od 12 do 20 [24].



Rys. 7. Filtracja sygnału bankiem filtrów o charakterystyce trójkątnej [32]

Obliczanie współczynników MFCC zostało podzielone na kilka kroków [23].

Krok 1: Poddanie otrzymanego sygnału mowy procesowi preemfazy. We wspomnianym procesie dokonywana jest filtracja formująca. Podczas filtracji formującej następuje osłabienie składowych (dotyczy tylko składowych o małych częstotliwościach), składowe o wysokich częstotliwościach zostają wzmocnione [23].

Wzór preemfazy określono jako:

$$x'_n = x_n - a x_{n-1} \tag{5}$$

gdzie: x_n - sygnał przed procesem preemfazy, x'_n - sygnał po procesie preemfazy, a - wartość zazwyczaj ustalana jako 0,97.

Krok 2: Następnie dokonywany jest proces ramkowania sygnału. Nazwa procesu wywodzi się z faktu, iż w omawianym kroku dokonywane jest podzielenie sygnału na krótkie fragmenty. Fragmenty te nazwano ramkami (z ang. frames). Dopuszcza się, aby kolejne ramki nakładały się na siebie [23].

Krok 3: Gdy ramkowanie sygnału zostało wykonane, rozpoczyna się proces okienkowania (z ang. windowning). Okienkowanie wykonywane jest przy użyciu tzw. okna Hamminga [23].

$$Ham(N) = 0{,}54 - 0{,}46 \cos\left(2\pi \frac{n-1}{N-1}\right) \tag{6}$$

gdzie: Ham(N) – okno Hamminga, N – długość ramki, n = 1, 2,..., N.

Krok 4: Na wszystkich ramkach przeprowadzany zostaje proces szybkiej transformacji Fouriera (FFT). W zależności od zaprogramowanego algorytmu, uzyskuje się widmo mocy $|FFT|^2$ albo widmo amplitudowe |FTT| [23].

Krok 5: Do zestawu filtrów H_m wprowadzone zostają widma opisane w kroku 4. Liczba filtrów jest zróżnicowana, a definiuje ją sposób zaprogramowania algorytmu. Najpopularniejszymi filtrami są filtry trójkątne.

Efektem zastosowania filtrów jest wygenerowanie na wyjściu każdego z filtrów energii pasma S_m.

Krok 6: Za pomocą logarytmu energii dokonywane jest osłabienie czułości filtrów na bardzo głośne oraz bardzo ciche dźwięki. Przeprowadzony również zostaje proces modelowania nieliniowej amplitudowej wrażliwości ucha ludzkiego. Dzięki zastosowaniu logarytmu energii polepsza się jakość rozpoznawania [23].

Krok 7: W ostatnim kroku przeprowadzana zostaje dyskretna transformata kosinusowa (z ang. Discrete Cosine Transform - DCT).

Końcowe wartości współczynników MFCC określone zostały za pomocą wzoru [23]:

$$c_i = \sqrt{\frac{2}{M}} \sum_{m=1}^{M} \log(S_m) \cos\left(\frac{\pi i}{M}\right)(m - 0{,}5) \tag{7}$$

gdzie: c_i – wartość współczynnika MFCC, i – numer współczynnika (zazwyczaj wartość i>1), M – liczba użytych filtrów, S_m – energia pasma, m – numer filtra.

5. Badania skuteczności

W ramach badania skuteczności wybrano dwa zestawy próbek głosu – treningowy i testowy. Do testu wybrano nagrania dziesięciu mówców – po 30 w zestawie treningowym i po 10 w testowym. Sumarycznie – 400 próbek głosu – średnio każda o długości ok. 2s. W procesie trenowania algorytmu wszystkie z 300 próbek zostało przetworzonych pod kątem cech MFCC i LPC oraz stworzono na ich podstawie książki kodowe. W ramach testu dla każdej ze 100 próbek testowych wyliczono dystans euklidesowy w stosunku do każdego z wpisów w książce kodowej. Wpis z najmniejszą odległością od testowanej próbki traktowany był jako pasujący. Każda z próbek oznaczona była kodem mówcy, dzięki któremu można było jednoznacznie identyfikować autora wypowiedzi. Jeśli w trakcie testu dopasowano wytrenowaną próbkę tego samego autora – zaliczano test jako udany. Na podstawie ilości prawidłowych dopasowań (osobno dla ekstrakcji cech MFCC i LPC) oraz ilości wszystkich prób wyliczono procentowe współczynniki skuteczności.

Poniżej przedstawiono wynik badania skuteczności algorytmu porównującego próbki głosowe:

Tabela 2. Wynik badania skuteczności algorytmu porównującego próbki głosowe

Parametr	MFCC	LPC
Skuteczność procentowa	34%	61%

Zdecydowanie większą skuteczność można zauważyć w przypadku porównania cech ekstrahowanych z użyciem algorytmu LPC. Skuteczność weryfikacji mówców przy użyciu algorytmu LPC wyniosła 61%, tym samym osiągając wynik lepszy o 27% aniżeli skuteczność algorytmu MFCC. Biorąc pod uwagę fakt, iż metoda LPC osiągnęła wynik 61%, a MFCC 34% oznacza to, iż LPC jest prawie dwa razy skuteczniejszą metodą weryfikacji mówców. W przypadku próby wdrożeń wybranego algorytmu do danych rozwiązań technologicznych z pewnością lepszym rozwiązaniem będzie wdrożenie algorytmu LPC. Pod uwagę należy wziąć fakt, iż skuteczność weryfikacji mówcy przy użyciu metod biometrycznych na poziomie 61% może nie być wystarczająca do zastosowań na szeroką skalę. Kolejnym kierunkiem badań w celu usprawnienia działania rozpoznawania mówcy powinna być analiza algorytmu LPC pod kątem rodzaju analizowanych próbek – jakości dźwięku, zaszumienia otoczenia, płci mówcy

czy wieku. Warto również sprawdzić jak ten algorytm zachowa się w stosunku do różnej liczby próbek treningowych oraz testowych.

6. Podsumowanie

Przeprowadzono badania, w których określono skuteczność obu z tych rodzajów autoryzacji. Fakt dużego wpływu procesu ekstrakcji parametrów z próbek danych dowodzą badania, w których ten sam sposób porównania próbki zestawiono w stosunku do dwóch różnych repozytoriów (badania autoryzacji odcisku palca). Wyniki znacznie się różnią – zależnie od sposobu analizy próbki i ekstrakcji danych. Dużo większą skutecznością może pochwalić się algorytm, który wykrywał minucje w liniach papilarnych. Znacznie niższe wyniki zaobserwowano w przypadku podania punktów kluczowych z detektora Harrisa do deskryptora obrazu. Warto również zauważyć fakt, że sam mechanizm porównania odcisków bazujący na deskryptorach daje wysokie, lecz nie stuprocentowe wyniki. Kolejnym kierunkiem badań w tym kontekście powinna być modyfikacja algorytmu porównania modeli próbek, uzyskanych w wyniku ekstrahowania cech. Dzięki badaniom wiadomo, że dużo większą skuteczność da proces ekstrakcji bazujący na wykrywaniu minucji.

W przypadku autoryzacji głosowej na podstawie badań udowodniono, że dużo lepszą skuteczność autoryzacji daje algorytm LPC – jego wynik był dwukrotnie lepszy niż w przypadku MFCC. Procentowa skuteczność autoryzacji jest na zadowalającym poziomie, lecz w celu wykorzystania tego rozwiązania w produkcyjnym środowisku należałoby ją znacznie zwiększyć. Kolejnym kierunkiem badań w tym aspekcie powinna być analiza algorytmu pod kątem trenowanych i badanych próbek – ich jakości oraz cech dotyczących mówcy (takich jak wiek, płeć itd.).

Literatura

[1] Thai R. - Fingerprint ImageEnhancement and Minutiae Extraction - https://www.peterkovesi.com/studentprojects/raymondthai/RaymondThai.pdf - 8.06.2019

[2] Dokumentacja OpenCV - adaptacyjna normalizacja histogramu https://docs.opencv.org/3.1.0/d5/daf/tutorial_py_histogram_equalization.html - 28.05.2019

[3] Repozytorium python-fingerprint-recognition https://github.com/kjanko/python-fingerprint-recognition - 28.05.2019

[4] TheAILearner - Adaptive Histogram Equalization (AHE) - https://theailearner.com/2019/04/14/adaptive-histogram-equalization-ahe/ - 08.06.2019

[5] WaveMetrics - progowanie obrazu https://www.wavemetrics.com/products/igorpro/imageprocessing/thresholding - 28.05.2019

[6] Repozytorium biometrics - GitHub https://github.com/rtshadow/biometrics - 28.05.2019

[7] Shah A. - Through The Eyes of Gabor Filter https://medium.com/@anuj_shah/through-the-eyes-of-gabor-filter-17d1fdb3ac97 - 28.05.2019

[8] Błaszczyk Ł. - Filtry Gabora i ich zastosowanie w obrazowaniu medycznym http://pages.mini.pw.edu.pl/~blaszczykl/nauka/prace/inz_biomed.pdf - 28.05.2019

[9] Vector magnitude and direction review - https://www.khanacademy.org/math/precalculus/vectors-precalc/component-form-of-vectors/a/vector-magnitude-and-direction-review - 08.06.2019

[10] Bazy odcisków FVC2004 http://bias.csr.unibo.it/fvc2004/databases.asp - 28.05.2019

[11] Khanyile N.P., Tapamo J.R., Dube E. - A Comparative Study of Fingerprint Thinning Algorithms https://pdfs.semanticscholar.org/f0e3/947dae712e8c27ee297e2ef569a369dcccbe.pdf - 28.05.2019

[12] Gońda S., Juszczak D. - Identyfikacja osób poprzez ich odciski palców http://sequoia.ict.pwr.wroc.pl/~witold/aiarr/2009_projekty/odciski/ - 28.05.2019

[13] Więcław Ł.- A minutiae-based matching algorithms in fingerprint recognition systems https://www.researchgate.net/publication/228644313_A_minutiae-based_matching_algorithms_in_fingerprint_recognition_systems - 28.05.2019

[14] Chaudhari A. S., Dr. Girish K. Patnaik, Patil S. S. - Implementation of Minutiae Based Fingerprint Identification Systemusing Crossing Number Concept https://pdfs.semanticscholar.org/49ad/2473ecc1dcbfdfc9a981f5191a1224107ef1.pdf - 28.05.2019

[15] Paulino A. A., Feng J., Jain - Latent A. K. - Fingerprint Matching using Descriptor-Based Hough Transform http://biometrics.cse.msu.edu/Publications/Fingerprint/PaulinoFengJain_LatentFPMatching_DescriptorBasedHoughTransform_IJCB11.pdf - 28.05.2019

[16] Ślot K., Wybrane zagadnienia biometrii, Wydawnictwa Komunikacji i Łączności, Warszawa, 2008

[17] Bolle R.M., Connell J. H., Pankanti S., Ratha N. K., Senior A. W., - Biometria, Wydawnictwa Naukowo-Techniczne, Warszawa, 2008

[18] Zrozumiałość mowy - https://livesound.pl/tutoriale/4629-zrozumialosc-mowy - 28.05.2019

[19] Nasal speech sound - https://www.britannica.com/topic/nasal-speech-sound - 28.05.2019

[20] Termin ‘kadencja’ - https://sjp.pl/kadencja - dostęp 28.05.2019

[21] Termin ‘inflection’ - https://www.vocabulary.com/dictionary/inflection , [28.05.2019]

[22] Dustor A., Izydorczyk J., Rozpoznawanie mówców - http://www.przegladtelekomunikacyjny.pl/archive/WWW/artrec/dustor2-3'2003.pdf - 28.05.2019

[23] Kacprzak S., Inteligentne metody rozpoznawania dźwięku, Politechnika Łódzka, Łódź, 2010 http://www.dsp.agh.edu.pl/_media/pl:homepage:msc_s_kacprzak.pdf - dostęp 28.05.2019

[24] Parametryzacja sygnału mowy. Perceptualne skale częstotliwości - https://sound.eti.pg.gda.pl/student/amowy/am_04_parametryzacja.pdf - 28.05.2019

[25] Ogólny schemat parametryzacji [rys.] https://sound.eti.pg.gda.pl/student/amowy/AM_04_parametryzacja.pdf - 28.05.2019

[26] Schemat klasyfikacji parametrów [rys.] https://sound.eti.pg.gda.pl/student/amowy/AM_04_parametryzacja.pdf - 28.05.2019

[27] Gałka J., Optymalizacja parametryzacji sygnału w aspekcie rozpoznawania mowy polskiej, Akademia Górniczo–Hutnicza im. Stanisława Staszica w Krakowie, Kraków, 2008 https://docplayer.pl/24752866-Optymalizacja-parametryzacji-

sygnalu-w-aspekcie-rozpoznawania-mowy-polskiej.html - 28.05.2019

[28] Rabiner L. R., Schafer R. W. - Digital procesing of speech signal. Prentice Hall, Englewood Cliffs, NJ, 1978.

[29] Basztura C. - Źródła, sygnały i obrazy akustyczne. Wydawnictwo Komunikacji i Łacznosci, Warszawa, 1988.

[30] Davis S., Mermelstein P. - Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. Acoustics, Speech and Signal Processing, IEEE Transactions on, Sier. 1980.

[31] Uogólniony schemat powstawania sygnału mowy [rys.] http://www.przegladtelekomunikacyjny.pl/archive/WWW/artrec/dustor2-3'2003.pdf - 28.05.2019

[32] Filtracja sygnału bankiem filtrów o charakterystyce trójkątnej [rys.] https://sound.eti.pg.gda.pl/student/amowy/AM_04_parametryzacja.pdf - 28.05.2019

[33] Wanat I., Iwaniec M., Tworzenie modelu akustycznego na potrzeby weryfikacji mówcy przy użyciu Ukrytych Modeli Markowa - http://www.kms.polsl.pl/mi/pelne_9/31.pdf - 28.05.2019

JOURNAL of COMPUTER SCIENCES INSTITUTE

JCSI 13 (2019) 332-338
Wysłane: 2019-09-05
Przyjęte: 2019-09-11

WebAssembly jako alternatywa dla JavaScript w tworzeniu nowoczesnych aplikacji internetowych

Dawid Suryś*, Piotr Szłapa, Maria Skublewska-Paszkowska
Politechnika Lubelska, Instytut Informatyki, Nadbystrzycka 36B, 20-618 Lublin, Polska

Streszczenie. W artykule opisano wpływ wykorzystania standardu WebAssembly na wydajność aplikacji internetowych. Wykorzystano oparty o WebAssembly szkielet aplikacji Blazor. Pokazano, iż można wykonać w pełni funkcjonalną aplikację SPA (ang. Single Page Application) pisząc kod aplikacji w języku C#. Wykonano drugą aplikację wykorzystując szkielet Angular. W obu aplikacjach zaimplementowano te same funkcjonalności. Dla wykonanych aplikacji porównano czasy ładowania w przeglądarce oraz rozmiar przesyłanych danych. Zbadano wpływ pamięci podręcznej przeglądarki i kompresji gzip. Zbadano wydajność obsługi żądań HTTP. Porównano wydajność kodu WebAssembly z kodem JavaScript w zadaniu sortowania n-elementowej listy obiektów. Zbadano wydajność aplikacji Blazor w modyfikowaniu drzewa DOM. Porównano wybrane metryki kodu obu aplikacji. JavaScript okazał się wydajniejszy w zadaniach związanych z wykorzystaniem API przeglądarki. WebAssembly był lepszy w zadaniach obliczeniowych.

Słowa kluczowe: JavaScript; WebAssembly; Blazor; Angular

*Autor do korespondencji.

Adres e-mail: dawid.surys@pollub.edu.pl

WebAssembly as an alternative solution for JavaScript in developing modern web applications

Dawid Suryś*, Piotr Szłapa, Maria Skublewska-Paszkowska
Institute of Computer Science, Lublin University of Technology, Nadbystrzycka 36B, 20-618 Lublin, Poland

Abstract. The article describes the impact of using WebAssembly on the performance of web applications. A Blazor framework based on WebAssembly was used. The paper shows that it is possible to create fully functioning Single Page Application using C# programming language. The second application was made using Angular framework. Both applications implement the same functionalities. For prepared applications loading time and size of transferred data was measured. The impact of using browser cache memory and gzip compression was examined. The performance of handling http requests using GET and POST methods were measured. The performance of WebAssembly against JavaScript code in task of sorting a list of N objects was compared. The performance of modifying html DOM was examined. Selected code metrics of written applications were compared. JavaScript presents better performance with tasks related with browser API. WebAssembly was better for computing.

Keywords: JavaScript; WebAssembly; Blazor; Angular

*Corresponding author.

E-mail addresses: dawid.surys@pollub.edu.pl

1. Wstęp

Ciągły rozwój Internetu i jego rozpowszechnienie pociąga za sobą wzrost wymagań użytkowników. Strony www nie zawierają już tylko statycznych informacji, ale coraz częściej wspierają one różne procesy i tworzą pełnoprawne aplikacje nieustępujące możliwościom tradycyjnym aplikacjom desktopowym. Widoczna jest zmiana sposobu tworzenia takich aplikacji. W nowoczesnych aplikacjach webowych część serwerowa jest odseparowana od części klienckiej. Ta druga z roku na rok staje się coraz ważniejsza i coraz bardziej rozbudowana. Według ankiet przeprowadzonych przez portal *stackoverflow.com* w latach 2015/2016 ok. 6% programistów określało siebie jako front-end developer [1][2], w 2017 jest to 12% [3], w roku 2018 ponad 35% [4]. Od nowoczesnych aplikacji wymaga się również dużej wydajności. Według artykułu [5] 40% użytkowników opuszcza stronę, jeśli wczytuje się on dłużej niż 3 sekundy. Natomiast 79% osób robiących zakupy online deklaruje, że jest mniej prawdopodobne, że wrócą do danego sklepu, jeśli nie byli zadowoleni z wydajności strony.

Do tworzenia warstwy prezentacji aplikacji internetowych od wielu lat wykorzystywany jest język JavaScript. Początkowo został on zaprojektowany do prostych manipulacji na stronie www. Jednak bardzo szybki rozwój tego języka spowodował, że aktualnie oferuje on bardzo wiele możliwości i jest wykorzystywany zarówno do tworzenia warstwy usług jak i warstwy prezentacji aplikacji internetowych. Mimo dużej ewolucji jaką przeszedł język JavaScript wciąż posiada on problemy związane z wydajnością. Odpowiedzią na część z tych problemów jest niskopoziomowy język WebAssembly, który pozwala na

wykonanie kodu binarnego w przeglądarce z wydajnością bliską wydajności natywnego kodu [6]. WebAssembly nie jest językiem, w którym kod może być bezpośrednio pisany przez programistę. Stanowi raczej kod pośredni do którego mogą być skompilowane programy napisane w takich językach programowania jak np. C, C++, Rust [7].

W niniejszym artykule postanowiono zbadać możliwości standardu WebAssembly w kontekście tworzenie aplikacji internetowych. Postawiono tezę mówiącą, iż aplikacje wykorzystujące technologię WebAssembly oferują większą wydajność niż te które jej nie wykorzystują. Skupiono się na praktycznych przykładach. W tym celu wykorzystano rozwijany przez Microsoft framework *Blazor,* który swoje działanie opiera właśnie na języku WebAssembly. Przy pomocy wspomnianego szkieletu aplikacji (ang. framework) stworzono demonstracyjną aplikację. Aplikacja posłużyła do przeprowadzenia badań. Jako punkt odniesienia w analizie porównawczej wykonano analogiczną aplikację w technologii Angular która opiera swoje działanie o JavaScript.

2. Przegląd literatury

Brakuje publikacji porównującej aplikacje wykorzystujące WebAssembly z aplikacjami zbudowanymi na bazie szkieletów programistycznych opartych na języku JavaScript.

W artykule zatytułowanym "Gap:Analyzing the Performance of WebAssembly vs. Native Code" [6] autorzy dokonali porównania kodu skompilowanego do WebAssembly (WASM) uruchamianego w przeglądarce z natywnym kodem uruchamianym bezpośrednio z poziomu systemu operacyjnego. Autorzy przytoczyli wyniki podobnych badań tego typu. W tych badaniach kod binarny uruchamiany z przeglądarki okazywał się ok. 10% wolniejszy od kodu natywnego. Jednak wszystkie te badania opierały się na kodzie niewielkich rozmiarów (ok 100 linii kodu). W badaniu ze wspomnianego artykułu użyto bardziej rozbudowanej aplikacji napisanej w języku C++. Wyniki wskazały na znacznie gorszą wydajność WebAssemby. W przeglądarce Firefox zanotowano spadek o ok. 50% względem natywnego kodu, natomiast w przeglądarce Google Chrome o 89%.

Artykuł [8] porusza temat wydajności aplikacji SPA Autorzy postanowili poddać analizie metody przyśpieszania procesu ładowania aplikacji tego typu. Na potrzeby badania wykonano prostą aplikację z użyciem frameworka AngularJS. Następnie przeprowadzono serię eksperymentów sprawdzających skuteczność różnych metod poprawy wydajności ładowania aplikacji. Badano rozmiar przesyłanych do przeglądarki plików oraz czas ładowania aplikacji. Pierwsza metoda optymalizacji polegała na połączeniu wszystkich plików CSS oraz JavaScript i przesłaniu ich do klienta w jednym żądaniu. Druga metoda opierała się na usunięciu nieużywanych reguł CSS. Kolejną badaną metodą była minifikacja kodu JavaScript, czyli usunięcie wszystkich białych znaków, komentarzy oraz zmiana nazw zmiennych na krótsze. Ponadto sprawdzono jaki wpływ na wydajność ma użycie protokołu HTTP/2 oraz kompresja z użyciem programu gzip. Najlepsze rezultaty zmniejszenia rozmiaru plików otrzymano dla metody minifikacji kodu JS. Z oryginalnego rozmiaru aplikacji 4,37MB udało się osiągnąć 1,57MB oraz 1,54MB przy dodatkowo zastosowanej kompresji. Rozmiar nagłówków zależy w dużej mierze od liczby żądań. Znaczne ograniczenie liczby żądań osiągnięto przez złącznie plików *.js oraz *.css. Przy takiej samej liczbie żądań lepsze kompresja nagłówków jest zauważalna w protokole HTTP/2. W przypadku analizy czasu ładowania aplikacji nie zaobserwowano widocznej różnicy między zastosowanie protokołu HTTP 1.1 czy HTTP/2. Również mechanizm PUSH PROMISE nie spowodował skrócenia czasu ładowania aplikacji.

Istnieje wiele artykułów, w których dokonano porównania frameworka Angular2 z innymi bibliotekami JavaScript. Jednym z takich artykułów jest ten autorstwa J. Kalinowska, B. Pańczyk [9]. Dokonano w nim analizy porównawczej frameworka Angular2 z ReactJS. Analiza oparta była na przygotowanych specjalnie w tym celu aplikacjach. W porównaniu uwzględniono strukturę i wydajność aplikacji, wybrane metryki kodu, jakość dokumentacji oraz wsparcie społecznościowe. Na podstawie badań przeprowadzono wnioski. Jednym z nich jest stwierdzenie, iż Angular2 jest lepszym narzędziem dla bardziej doświadczonego programisty.

Autorzy artykułu [10] w swojej pracy dokonują analizy porównawczej kilkunastu popularnych bibliotek. Przeprowadzono badanie ankietowe. W badaniu wzięło udział 18 osób z 6 krajów z całego świat. Sklasyfikowano badane osoby według następujących cech: doświadczenie zawodowe, używane biblioteki, dziedzina działalności (programista back-end lub front-end), rozmiar firmy, typ firmy. Największą popularność wśród ankietowanych miała biblioteka jQuery (8 osób) oraz framework AngularJS (6 osób). Dwie badane osoby używają swoich własnych rozwiązań w pracy, jedna osoba odpowiedziała, że nie korzysta z żadnych szkieletów aplikacji. Większość ankietowanych wskazało więcej niż jeden framework. Jako programistę front-end określiło się ośmiu programistów, siedmiu zaznaczyło, że zajmują się zarówno częścią kliencką jak i serwerową, natomiast trzech wyłącznie częścią serwerową. Większość osób pracowało w średniej wielkości firmie. Autorzy badania przyjęli następujące kryteria oceny: wydajność i rozmiar frameworka, stopień skomplikowania i łatwość nauki, wsparcie społeczności, funkcjonalności i cena. Wyciągnięto następujące wnioski. Dokumentacja frameworka powinna jasno wskazywać, jeśli jej poprawne działanie polega głównie na zasobach sprzętowych środowiska, w którym jest uruchamiany. Liczba linii kodu potrzebnych do napisania aplikacji w danym frameworku powinna być tak mała jak to tylko możliwe. Dokumentacja frameworka powinna być precyzyjna, zawierać przykłady implementacji typowych zadań oraz umożliwiać programistom na szybkie znalezienie wskazówek dotyczących implementacji danej funkcjonalności. Biblioteki, które mają rozbudowaną społeczność i są rozwijane od dłuższego czasu, powinny być wybierane. Struktura aplikacji zbudowanej według frameworka powinna być modułowa, a zamiana w jednym module nie powinna wymagać zmiany w inny. Framework

powinien pozwalać na łatwe importowanie do projektu zewnętrznych bibliotek. Darmowe rozwiązania oraz biblioteki rozwijane jako projekty open-source są chętniej wybierane od tych płatnych.

3. Aplikacja zapisu studentów na dodatkowe zajęcia

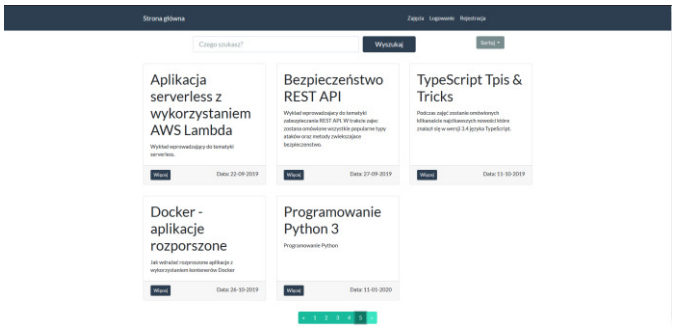
Na potrzeby badań powstały 2 aplikacje o takim samym graficznym interfejsie użytkownika. Każda z nich wykonuje te same zadania. Omawiany system informatyczny wspomaga obsługę zapisywania się studentów na dodatkowe zajęcia. Służy również jako ułatwienie prowadzenia kursów przez wykładowców. Umożliwia tworzenie oraz edycję zajęć dla prowadzących, a także zapisywanie się na zajęcia oraz potwierdzanie obecności w przypadku roli studenta. System wyróżnia następujące poziomy uprawnień: administrator, firma, wykładowca, student. Zawiera mechanizmy uwierzytelniania i autoryzacji użytkowników. Oczywiście w rozdziale zostały wymienione tylko najważniejsze możliwości omawianego systemu informatycznego.

Realizowana aplikacja jest aplikacją webową z rozdzieloną częścią serwerową oraz częścią kliencką. Część kliencką stanowią dwie aplikacje SPA. Do napisania jednej aplikacji wybrano framework Blazor. Aktualnie jest to projekt, który otrzymał oficjalne wsparcie od Microsoft i wchodzi w skład .Net Foundation. Framework ten wybrano, ponieważ działa on w oparciu o WebAssembly, tzn. wykonuje kod binarny w przeglądarce. Standardowe aplikacje wykonują skrypty JavaScript. Kod aplikacji Blazor został napisany w języku C#. Do napisania drugiej aplikacji klienckiej wybrano framework Angular. Od pewnego czasu znajduje się on w czołówce popularności frameworków i bibliotek JavaScript. Wyboru Angulara dokonano, ponieważ pozwala on na zachowanie bardzo przejrzystej struktury projektu. Ponadto oferuje możliwość korzystania z zalet języka TypeScript, takich jak statyczne typowanie. Dzięki temu można częściowo ograniczyć popełnione błędy w trakcie pisania aplikacji.

Część serwerowa została stworzona zgodnie ze wzorcem REST API. Wykonane ją z użyciem języka C# oraz frameworka ASP.NET CORE w wersji 2.2. Serwer stanowi aplikację typu Web API. Aplikacja serwerowa współpracuje z bazą danych SQL Server 2016. Baza danych od firmy Microsoft posiada duże zastosowanie w aplikacjach ASP.NET. Spowodowane jest to tym, iż firma z Redmond udostępnia dość szczegółową oficjalną dokumentację oraz inne źródła wiedzy. Do obsługi bazy danych w aplikacji wykorzystano ORM Entity Framework Core. Do stworzenia struktury bazy danych i relacji wykorzystano popularne środowisko SQL Server Management Studio.

Głównym środowiskiem pracy był rozbudowany edytor tekstu Visual Studio Code. Oferuje on szereg rozszerzeń. Tym samym był wykorzystywany do napisania każdej ze składowych części projektu tj. aplikacji serwerowej i aplikacji klienckich. Jako dodatkowe wsparcie momentami wykorzystywano pełne środowisko Visual Studio.

Rys. 1 przedstawia zrzut ekranu zawierający stronę startową aplikacji.



Rys. 1 Strona startowa aplikacji

4. Przebieg badania

Podstawowym zdaniem, jakie sobie postawiono, było udowodnienie tezy, „Możliwe jest stworzenie funkcjonalnej aplikacji SPA z wykorzystaniem standardu WebAssembly i języka programowania C#". Ponadto aplikacja powinna dać się uruchomić w większości dostępnych przeglądarek internetowych bez konieczności instalowania dodatkowych rozszerzeń lub wtyczek. Wszystkie przewidziane funkcjonalności zostały zaimplementowane. Aplikacja została z powodzeniem przetestowana w przeglądarkach Google Chrom, Firefox, Edge oraz Google Chrome dla systemu Android.

Właściwym etapem badania było zmierzenie wydajności stworzonej aplikacji oraz samego kodu WebAssembly. Jako punkt odniesienia przy komentowaniu wyników wydajności aplikacji Blazor była bliźniacza wersja aplikacji napisana w technologii Angular. Wybrano ten szkielet aplikacji, ponieważ rozwijany jest od dłuższego czasu i powszechnie wykorzystywany przez deweloperów. Porównano czas ładowania aplikacji, ilość przesłanych danych oraz czasy obsługi żądań http.

Obok głównej aplikacji przygotowano osobno implementację dla dwóch scenariuszy badawczych. Pozwoliły one na zmierzenie czasu wykonania kodu sortowania listy obiektów oraz czasu generowania elementów DOM.

4.1. Stanowisko badawcze

Wszystkie opisane w procedurze badawczej eksperymenty zostały wykonane na jednym stanowisku badawczym. Stanowisko badawcze składa się z komputera stacjonarnego PC. Szczegółowa specyfikacja została zaprezentowana w tabeli 1.

Tabela 1. Specyfikacja stanowiska badawczego

System operacyjny	Procesor	Pamięć	Dysk twardy
Windows 10 Pro 1809 64bit	Intel Core i3-4100M 2,5GHz	12GB DDR3 (800MHz)	Samsung SSD 840 EVO 120GB

Podczas przeprowadzania badania zostały wykorzystane narzędzia:

- Nginx 1.16
- Kestrel Web Server
- Sql Server 13.0.4001.0
- Google Chrome 75
- Emscripten 1.38.21

4.2. Wydajność ładowania aplikacji

Celem eksperymentu było określenie wydajności ładowania aplikacji w przeglądarce. Do określenia wydajności przyjęto takie parametry jak całkowity czas ładowania w przeglądarce, moment zdarzenia „DOMContentLoaded", moment zdarzenia „load", liczba wykonanych żądań http i rozmiar pobranych danych. Eksperyment został wykonany w przeglądarce Google Chrome. Testowane aplikacje były ładowane z uruchomionego lokalnie serwera Nginx. Serwer nasłuchiwał na portach 81 i 82 kolejno dla aplikacji Blazor oraz aplikacji Angular. Aplikacje komunikowały się przez protokół http z aplikacją serwerową ASP.NET Core uruchomioną lokalnie w tym samym systemie przy użyciu Kestrel Web Server. Aplikacja serwerowa ze swojej strony łączyła się z bazą danych SQL Server również hostowaną lokalnie na tym samym systemie.

Eksperyment wykonano w 3 wariantach:

- wariant A - została wyłączona pamięć podręczna w przeglądarce oraz kompresja danych gzip na serwerze;
- wariant B – została włączona pamięć podręczna przeglądarki z pozostawioną nieaktywną kompresją;
- wariant C - wyłączono pamięć podręczną, natomiast włączono kompresję gzip na serwerze Nginx;

W przykładzie 1 przedstawiono użytą konfigurację modułu gzip.

Przykład 1. Konfiguracja Gzip

```
gzip  on;
gzip_min_length 1000;
gzip_proxied    expired no-cache no-store private auth;
gzip_types text/plain text/css application/json application/javascript
application/x-javascript text/javascript text/xml application/xml
application/rss+xml application/atom+xml application/rdf+xml
application/octet-stream application/wasm image/jpeg;
```

4.3. Wydajność operowania na kolekcji obiektów

Celem eksperymentu było zmierzenie czasu jaki jest potrzebny aplikacji na wykonanie sortowania listy obiektów n elementowych. Wykorzystano algorytm sortowania szybkiego (quick sort). Przygotowano cztery implementacje opisanego eksperymentu, są to:

- TypeScript z frameworkiem Angular
- JavaScript bez dodatkowych bibliotek
- C# z frameworkiem Blazor (WebAssembly)
- C++ skompilowany do kodu WASM

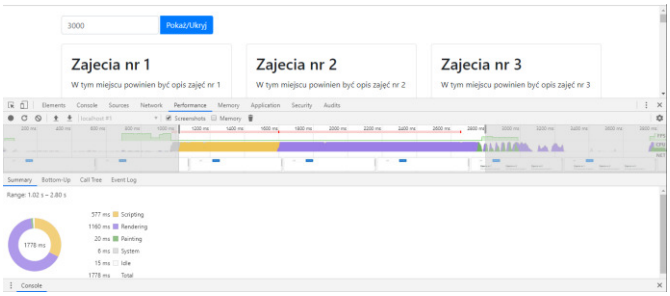
Czas wykonania sortowania zmierzono bezpośrednio w kodzie programu. Dla implementacji wykorzystujących JavaScript skorzystano z funkcji performance.now(), dla języka C# użyto klasy StopWatch z przestrzeni nazw System.Diagnostics, natomiast w przypadku C++ wykorzystano funkcję high_resolution_clock. Fragment kodu odpowiedzialny za pomiar czasu wykonania operacji został przedstawiony w przykładzie 2. Eksperyment przeprowadzono dla kolekcji 1 tys. elementów, 3 tys. elementów oraz 10 tys. elementów. Dla każdego rozmiaru wykonano po 10 prób dla każdej z implementacji.

Przykład 2. Pomiar czasu sortowania w C++

```
std::chrono::time_point<std::chrono::high_resolution_clock> t1 =
std::chrono::high_resolution_clock::now();
quicksort(lectures, 0, n);
std::chrono::time_point<std::chrono::high_resolution_clock> t2 =
std::chrono::high_resolution_clock::now();
std::chrono::duration<double> time_span =
std::chrono::duration_cast<std::chrono::milliseconds>(t2 - t1);
```

4.4. Wydajność generowania elementów DOM

W eksperymencie zamierzano sprawdzić jak framework Blazor poradzi sobie z dynamicznym modyfikowaniem drzewa DOM. Ze względu na fakt, że WebAssembly obecnie nie posiada bezpośredniego dostępu do DOM`u strony spodziewano się opóźnienia w działaniu względem aplikacji Angular. Do zmierzenia czasu renderowania w aplikacji wykorzystano dostępne w przeglądarce Google Chrome narzędzie do pomiaru wydajności. Wykonano testy generowania 1000, 3000 oraz 10000 elementów na stronie. Dla każdej serii wykonano 10 prób dla aplikacji Blazor oraz Angular. Rys.2 przestawia przykładowy pomiar wykonany dla 3 tys. elementów.



Rys. 2 Test wydajności renderowania widoku

4.5. Wydajność obsługi żądań HTTP

W tym eksperymencie postanowiono zmierzyć opóźnienie powstałe podczas wykonywania żądań HTTP. Na podstawie napisanych aplikacji Angular oraz Blazor zmierzono czas potrzebny na wysłanie żądania oraz otrzymania odpowiedzi. Podobnie jak w poprzednich eksperymentach aplikacje klienckie oraz aplikacja serwerowa zostały uruchomione lokalnie na tej samej maszynie. Wykonano żądania GET oraz POST w obydwu aplikacjach. Każdą operację powtórzono 10 razy. Żądanie GET zwracało do aplikacji listę zajęć dla pojedynczej wyświetlanej strony (6 elementów). Żądanie POST zapisywało w bazie nowe zajęcia na podstawie danych wysłanych z formularza.

4.6. Metryki kodu

W sytuacji, gdy wydajność aplikacji nie jest tak istotna lub dostępne rozwiązania nie różnią się znacząco w tym zakresie, większego znaczenia nabiera sam proces wytwarzania oprogramowania i związany z tym komfort pracy programisty. Mimo tego, że język JavaScript różni się znacząco od języka C# to dzięki zmianom wprowadzonym w standardzie ECMAScript 6 (m.in. klasy i dziedziczenie) oraz rozszerzeniu

możliwości języka oferowanej przez TypeScript tworzony kod ma wiele cech charakterystycznych dla paradygmatu OOP ang. Object Oriented Programming. W związku z powyższym kod aplikacji Blazor napisany w języku C# nie różni się znacząco od kodu aplikacji Angular napisanego w języku TypeScript. W obydwu aplikacjach zaimplementowano te same funkcjonalności. Ponadto obydwa frameworki posiadają podobną strukturę opartą o komponenty. Jednak każde rozwiązanie posiada swoje własne rozwiązania bardziej szczegółowych zagadnień oraz inną składnię tworzenia warstwy wizualnej. Blazor korzysta z znanego z aplikacji ASP silnika Razor, natomiast Angular posiada swój autorski silnik. Opisane różnie przekładają się na ilość kodu wymaganą do napisania przez programistę a pośrednio na jego wydajność. Zmierzono napisany kod przy wykonanych aplikacjach w postaci linii kodu programu. Porównano ze sobą osobno kod dla każdego komponentu aplikacji. Rozdzielono również część logiki komponentu od warstwy prezentacji. Usunięto wszystkie puste linie oraz komentarze.

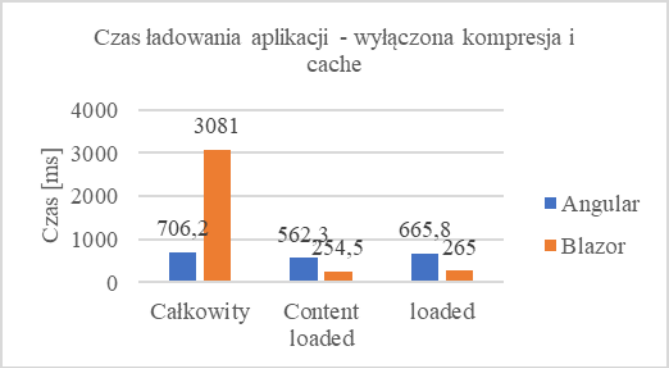
5. Wyniki

Rozmiar zbudowanej aplikacji Blazor jest znacznie większy od aplikacji Angular. Jest to spowodowane koniecznością pobrania środowiska uruchomieniowego Mono w formie kodu WebAssembly oraz wszystkich wykorzystywanych standardowych bibliotek środowiska .Net jako pliki .dll. Nie ma potrzeby pobierania wszystkich tych danych przy każdorazowym ładowaniu aplikacji. Przeglądarki internetowe przechowują je w pamięci cache, co znacząco zwiększa wydajność. Zastosowanie pamięci cache w przeglądarce Google Chrome pozwoliło na zmniejszenie początkowego rozmiaru 5,9MB do 4,1KB. Aplikacja Angular przy włączonej pamięci podręcznej posiadała rozmiar 7,7KB. Zastosowanie kompresji gzip na serwerze nginx dało podobny efekt dla obydwu typów aplikacji. Rozmiar aplikacji Angular został zmniejszony o 57% natomiast aplikacji Blazor o 56%. W tabeli 2 przedstawiono rozmiary aplikacji w różnych wariantach.

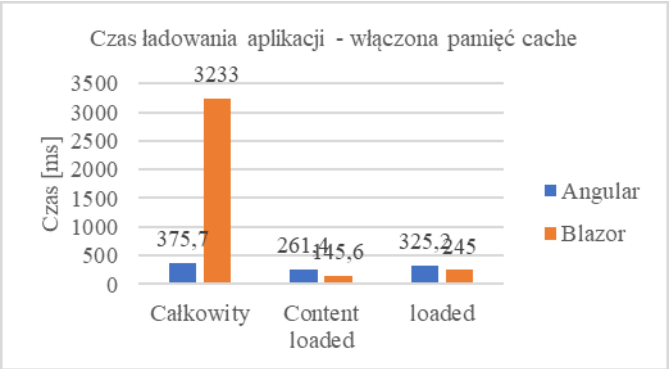
Tabela 2 Rozmiar aplikacji Blazor i Angular

Aplikacja / Rozmiar[KB]	Angular	Blazor
Całkowity	1500	5900
Włączony cache	7,7	4,1
Włączona kompresja gzip	644	2600

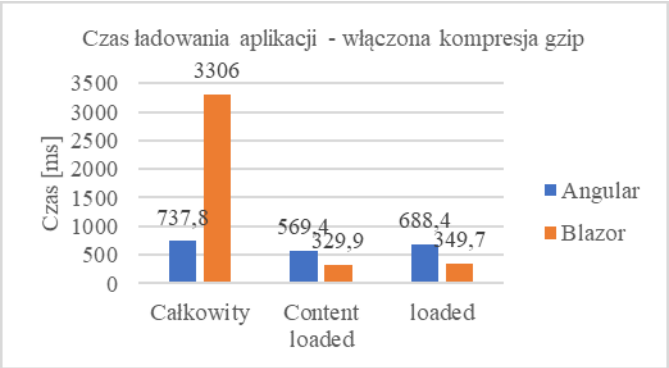
Na czas ładowania badanych aplikacji składa się ich rozmiar oraz liczba wykonanych zapytań. W obydwu kategoriach aplikacja Blazor posiada gorsze wyniki, co przekłada się na znacznie dłuższy czas ładowania. Aplikacja Blazor potrzebuje wykonać 44 zapytania, natomiast aplikacja Angular potrzebuje tych zapytań jedynie 22. Średni całkowity czas ładowania aplikacji Angular to 706.2ms, dla aplikacji Blazor jest to 3081ms. Użycie pamięci cache w przeglądarce spowodowało spadek całkowitego czasu ładowania dla aplikacji Angular a w aplikacji Blazor nie przyniosło żadnej korzyści w tym zakresie. Kompresja gzip choć zmniejszyła rozmiar przesyłanych danych w ostatecznym rezultacie spowodowała wydłużenie czasu ładowania aplikacji. Spowodowane jest to faktem, iż testy przeprowadzono na jednym stanowisku wykluczając tym samym opóźnienia spowodowane przez transfer danych przez sieć. Wykonane testy pokazały, że moment wywołania zdarzeń DOMContentLoaded oraz Load następuje wcześniej w aplikacji Blazor, w każdym z badanych scenariuszy. Rys.3, rys. 4, rys.5 zawierają wykresy kolumnowe przedstawiające dane o czasie ładowania aplikacji.



Rys. 3 Średni czas ładowania dla wariantu A

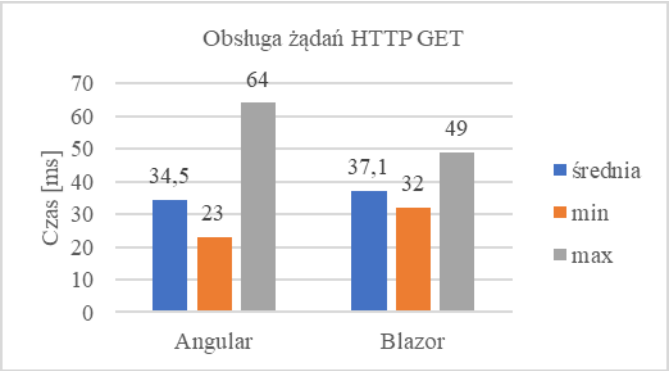


Rys. 4 Średni czas ładowania dla wariantu B

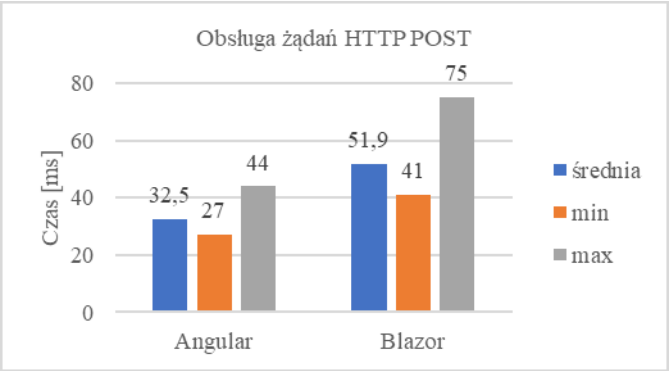


Rys. 5 Średni czas ładowania dla wariantu C

Opóźnienie obsługi żądań HTTP wykonywanych w aplikacji jest większe w przypadku szkieletu Blazor. Jest to zgodne z oczekiwaniami, ponieważ WebAssembly obecnie nie posiada możliwości bezpośredniego wykonywania operacji HTTP. Rozwiązaniem jest interakcja modułu WASM z Fetch API przeglądarki [11]. Żądania GET były obsługiwane średnio 2.6ms dłużej w aplikacji Blazor, żądania POST średnio 19.4ms dłużej. Na rys.6 i rys.7 przedstawiono wykresy średniego, minimalnego i maksymalnego czasu obsługi żądania GET i POST dal obydwu aplikacji.



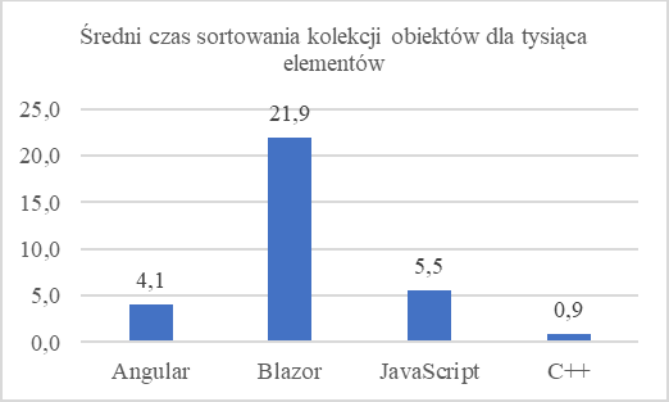
Rys. 6 Czas obsługi żądań http GET



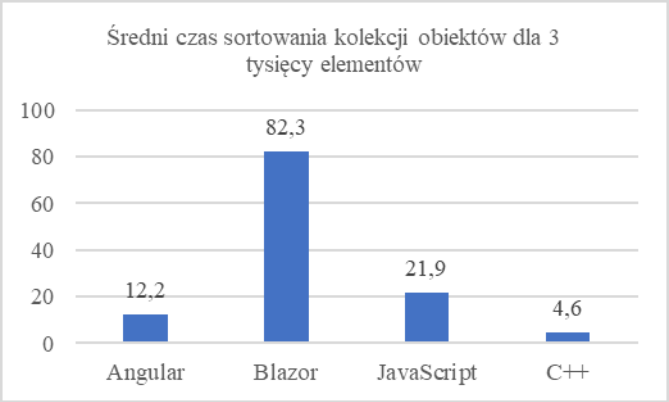
Rys. 7 Czas obsługi żądań http POST

Eksperyment sortowania n-elementowej listy obiektów został przeprowadzony w innej aplikacji niż eksperyment wydajności ładowania. Dla każdego z czterech wariantów (Angular, Blazor, JavaScript, C++) przygotowano osobną aplikację do tego celu. Uzyskane wyniki wskazują znaczną różnicę w wydajności pomiędzy kodem JavaScript a WebAssembly. Kod napisany w C++ i skompilowany do modułu WASM wykonywał operację znacznie szybciej niż analogiczny kod JavaScript. Różnica jest widoczna dla tysiąca, trzech tysięcy i dziesięciu tysięcy elementów. Można również zauważyć, że kod wykorzystany w aplikacji Angular nie traci na wydajności względem kodu napisanego jako pojedynczy skrypt. Uzyskane czasy w obydwu przypadkach są porównywalne. Aplikacja Blazor nie wykorzystuje w tym teście możliwości WebAssembly. Uzyskane w tej aplikacji czasy sortowania są dla wszystkich badanych rozmiarów kolekcji kilkukrotnie dłuższe od tych uzyskanych w aplikacji Angular. Uśrednione wartości czasu sortowania dla poszczególnych rozmiarów kolekcji przedstawiono na wykresach (rys8, rys.9, rys.10).

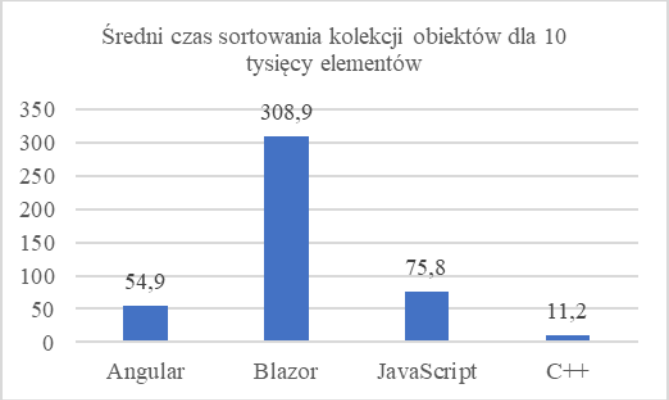
W badaniu wydajności modyfikowania elementów DOM wykorzystano aplikacje Angular i Blazor użyte w eksperymencie sortowani. Część odpowiedzialną za generowanie elementów umieszczono w osobnych komponentach. Aplikacja Blazor prezentuje gorsze wyniki od tych uzyskanych w aplikacji Angular. Czas dodania do drzewa dokumentu tysiąca elementów jest niemal sześciokrotnie dłuższy w aplikacji Blazor. Dla trzech tysięcy elementów jest już ponad sześciokrotnie dłuższy, natomiast dla dziesięciu tysięcy ponad siedmiokrotnie dłuższy. Należy zaznaczyć, że kod WebAssembly aktualnie nie ma możliwości bezpośredniego dostępu do drzewa DOM a wszystkie modyfikacje jego struktury zapisane w aplikacji Blazor są wykonywane przez wygenerowany kod JavaScript. Podobnie jak w przypadku wykonywania operacji HTTP, spodziewano się opóźnienia względem aplikacji Angular. Uśrednione wyniki z podziałem na liczbę generowanych elementów przedstawiono na rys. 11.



Rys. 8 Średni czas sortowania kolekcji tysiąca elementów



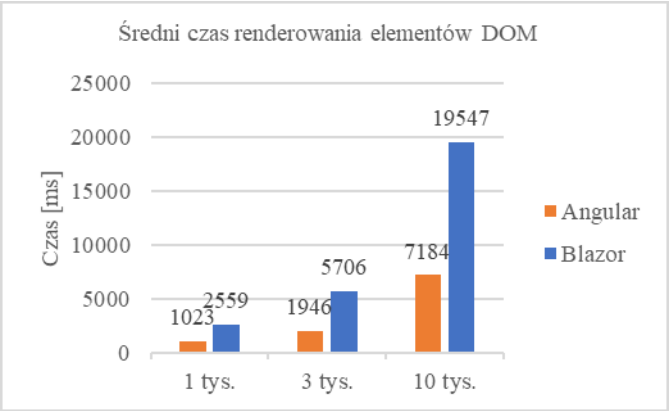
Rys. 9 Średni czas sortowania kolekcji trzech tysięcy elementów



Rys. 10 Średni czas sortowania kolekcji dziesięciu tysięcy elementów

Zbadane metryki kodu nie pozwalają na jednoznaczne wskazanie szkieletu aplikacji, który wymaga mniejszego nakładu pracy do zaimplementowania tej samej funkcjonalności. Sześć na dziesięć komponentów aplikacji Angular posiada mniejszą liczbę linii kodu niż analogiczne komponenty aplikacji Blazor. Rozpatrując oddzielnie logikę komponentu i część prezentacji można zauważyć pewne tendencje. W warstwie logiki pięć komponentów aplikacji Blazor posiada mniejszą liczbę linii kodu, w pozostałych

pięciu mniej kodu znajduje się w aplikacji Angular. W warstwie prezentacji w siedmiu komponentach kod aplikacji Angular zawierał mniej linii kodu, w jednym komponencie uzyskano dokładnie tę samą liczbę linii kodu. Można wysnuć wniosek, że składnia używana w aplikacjach Angular jest bardziej precyzyjna, a sam kod bardziej skondensowany. Uzyskane dane zebrano w tabeli 3.



Rys. 11 Średni czas renderowania elementów DOM

Tabela 3 Metryki kodu

Aplikacja / Komponent	Szablon		Logika	
	Angular	**Blazor**	**Angular**	**Blazor**
Home	60	87	45	105
Administrator	88	93	137	144
Lecture	30	44	115	97
Lecturer	145	175	218	198
Lectures	60	77	77	136
New Lecturer	52	52	97	142
Student	260	244	201	190
SignIn	26	29	87	94
SignUp	52	49	119	105
Opinions	24	34	81	47

6. Wnioski

Zbadano możliwości standardu WebAssembly jako alternatywy dla języka JavaScript w tworzeniu nowoczesnych aplikacji internetowych. Punktem wyjścia był dynamicznie rozwijany projekt Blazor. Przygotowana w oparciu o ten szkielet aplikacja wraz z analogiczną aplikację opartą o szkielet Angular i język JavaScript posłużyły do przeprowadzenia badań.

Uzyskane wyniki pozwalają wyciągnąć następujące wnioski. Szkielet aplikacji Blazor stanowi doskonały przykład wykorzystania potencjału WebAssembly. Pozwolił na stworzyć funkcjonalnej aplikacji SPA, wykorzystując inne podejście niż typowe z kodem JavaScript. Uzasadnione więc jest stwierdzenie, że standard WebAssembly może stanowić alternatywę dla języka JavaScript w tworzeniu nowoczesnych aplikacji internetowych. Jednak nie pozwala on całkowitą rezygnację z JavaScript`u. Część funkcjonalności przeglądarek internetowych może być osiągnięta wyłącznie przez JavaScript. W tych przypadkach użycie WebAssembly nie spowoduje zwiększenia wydajności, co zaobserwowano w badaniu czasu obsługi żądań HTTP oraz generowania elementów interfejsu użytkownika. Znaczną korzyść można natomiast uzyskać wykorzystując WebAssembly do zadań obliczeniowych. W eksperymencie sortowania kolekcji obiektów program napisany w C++ i skompilowany bezpośrednio do modułu WASM był kilkukrotnie szybszy od programu napisanego w JavaScript (rys.8-rys.10). Analiza wydajności ładowania wykazała dość duży rozmiar aplikacji Blazor. Jednak problem dotyczy tylko pierwszego załadowania aplikacji. Użyta pamięć cache znacząco zmniejsza rozmiar przesłanych danych.

Postawiona w artykule teza została częściowo potwierdzona. Wykorzystanie WebAssembly może w pewnych sytuacjach wpływać pozytywnie na wydajność aplikacji internetowych.

Zaprezentowany szkielet Blazor stanowi zaledwie jeden przykład implementacji standardu WebAssembly. Można oczekiwać, że pojawią się podobne rozwiązania dla innych języków programowania. W przyszłych wersjach WebAssembly ma się pojawić m.in. implementacji mechanizmu *Garbage collection* [12]. Ułatwi to stworzenie środowiska uruchomieniowego dla języków programowania wyższego poziomu takich jak *Python* czy *Go*.

Literatura

[1] 2015 Developer Survey https://insights.stackoverflow.com /survey/2015 [11.06.2019]

[2] Developer Survey Results 2016 https://insights.stackoverflow. com/survey/2016 [11.06.2019]

[3] Developer Survey Results2017 https://insights.stackoverflow. com/survey/2017 [11.06.2019]

[4] Developer Survey Results2018 https://insights.stackoverflow. com/survey/2018/ [11.06.2019]

[5] D. An, P. Meenan, Why marketers should care about mobile page speed, https://www.thinkwithgoogle.com/marketing-resources/experience-design/mobile-page-speed-load-time/ [11.06.2019]

[6] A. Jangda, A. Guha, B. Powers, E. Berger, Mind the Gap:Analyzing the Performance of WebAssembly vs. Native Code, 2019

[7] C. G. Gallant, WebAssembly in Action, Mananig Publications, 2019

[8] W. Stępniak, Z. Nowak Performance Analysis of SPA Web Systems, 2017

[9] J. Kalinowska, B. Pańczyk, Porównanie narzędzi do tworzenia aplikacji typu SPA na przykładzie Angular2 i React, 2019

[10] A. Pano, D. Graziotin Factors and actors leading to the adoption of a JavaScript framework, 2018

[11] Call a web API from ASP.NET Core Blazor, https://docs. microsoft.com/pl-pl/aspnet/core/blazor/call-web-api?view= aspnetcore-3.0 [25.08.2019]

[12] WASM Features to add after the MVPhttps://webassembly.org /docs/future-features [10.08.2019]

JOURNAL of COMPUTER SCIENCES INSTITUTE

JCSI 13 (2019) 339-344
Wysłane: 2019-09-07
Przyjęte: 2019-09-17

Analiza możliwości obrony przed atakami SQL Injection

Chrystian Byzdra*, Grzegorz Kozieł
Politechnika Lubelska, Instytut Informatyki, Nadbystrzycka 36B, 20-618 Lublin, Polska

Streszczenie. W artykule scharakteryzowano różne metody ochrony bazy danych oraz typy ataków SQL Injection. Są to wyjątkowo niebezpieczne ataki, ponieważ zagrażają poufności wrażliwych danych. W celu szczegółowej analizy metod ochrony i sposobów ataków zostały wykonane symulacje ataków i obrony w językach : C#, PHP, Java. Na podstawie wyników symulacji dla poszczególnych języków porównano skuteczność oraz wydajność metod ochrony bazy danych.

Słowa kluczowe: wstrzyknięcie kodu SQL; ochrona; sprawdzanie danych wejściowych

*Autor do korespondencji.

Adres e-mail: cbyzdra@gmail.com

Analysis of the defending possibilities against SQL Injection attacks

Chrystian Byzdra*, Grzegorz Kozieł
Institute of Computer Science, Lublin University of Technology, Nadbystrzycka 36B, 20-618 Lublin, Poland

Abstract. The article describes various protection methods of database and types of SQL Injection attacks. These are extremely dangerous attacks because they threaten the confidentiality of sensitive data. In order to analyze in detail protection methods and methods of attacks, simulations of attacks and defence were performed in the following languages: C #, PHP, Java. Based on the simulation results for particular languages, the effectiveness and efficiency of database protection methods were compared.

Keywords: SQL injection; prevention; input validation

*Corresponding author.

E-mail address: cbyzdra@gmail.com

1. Wstęp

Narzędziem umożliwiającym komunikację z bazą danych jest język SQL. Poprzez formularz w aplikacji atakujący może dodać złośliwy kod do zapytania SQL i wykonać na niej dowolne operacje. Takie działanie jest nazywanie atakiem SQL Injection. Używając tej metody atakujący jest w stanie wykorzystać lub modyfikować wrażliwe informacje. Niewystarczająca walidacja danych wejściowych umożliwia ten atak i zagraża polityce bezpieczeństwa przechowywanych danych.

Ataki SQLIA (z ang. *SQL injection attack*) są obecnie dość często wykorzystywane do pozyskiwania poufnych informacji o klientach i o poszczególnych zamówieniach z baz danych typu „*e- commerce*", także do obchodzenia mechanizmów bezpieczeństwa [1]. Typowe komunikaty o błędach SQL mogą niekiedy ułatwić atak na bazę danych, ponieważ w przypadku braku wiedzy na temat zapytania SQL lub tabel, technika wymuszania wyjątków okazuje się być skuteczna w praktyce [1]. Udoskonalanie technik programowania jest jednym ze sposobów zapobiegania podobnym atakom. Można wykorzystać do tego m. in. unikanie poszczególnych pojedynczych cudzysłowów, wprowadzić limit długości znaków wejściowych, a także uzupełniać komunikaty o różnych wyjątkach.

2. Ataki SQL Injection

Ataki Sql Injection pojawiły się wraz z pojawieniem bazy danych opartych o język SQL i aplikacji podłączonych do Internetu. Są jednym z bardziej destrukcyjnych podatności prowadzą do ekspozycji wrażliwych danych przechowywanych w aplikacjach. Jest to mocno zaakcentowane we wstępie książki J. Clarke'a [1]. Dla podkreślenia swojej tezy nawiązuje do artykułu, który został opublikowany w 1998 roku. Artykuł opisywał wyżej wymieniony atak na popularną stronę „PacketStorm". W dalszej części książki przedstawia różne sposoby ataków w celu uświadomienia czytelnika o możliwych zagrożeniach.

Temat ataków jest również często poruszany w artykułach naukowych. W swoim artykule Sadeghian [4] uświadamia czytelnika, że ten atak ma najwyższą pozycję w rankingu w podatnościach stron internetowych. Proponuje również podział tych ataków na trzy kategorie [4]:

- w paśmie - dane są pobierane tym samym kanałem, którym przeprowadzany jest atak,
- poza pasmem - pobrane dane są odsyłane do atakującego innym kanałem (np. email),
- inferencyjne – znane również jako ślepe wstrzyknięcie z ang. „*blind injection"*).

Ataki mogą zostać podzielone również pod względem celu atakującego [6]:

- tautologie,
- nielegalne lub logicznie niepoprawne zapytania,
- zapytania z użyciem słowa Union,
- zapytania Piggy-Backed,
- procedury składowane,
- wnioskowanie,
- ślepe wstrzyknięcie,
- ataki czasowe

3. Obrona przed atakiem SQL Injection

W swoje książce Clarke [1] opisuje metody ochrony przed atakami na poziomie warstwy kodu oraz określa następujące metody ochrony przed atakami:

- wyrażenia parametryzowane,
- sprawdzanie poprawności danych wejściowych,
- znak wyjścia,
- procedury składowane,
- warstwy abstrakcji

W literaturze zaproponowane są również inne podejścia do obrony przed SQLIA. Lambert [7] proponuje podzielenie zapytania spodziewanego i otrzymanego względem znaków takich jak „' ” lub „ ” oraz porównanie długości w otrzymanych tablicach. Jeżeli długości otrzymanych tablic są różne, to mechanizm stwierdza SQLIA.

Kolejny ciekawy wątek został poruszony przez Kar i Panigrahi [8]. Twórcy artykułu demonstrują technikę polegającą na transformacji zapytania z formy strukturalnej na formę parametryzowaną. W celu utworzenia pełnego zapytania, autorzy utworzyli schemat transformacji. Schemat (w postaci tabeli) został utworzony tak, aby obsłużyć dużą ilość możliwych zapytań. W celu przechowywania transformowanych zapytań i ich szybkiego wyszukiwania, postanowiono skorzystać z funkcji skrótu MD5 (z ang. *Message-Digest algorithm 5*).

Inne podejście zostało przedstawione przez Amutha Prabakar [9], czyli identyfikowanie podejrzanego kodu poprzez wyszukiwanie wzorca we wprowadzonych danych przez użytkownika. Jest wiele różnych podejść do rozpoznawania wzorca w tekście używających automatów skończonych. Algorytm użyty do wyszukiwania wzorca w tekście to Aho-Corasick.

Do ochrony przed SQLIA można użyć wcześniej wspomnianych procedur składowanych [10] (z ang. „*Stored Procedures”*). Wei przedstawia technikę obrony opartą o procedurę składowaną. Ta procedura zawiera parser dla każdego zapytania SQL wprowadzanego przez użytkownika [10]. Jeżeli zapytanie zmieni strukturę oryginalnej instrukcji to atak zostanie zidentyfikowany.

W niniejszej pracy przebadano wybrane metody obrony przed atakami SQL Injection. Należą do nich:

- wyrażenia parametryzowane,
- procedury składowane,
- funkcja skrótu – MD5,
- znak wyjścia,
- tokenizacja zapytania

4. Symulacje ataków i metod obrony

Symulacje ataków przeprowadzono z uwzględnieniem języka w jakim został zaimplementowany kod aplikacji uzyskującej dostęp do bazy danych oraz specyficznych dla nich bibliotek i wzorców programistycznych. Typ ataku wykorzystany do symulacji został oparty o tautologie.

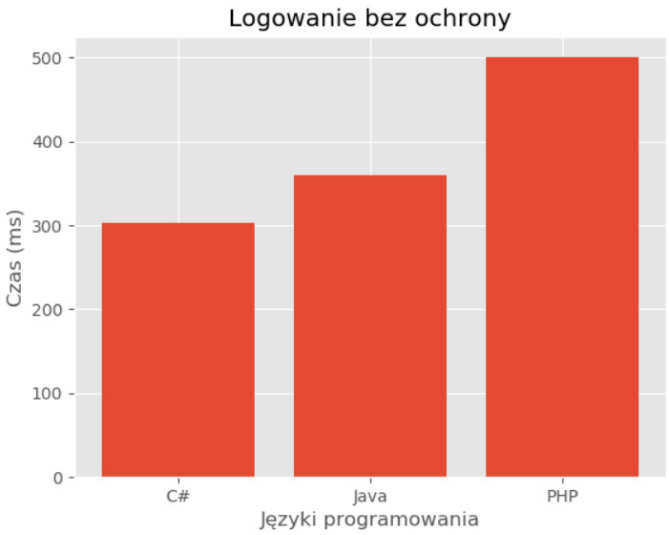
4.1. Logowanie bez ochrony

Logowanie bez ochrony zostało zrealizowane niezależnie w każdym z badanych języków programowania poprzez pobranie danych z formularza i przesłanie ich bezpośrednio do serwera bazodanowego w postaci nieparametryzowanego zapytania SQL. Przykład realizacji tej funkcji w języku C# przedstawiono na przykładzie 1.

Przykład 1. Logowanie niezabezpieczone w języku C#

```
sqlCommand.Connection = connection;
sqlCommand.CommandText = @"SELECT * FROM Users
WHERE UserLogin ='"
+ loginCredentials.email + "' AND UserPass='"
+ loginCredentials.password + "';";
SqlDataReader reader;
reader = sqlCommand.ExecuteReader();
if (reader.Read())
{ ret = true;
}
```

Badanie polegało na zmierzeniu czasu logowania we wszystkich trzech językach. Uzyskane wyniki przedstawiono na Rysunku 1.



Rys. 1. Porównanie czasów logowania bez ochrony dla 1000 prób

4.2. Wyrażenia parametryzowane

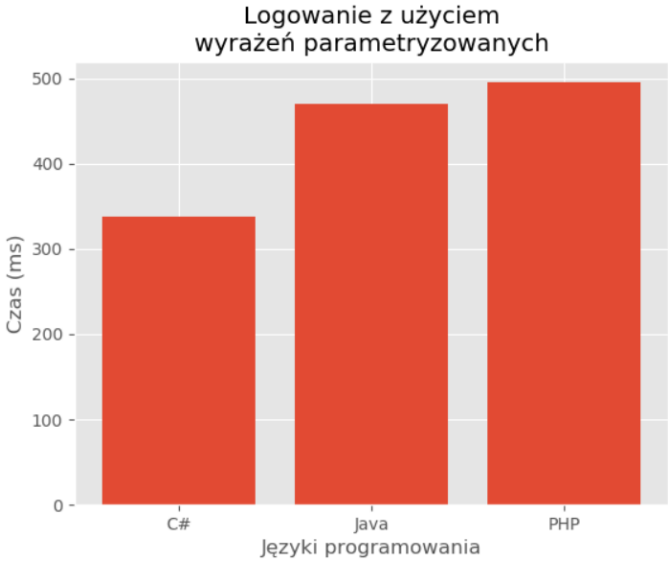
Nowoczesne języki programowania umożliwiają dostarczanie parametrów do zapytania SQL poprzez użycie

symboli zastępczych lub zmiennych wiążących, zamiast pracować bezpośrednio z danymi wejściowymi użytkownika. Powszechnie znane jako wyrażenia parametryzowane. Implementacja wyrażeń parametryzowanych w języku Java została przedstawiona na Przykładzie 2.

Przykład 2. Realizacja logowania z użyciem wyrażeń parametryzowanych w Java

```
String queryString = "SELECT * FROM Users WHERE
UserLogin =? AND UserPass=?;";
PreparedStatement statement =
con.prepareStatement(queryString);
statement.setString(1, _loginCredentials.email);
statement.setString(2, _loginCredentials.password);
ResultSet rs = statement.executeQuery();
if (rs.next()) {
ret = true;
}
```

Na Rysunku 2 zostały pokazane wyniki badań. Przedstawiono na nim wykresy czasu logowania używając procedur składowanych dla 1000 prób.



Rys. 2. Porównanie czasów logowania z użyciem wyrażeń parametryzowanych dla 1000 prób

4.3. Procedury składowane

Procedury składowane są ważną częścią współczesnej relacyjnej bazy danych. Dodają one dodatkową warstwę abstrakcji do systemu oprogramowania. Procedury składowane mogą być bardzo przydatne w ograniczaniu potencjalnej podatności na wstrzyknięcie SQL. Sposób wywołania procedury składowanej w PHP został przedstawiony na Przykładzie 3.

Przykład 3. Realizacja logowania z użyciem procedur składowanych w PHP

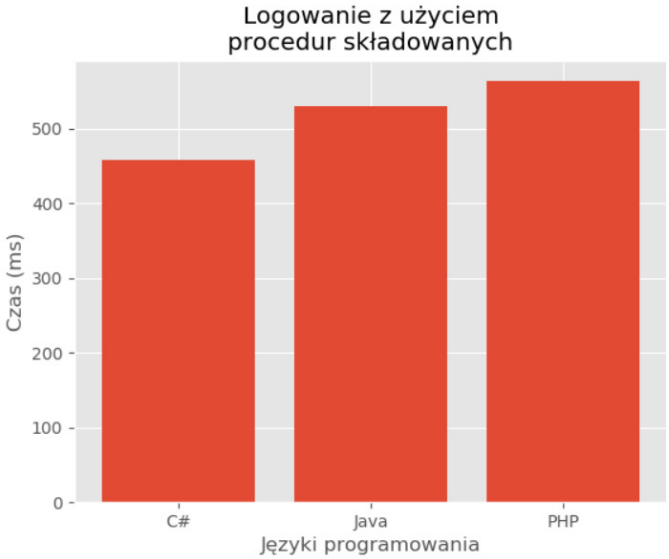
```
$stmt = $conn->prepare('{CALL loginCheck(?,?)}');
    $stmt->bindParam(1, $email, PDO::PARAM_STR |
pdo::PARAM_INPUT_OUTPUT, 50);
$stmt->bindParam(2, $pass, PDO::PARAM_STR |
pdo::PARAM_INPUT_OUTPUT, 50);
$result = $stmt->execute();
$row = $stmt->fetch( PDO::FETCH_ASSOC );
if($row)
{
return true;
}
```

W Przykładzie 3 jest zawarta instrukcja wywołująca procedurę składowaną, która jest widoczna na Przykładzie 4.

Przykład 4. Procedura składowana **LoginCheck**

```
CREATE PROCEDURE [dbo].[LoginCheck]
@login varchar(50) output,
@pwd varchar(50) output
AS
select * from Users
    where UserLogin = @login and UserPass = @pwd
return
```

Rysunek 3 przedstawia wyniki zrealizowanych symulacji dla logowania z użyciem procedur składowanych. Na podstawie wykresów można wywnioskować, że dłużej wykonuje się logowanie z użyciem procedur składowanych niż przy pomocy wyrażeń parametryzowanych.



Rys. 3. Porównanie czasów logowania z użyciem procedur składowanych dla 1000 prób

4.4. Funkcja skrótu MD5

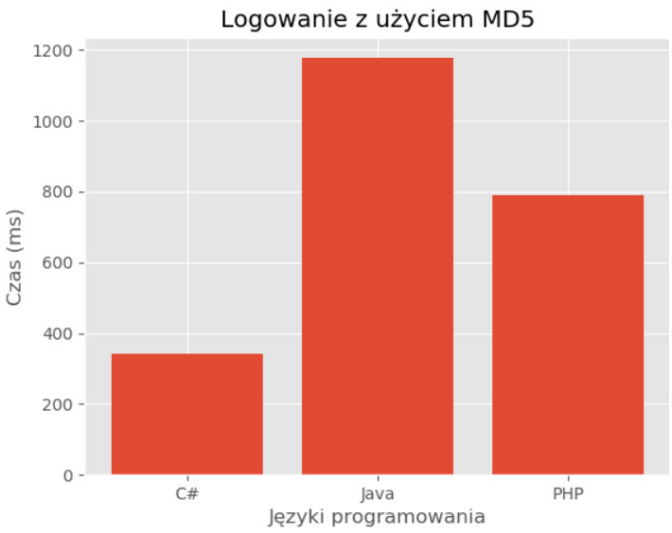
Wiele zapytań można zredukować do tej samej struktury [8]. Forma strukturalna wstrzykniętego zapytania różni się zasadniczo od poprawnej kwerendy, więc można ją łatwo określić jako SQLIA. Redukując zapytania do tej samej formy strukturalnej minimalizowany jest rozmiar przestrzeni wyszukiwania dla dopasowania w czasie wykonywania. W celu przyspieszenia wyszukiwania formę strukturalną zapytania przedstawia się w postaci wyniku funkcji skrótu MD5. Na Przykładzie 5 została przedstawiona zmodyfikowana wersja transformacji zapytania i kodowania, która ogranicza się do obliczenia MD5 dla danych

wejściowych i porównaniu wyniku z obliczonymi wartościami przechowywanymi w bazie danych.

Przykład 5. Realizacja logowania z użyciem MD5 w PHP

```
$hashedLogin = md5($email);
$hashedPass = md5($pass);
$tsql = "select * from Users as u where
CONVERT(VARCHAR(32), HashBytes('MD5', u.UserLogin), 2)
='".$hashedLogin."'and CONVERT(VARCHAR(32),
HashBytes('MD5', u.UserPass), 2) ='" .$hashedPass."';";
$getResults = $conn->prepare($tsql);
$getResults->execute();
$results = $getResults->fetchAll(PDO::FETCH_BOTH);
if($results)
{
return true;
}
```

Na Rysunku 4 przedstawione zostały wyniki czasów logowania z wykorzystaniem funkcji MD5 w trzech językach. W tej symulacji najdłuższy czas wykonania miała metoda napisane w języku Java. Na czas realizacji funkcji ma wpływ szybkość obliczenia wyniku funkcji skrótu.



Rys. 4. Porównanie czasów logowania z użyciem MD5 dla 1000 prób

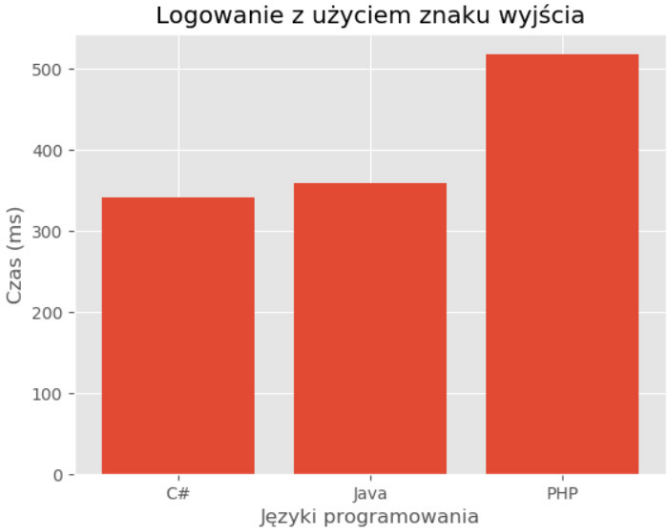
4.5. Znak wyjścia

Oprócz sprawdzania poprawności danych wejściowych odbieranych przez aplikację często konieczne jest również zakodowanie tego, co jest przekazywane między różnymi modułami lub częściami aplikacji. W SQL jest używany pojedynczy cudzysłów jako koniec dla wprowadzonej treści przez użytkownika. Wprowadzenie napisu posiadającego pojedynczy cudzysłów jest problematyczne. Rozwiązaniem tego problemu i jednocześnie zabezpieczeniem przed SQLIA jest zakodowanie napisu poprzez dodaniu znaku wyjścia. Na Przykładzie 6 została zaprezentowana przykładowa metoda napisana w języku C#, która zamienia pojedynczy cudzysłów na podwójny.

Przykład 6. Logowanie z użyciem znaku wyjścia w C#

```
string preparedEmail, preparedPass;
preparedEmail = loginCredentials.email.Replace("'", "''");
preparedPass = loginCredentials.password.Replace("'", "''");
SqlCommand sqlCommand = new SqlCommand();
sqlCommand.Connection = connection;
sqlCommand.CommandText = @"SELECT * FROM Users
WHERE UserLogin ='"
+ preparedEmail + "' AND UserPass='" + preparedPass +
"';";
SqlDataReader reader;
reader = sqlCommand.ExecuteReader();
if (reader.Read())
{
ret = true;
}
```

Na Rysunku 5 zostały przedstawione wyniki symulacji logowania z użyciem znaku wyjścia dla wybranych języków programowania. Wskazane czasy na wykresie są bardzo zbliżone do czasów logowania bez metody ochrony.



Rys. 5. Porównanie czasów logowania z użyciem znaku wyjścia dla 1000 prób

4.6. Tokenizacja zapytania

Metoda polegająca na dzieleniu wprowadzonego zapytania względem ustalonych tokenów, Najczęściej wybieranymi tokenami są pojedynczy cudzysłów oraz spacja. Po podzieleniu napisu i wprowadzeniu wartości do tablicy lub listy, można porównywać właściwości tych struktur danych z właściwościami oczekiwanymi. Na Przykładzie 7 została przedstawiona realizacja tej metody w języki Java.

Przykład 7. Realizacja logowania z użyciem tokenizacji zapytania w języku Java

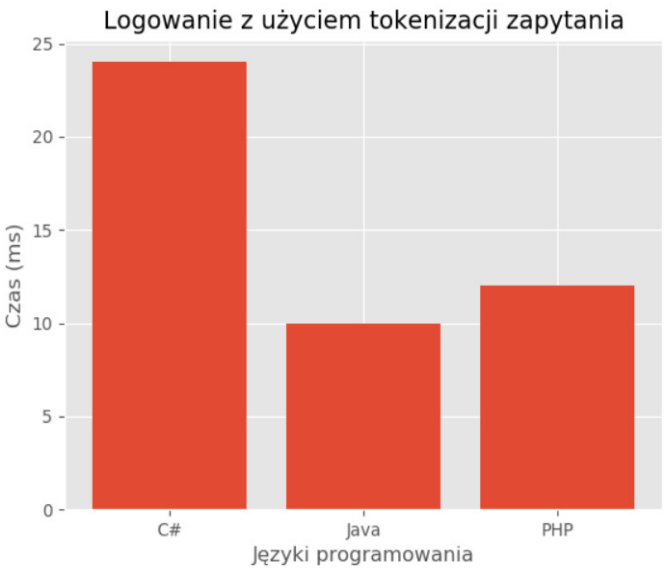
```
String queryToCmp = "SELECT * FROM Users WHERE
UserLogin ='' AND UserPass='';";
String insertedQuery = "SELECT * FROM Users WHERE
UserLogin ='"
+ _loginCredentials.email + "' AND UserPass='"
+ _loginCredentials.password + "';";
if (LoginCredentials.validateByQueryToken(insertedQuery,
queryToCmp))
```

Sprawdzenie poprawności zapytania jest realizowane przed nawiązaniem połączenia z bazą danych. Implementacja metody walidującej zapytanie została pokazana na Przykładzie 8.

Przykład 8. Metoda sprawdzająca wprowadzone dane

```
tokensToCheck.add(' ');
tokensToCheck.add(';');
for (Character item : tokensToCheck) {
tokensInsQuery = _insertedQuery.split(item.toString());
tokensQueryToCmp = _queryToCmp.split(item.toString());
if (tokensInsQuery.length != tokensQueryToCmp.length) {
ret = false;
break;
```

Na Rysunku 6 zostały zaprezentowane wyniki symulacji. Najniższy czas wykonania miała funkcja zaimplementowana w języku Java. Czas wykonania metody jest związany z bardzo szybkim dostępem do struktur danych w tym języku. Na wynik nie miała wpływu implementacja sterownika, ponieważ system w przypadku wykrycia SQLIA nie uruchamiał procesu połączenia z bazą.

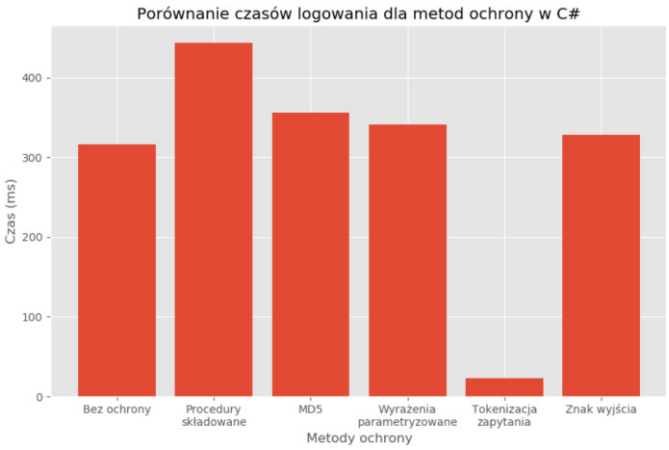


Rys. 6. Porównanie czasów logowania z użyciem tokenizacji zapytania dla 1000 prób

5. Wnioski

Celem pracy było porównanie metod ochrony przed atakiem SQL Injection. Wykonano programy, które symulowały obronę przed wstrzykniętym kodem SQL. Porównanie czasów wykonania symulacji w języku C# dla 1000 prób zostało zaprezentowane na Rysunku 7.

Metody obrony oparte na analizie wprowadzonego tekstu (np. tokenizacja zapytania) są zdecydowanie najszybsze, co wynika z faktu, że w przypadku wykrycia złośliwego kodu SQL nie zostaje nawiązywane połączenie z bazą danych. Jednak tokenizacja zapytania nie jest w pełni bezpieczna. W przypadku wprowadzenia danych w zakodowanej postaci (np. heksadecymalnej) funkcja nie wykryje ataku. Dosyć szybkim sposobem ochrony okazała się metoda dodająca znak wyjścia do zapytania. Jest to bezpieczne rozwiązanie, aczkolwiek należy pamiętać, że dla niektórych danych takie zapytania mogą być mało czytelne. Najmniej wydajną metodę stanowiła oparta na funkcji skrótu MD5, jednakże jest ona jednocześnie bardzo bezpieczna dla wybierania informacji, ponieważ wszystkie wprowadzone dane zostają kodowane. W rozbudowanych zapytaniach w aplikacjach biznesowych, obliczanie funkcji skrótu dla każdej wprowadzonej danej może być problematyczne.



Rys. 7. Porównanie czasów logowania dla metod ochrony w C#

Rozwiązanie zabezpieczające przed wstrzyknięciem złośliwego kodu to także procedury składowane. W przypadku poprawnego zaimplementowania są bezpieczne, szybkie oraz mogą zostać ponownie wykorzystane w innych miejscach w aplikacji. Jako wadę takiego rozwiązania należy wskazać fakt, że w przypadku zmiany bazy danych, wszystkie procedury musiałyby zostać na nowo zaimplementowane.

Następną metodą, która posiada wiele zalet jest oparta na wyrażeniach parametryzowanych. Działa ona szybko, implementowana zostaje w kodzie niezależnie od bazy danych i zachowuje bezpieczeństwo danych. Wbudowane mechanizmy, napisane w danym języku zapewniają wprowadzenie informacji do zapytania tylko w postaci napisu.

Literatura

[1] Clarke J.: SQL Injection Attacks and Defense, Syngress Publishing, 2009

[2] Somesh J., Christodorescu M., Wang C., Maughan D., Song D.: Malware Detection, Springer, 2006

[3] Snyder C., Southwell M.: Pro PHP Security, Apress, 2005

[4] Sadeghian A., Zamani M., Ibrahim S.: SQL Injection is Still Alive:A Study on SQL Injection Signature Evasion Techniques, IEEE, 2013

[5] Heydari M.Z.: Comparision of SQL Injection Detection and Prevention Techniques, ICETC, 2010

[6] Halfond W.G.J., Viegas J., Orso A.: A Classification of SQL Injection Attacks and Countermeasures, IEEE, 2006

[7] Lambert N., Song Lin K.: Use of Query Tokenization to detect and prevent SQL Injection Attacks, IEEE, 2010

[8] Kar D., Panigrahi S.: Prevention of SQL Injection Attack Using Query Transformation and Hashing, IEEE, 2012

[9] Amutha Prabakar M., KarthiKeyan M., Marimuthu K.: An efficient technique for preventing SQL Injection attack using pattern matching algorithm, IEEE, 2013

[10] Wei K., Muthuprasanna M., Kothari S.: Preventing SQL Injection Attacks in Stored Procedures, IEEE, 2006

[11] Specyfikacja języka C# http://docs.microsoft.com/pl-pl/dotnet/csharp/language-reference/language-specification/introduction [20.05.2019]

[12] Podstawy programowania w języku Java, https://docs.oracle.com/javase/tutorial/java/index.html [13.05.2019]

[13] Dokumentacja techniczna języka PHP, https://www.php.net/manual/en/ [11.04.2019]

[14] Opis standardów i składni języka SQL, http://bazy.rzeszow.pl/klasy/klasa3bazy/sql.pdf [15.05.2019]

[15] Wykład z języka SQL przedstawiający podstawowe funkcje, https://www.mechanikryki.pl/renata/pliki_pdf/SQL.pdf [15.05.2019]

JOURNAL of COMPUTER SCIENCES INSTITUTE

JCSI 13 (2019) 345-348
Wysłane: 2019-09-10
Przyjęte: 2019-09-20

Porównanie wydajności trójwymiarowych gier z użyciem silników CryEngine i Unity

Hubert Żukowski*
Politechnika Lubelska, Instytut Informatyki, Nadbystrzycka 36B, 20-618 Lublin, Polska

Streszczenie. W artykule przedstawiono wyniki badań wydajności aplikacji utworzonych na silnikach gier wideo: Unity i CryEngine. Przeprowadzone badania skupiały się na zestawieniu dwóch aplikacji utworzonych na wskazanych silnikach. Wybrano parametry, dla których zostało wykonane porównanie, mianowicie liczbę klatek na sekundę, wykorzystanie procesora, zużycie pamięci RAM, a także czas generowania obiektów 3D przez aplikacje. Wykonane aplikacje korzystały z tych samych zasobów i zbliżonego kodu. Postawiona w artykule hipoteza – Unity jest silnikiem wydajniejszym niż CryEngine – została zweryfikowana i częściowo udowodniona: oba silniki są wydajniejsze w zależności od sposobu wykorzystania.

Słowa kluczowe: unity; cryengine; silnik gier; wydajność

*Autor do korespondencji.

Adres e-mail: h.zukowski@outlook.com

Comparison of 3D games' efficiency with use of CRYENGINE and Unity game engines

Hubert Żukowski*
Institute of Computer Science, Lublin University of Technology, Nadbystrzycka 36B, 20-618 Lublin, Poland

Abstract. This article presents the results of the performance studies of applications created with use of CryEngine and Unity game engines. Presented research was mainly focused on comparison of two applications created on selected engines. Several parameters were selected for the research: frame rate, CPU usage, RAM usage and generation time of 3D objects. Created applications were built with the use of the same graphic resources and similar source code. The hypotheses set in the article – Unity game engine is more efficient than CryEngine – have been verified and partially confirmed: engines are more efficient in different environments.

Keywords: unity; cryengine; game engine; performance

*Corresponding author.

E-mail address: h.zukowski@outlook.com

1. Wstęp

Branża gier wideo jest jedną z najszybciej rozwijających się dziedzin z pogranicza informatyki. Od 2017 nieprzerwanie dominuje nad rynkiem muzycznym, filmowym, a także książkowym [11]. W roku 2018 wartość rynku gier wideo wzrosła niemal o 27% do wartości 135 mld dolarów [1].

Koncerny specjalizujące się w produkcji gier odnoszą sukcesy od początku lat 70, a roczna wartość ich przychodów szacowana jest na ponad 100 mld dolarów [13]. Nie może więc dziwić fakt, że każdego roku pojawia się coraz więcej firm zajmujących się tworzeniem tego typu oprogramowania.

Jednym z najpopularniejszych i najszerszych podziałów gier wideo jest pogrupowanie ze względu na platformy, na których są dostępne. Wobec tego najczęściej mówa o grach PC, grach mobilnych oraz grach konsolowych [5].

Gry wideo są interaktywnym oprogramowaniem komputerowym. Ich celem jest najczęściej rozrywka bądź edukacja wymagająca od użytkownika rozwiązywania zadań logicznych i/lub zręcznościowych [5], a ich początki sięgają końca pierwszej połowy XX wieku. Uruchomienie gry wiąże się z wykorzystaniem zasobów komputera. Nowoczesne gry wymagają coraz potężniejszych kart graficznych, procesorów, czy też większej ilości pamięci RAM, przez co rynek gier wideo napędza także producentów podzespołów, czy samych komputerów, laptopów, konsol i smartfonów.

Dzisiejsze gry są tworzone za pomocą silników gier [5] projektowanych w oparciu o sprecyzowany paradygmat obiektowy i wykorzystywanych przez przedsiębiorstwa, które produkują gry wideo zarówno na komputery personalne, smartfony, jak i konsole. Aktualne rozwiązania w dziedzinie produkcji gier opierają się o dwa typy silników: tworzone od podstaw przez wytwórnię gier oraz wykorzystanie istniejących rozwiązań, wśród których najpopularniejsze są Unreal Engine, Unity oraz CryEngine [10]. Silniki te oferują pełny zestaw funkcji niezbędnych do zbudowania interaktywnej aplikacji 3D, tj.:

- silnik renderowania – moduł graficzny pozwalający na tworzenie obiektów 3D oraz efektów wizualnych;
- silnik fizyki – moduł odpowiedzialny za zjawiska fizyczne występujące w grze: kolizje, grawitację oraz reakcje obiektów 3D;
- silnik audio – umożliwiający odtwarzanie efektów dźwiękowych oraz muzyki;

- silnik sieciowy – moduł pozwalający na zaprogramowanie trybów wieloosobowych rozgrywanych w czasie rzeczywistym poprzez wymianę danych pomiędzy urządzeniami graczy.

W badaniach przeprowadzonych na platformie Steam najczęściej wykorzystywanym silnikiem gier jest Unreal Engine, Unity, id Tech, Source oraz CryEngine (tabela 1.). Badania te dotyczą gier VR, 3D oraz 2D, czyli wszystkich dostępnych rodzajów na platformie. W badaniu wzięto pod uwagę ponad 50 tys. tytułów [6].

Tabela 1. Wykorzystanie silników gier na platformie Steam [6]

Silnik gier	Wykorzystanie
Unreal Engine	23%
Unity	11%
id Tech	6%
Source	4%
CryEngine	3%
Inne (GameMaker, Fox, Frostbite...)	53%

Najpopularniejsze z wykorzystywanych silników umożliwiają produkcję gier na wiele platform jednocześnie. Dzięki temu producenci mogą zaoszczędzić na przenoszeniu gier na inne platformy.

W niniejszej pracy postawiona została następująca hipoteza: silnik Unity jest wydajniejszy niż CryEngine.

Do potwierdzenia przedstawionej tezy przeprowadzona została analiza porównawcza aplikacji wykonanych na obu silnikach. Zestawione gry testowe będą wykorzystywać takie same modele 3D oraz skrypty. Silniki Unity i CryEngine wybrane zostały z powodu podobieństwa wpieranych języków i platform (tabela 2.). Oba pozwalają na programowanie w języku C#, a wykonane z ich pomocą aplikacje działają na systemie Windows 10. Kolejnym podobieństwem jest wspieranie gier 3D oraz aplikacji Virtual Reality. Powodem wyboru CryEngine – jako konkurenta Unity – jest brak badań wydajnościowych tego silnika.

Tabela 2. Porównanie cech wybranych silników [3] [4]

Cecha	CryEngine	Unity
Wspierane platformy	Oculus Rift, Windows PC, Xbox One, Playstation 4	iOs, Android, Windows, Mac, Linux, WebGL, Playstation 4, Xbox One, Nintendo 3DS, Oculus Rift, Nintendo Switch
Silnik fizyki	CryPhysics	PhysX
Wspierany język	C#, C++	C#, UnityScript, JavaScript, Boo
Gry 2D	Nie	Tak
Gry 3D	Tak	Tak
VR	Tak	Tak

2. Przegląd literatury

Dostępne materiały badawcze przedstawiają silniki gier wideo nie tylko jako narzędzia do wytwarzania gier, lecz także wskazują na ich użyteczność w środowiskach naukowych. Silniki gier wykorzystuje się w dziedzinach z pogranicza medycyny, architektury [14], meteorologii [8], geologii [2] oraz edukacji [12].

Znaczna część publikacji skupia się na rozwoju silników gier wideo, sięgając do ich historii, niemniej w związku z ciągłymi postępami dziedziny rzadko można spotkać aktualną publikację dotyczącą aktywnie wykorzystywanych silników na obecnym rynku. Podobnie jest z porównywaniem silników.

Na podstawie pracy I. Pachoulakis i G. Pontikakis można dowiedzieć się o sposobie porównania gier bazujących na silnikach Unity i Unreal Engine [9], lecz brakuje podobnych badań dla CryEngine. W pracy [7] poruszono problem porównywania różnych silników, co umożliwiło utworzenie metody badania w niniejszym artykule.

Wobec powyższych argumentów wysunąć można wniosek, że badanie możliwości silników gier wideo ogranicza się do porównania ich zastosowań, a w nielicznych badaniach skupia się na ewaluacji ich wydajności, biorąc się pod uwagę gry różnych wydawców (opracowane przy odmiennych zasobach).

Niniejsza praca porusza więc problem porównywania silników gier wideo poprzez utworzenie – na wybranych środowiskach – aplikacji korzystających z takich samych zasobów (modeli 3D i skryptów).

3. Metoda badań

Na potrzeby badań utworzono dwie aplikacje w oparciu o silniki CryEngine i Unity. Następnie w obu grach odwzorowano takie same poziomy rozgrywki oraz w miarę możliwości utworzono podobne skrypty zarządzające generowaniem obiektów.

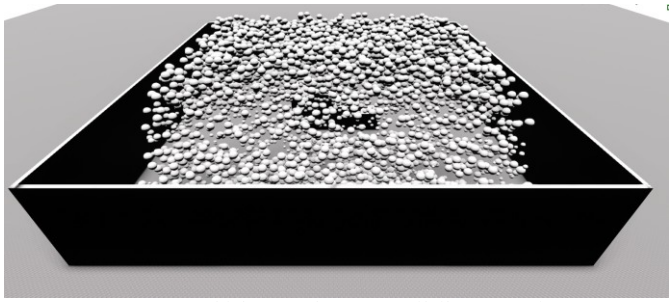
Stworzone w ten sposób aplikacje pozwoliły na stworzenie danej liczby obiektów, a w konsekwencji obciążenie aplikacji i stanowiska badawczego (tabela 3.), a także uzyskanie parametrów pamięci RAM, maksymalnego wykorzystania CPU, klatek na sekundę oraz czasu generowania obiektów przez silnik.

Tabela 3. Parametry stanowiska badawczego

Element	Wersja
System	Windows 10 Home Edition
Procesor	Intel Core i7 4710HQ
Pamięć RAM	8 GB
GPU	GeForce GTX860M
Dysk	SSD, Crucial BX500
Rozdzielczość	1920*1080

Wykonane aplikacje (rys 1.) pozwoliły na generowanie określonej liczby obiektów. Na potrzeby badań wybrano cztery poziomy obciążeń silników: 1000, 5000, 7500 oraz

10000 obiektów. Odczytów wybranych parametrów dokonano po 60 sekundach od wygenerowania wszystkich obiektów. Na wszystkich poziomach testów pomiary powtórzono trzykrotnie dla każdej aplikacji.



Rys. 1. Generator obiektów wykonany na silniku CryEngine

Obiektom generowanym przez aplikację nadano właściwości fizyczne: masę, podatność na grawitację oraz kolizje. Każdy obiekt rozmieszczono losowo (pod względem wysokości i położenia) w obrębie stworzonej sceny 3D.

4. Wyniki badań

Dla pierwszego testu (tabela 4.) przy minimalnym przyjętym obciążeniu – 1000 obiektów – lepsze rezultaty osiągała aplikacja zbudowana na Unity.

Tabela 4. Wyniki pomiarów dla obciążenia 1000 obiektów

CryEngine				
	FPS	CPU	RAM [MB]	Czas gen. [ms]
Badanie 1.	55	55,63%	687,3	104
Badanie 2.	53	55,67%	688,1	122
Badanie 3.	49	53%	687	101
Średnia	52,34	54,77%	687,47	109
Odchyl. std.	3,06	1,5%	0,57	11,36
Unity				
	FPS	CPU	RAM [MB]	Czas gen. [ms]
Badanie 1.	60	19,36%	243,7	53
Badanie 2.	60	19,96%	243,6	81
Badanie 3.	60	37,13%	243,6	87
Średnia	60	25,48%	243,63	73,67
Odchyl. std.	0	10,1%	0,06	18,15

W drugim teście (tabela 5.), dla 5000 wygenerowanych obiektów, rezultaty obu silników zbliżyły się, choć Unity osiągnęło lepsze wyniki w każdym z odczytywanych parametrów.

Wyniki dla trzeciego poziomu (7500 obiektów) badań zaprezentowano w tabeli 6. Na ich podstawie można stwierdzić, że silnik Unity gorzej poradził sobie tylko w przypadku liczby wyświetlanych klatek na sekundę. CryEngine osiągnął słabsze rezultaty w zestawieniu użycia pamięci RAM oraz czasu generowania obiektów.

Przy maksymalnym przyjętym obciążeniu (10000 obiektów) liczba klatek na sekundę w CryEngine nie spadła poniżej granicy płynności obrazu (30 FPS), ale silnik fizyczny CryPhysics nie był w stanie obsłużyć tak dużej liczby obiektów. Przeciążenie silnika fizyki nie miało wpływu na renderowaną grafikę, a jego zachowanie nie było obiektem badań, wobec czego badanie mogło być kontynuowane. W Unity wartości FPS wyniosły średnio 7 klatek na sekundę. Obciążenie CPU było zbliżone w przypadku obu aplikacji. Wartości czasu generowania obiektów oraz wykorzystania pamięci RAM przemawiają zdecydowanie na korzyść Unity. Wyniki przedstawiono w tabeli 7.

Tabela 5. Wyniki pomiarów dla obciążenia 5000 obiektów

CryEngine				
	FPS	CPU	RAM [MB]	Czas gen. [ms]
Badanie 1.	41	55,28%	688	385
Badanie 2.	43	55,30%	687	339
Badanie 3.	42	55,80%	687,4	360
Średnia	42	55,46%	687,47	361,34
Odchyl. std.	1	0,3%	0,5	23,03
Unity				
	FPS	CPU	RAM [MB]	Czas gen. [ms]
Badanie 1.	43	46,80%	243,5	157
Badanie 2.	39	46,40%	243,6	129
Badanie 3.	45	46,71%	243,6	149
Średnia	42,34	46,64%	243,57	145
Odchyl. std.	3,06	0,21%	0,06	14,42

Tabela 6. Wyniki pomiarów dla obciążenia 7500 obiektów

CryEngine				
	FPS	CPU	RAM [MB]	Czas gen. [ms]
Badanie 1.	34	55,46%	687,9	518
Badanie 2.	37	52,66%	687	593
Badanie 3.	37	57,29%	687	673
Średnia	36	55,14%	687,3	594,67
Odchyl. std.	1,73	2,3%	0,52	77,51
Unity				
	FPS	CPU	RAM [MB]	Czas gen. [ms]
Badanie 1.	17	51,97%	243,9	207
Badanie 2.	16	51,97%	243,9	168
Badanie 3.	17	52,23%	243,5	189
Średnia	16,67	52,06%	243,77	188
Odchyl. std.	0,58	1,5%	0,23	19,52

Tabela 7. Wyniki pomiarów dla obciążenia 10000 obiektów

CryEngine				
	FPS	CPU	RAM [MB]	Czas gen. [ms]
Badanie 1.	31	55,91%	687,9	698
Badanie 2.	28	55,64%	687,4	653
Badanie 3.	33	55,52%	687,5	802
Średnia	30,67	55,69%	687,6	717,67
Odchyl. std.	2,52	0,2%	0,26	76,42
Unity				
	FPS	CPU	RAM [MB]	Czas gen. [ms]
Badanie 1.	7	54,03%	243,9	207
Badanie 2.	7	55,31%	243,6	196
Badanie 3.	8	54,06%	243,8	205
Średnia	7,34	54,47%	243,77	202,67
Odchyl. std.	0,58	0,7%	0,15	5,86

5. Dyskusja wyników

Przy minimalnym i średnim obciążeniu wygenerowanymi obiektami (1000 i 5000) silnik Unity wypadł lepiej we wszystkich parametrach. W każdym z przeprowadzonych testów CryEngine potrzebował ponad 440 MB pamięci RAM więcej, ponadto czas generowania obiektów jest niższy w Unity.

W testach o wyższej liczbie wygenerowanych obiektów CryEngine osiągnął lepsze rezultaty tylko w parametrze klatek na sekundę. Wyniki te nie odzwierciedlają jednak rzeczywistej sprawności silnika, który przy maksymalnym obciążeniu nie radził sobie z obsługą fizyki wszystkich obiektów.

6. Wnioski

Przeprowadzone badania pozwoliły na sformułowanie następujących wniosków:

- Unity potrzebuje mniejszej ilości zasobów do sprawnego działania aplikacji;
- niezależnie od liczby obiektów w kadrze kamery wymagania względem zasobów silnika CryEngine są zbliżone;
- silnik fizyki CryPhysics nie radzi sobie z dużą liczbą obiektów fizycznych.

Wobec wyników badań potwierdzona zostaje teza postawiona we wcześniejszej części artykułu. Silnik Unity jest wydajniejszy w większości testów i bezkonkurencyjny podczas badań na mniejszym obciążeniu aplikacji. Unity w większym stopniu reaguje na zmiany zachodzące w poziomie gry i dostosowuje obciążenie procesora do konkretnych potrzeb, podczas gdy CryEngine stale wymaga takich samych zasobów.

Literatura

[1] Batchelor J., Global games market value rising to $134.9bn in 2018, 2018, https://www.gamesindustry.biz/articles/2018-12-18-global-games-market-value-rose-to-usd134-9bn-in-2018 [10.06.2019]

[2] Bellanca J. L., Developing a Virtual Reality Environment for Mining Research, 2019, Mining, Metallurgy & Exploration

[3] Cechy Unity, https://unity3d.com/unity [29-06-2019]

[4] CRYENGINE Feature Index, https://www.cryengine.com/features [03-03-2019]

[5] Gra komputerowa, https://pl.wikipedia.org/wiki/Gra_komputerowa [10.06.2019]

[6] I researched the market share of game engines on Steam, from over 60,000 Steam games., https://www.reddit.com/r/gamedev/comments/8s20qp/i_researched_the_market_share_of_game_engines_on/ [04.05.2019]

[7] Mishra P., Shrawankar U., Comparison between Famous Game Engines and Eminent Games, 2016 International Journal of Interactive Multimedia and Artificial Intelligence

[8] Nowak Ł., Bąk A., Czajkowski T., Wojciechowski K., Modeling and Rendering of Volumetric Clouds in Real-Time with Unreal Engine 4, Computer Vision and Graphics

[9] Pachoulakis I., Pontikakis G., Combining features of the Unreal and Unity Game Engines to hone development skills, 2015 9th International Conference on New Horizons in Industry, Business and Education

[10] Popularne silniki graficzne - którego warto się nauczyć?, https://www.komputerswiat.pl/gamezilla/artykuly/popularne-silniki-graficzne-ktorego-warto-sie-nauczyc/34fhly1 [12.06.2019]

[11] Radzewicz S., Ubisoft przypomina, że rynek gier już dawno zjadł filmy, muzykę i książki. Infografika nie pozostawia złudzeń, https://www.spidersweb.pl/rozrywka/2018/02/13/rynek-gier-wideo-2017/ [1.09.2019]

[12] Salomão A., Andaló F., Vieira M. L. H., How Popular Game Engine Is Helping Improving Academic Research: The DesignLab Case, AHFE 2018: Advances in Human Factors in Wearable Technologies and Game Design

[13] Świat kupuje coraz więcej gier komputerowych. Sprzedaż sięgnie 98 mld dol. Rocznie, https://www.forbes.pl/wiadomosci/swiat-kupuje-coraz-wiecej-gier-komputerowych-sprzedaz-siegnie-98-mld-dol-rocznie/v0t1s56 [10.06.2019]

[14] Wang S., Mao Z., Zeng Ch., Gong H., Li S., Chen B., A New Method of Virtual Reality Based on Unity3D, 2010 18th International Conference on Geoinformatics

JOURNAL of COMPUTER SCIENCES INSTITUTE

JCSI 13 (2019) 349-353
Wysłane: 2019-09-14
Przyjęte: 2019-09-22

Badanie wzorca architektonicznego Entity-component-system zaprojektowanego z wykorzystaniem techniki Data-oriented design

Dawid Masiukiewicz*, Daniel Masiukiewicz*, Jakub Smołka
Politechnika Lubelska, Instytut Informatyki, Nadbystrzycka 36B, 20-618 Lublin, Polska

Streszczenie. Celem niniejszego artykułu jest prezentacja i przetestowanie architektury Entity-component-system zaprojektowanej w oparciu o dane. Rozwiązanie pozwala usprawnić proces tworzenia aplikacji jednocześnie zwiększając jej wydajność. Do badań przygotowana została aplikacja testowa w oparciu o autorskie rozwiązania. Na łamach artykułu przedstawiono porównanie badanych technik z programowaniem zorientowanym obiektowo.

Słowa kluczowe: tworzenie gier; DOD; ECS

*Autor do korespondencji.

Adresy e-mail: wismerchil@interia.eu, danielmz25@gmail.com

Research of an Entity-component-system architectural pattern designed with using of Data-oriented design technique

Dawid Masiukiewicz*, Daniel Masiukiewicz*, Jakub Smołka
Institute of Computer Science, Lublin University of Technology, Nadbystrzycka 36B, 20-618 Lublin, Poland

Abstract. The purpose of this article is to present and evaluate Entity-component-system architecture designed based on data. The solution allows for improving application development process and increasing its efficiency. A test application was prepared for research using custom solutions. Evaluated techniques was compared with object-oriented programming in the article.

Keywords: game development; DOD; ECS

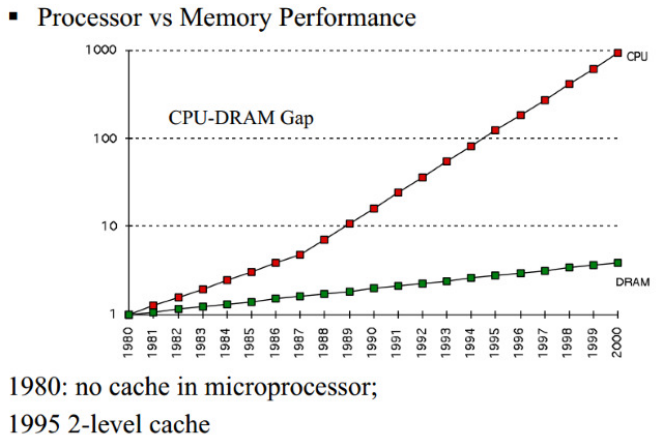
*Corresponding author.

E-mail addresses: wismerchil@interia.eu, danielmz25@gmail.com

1. Wstęp

Prężny rozwój gier komputerowych na przestrzeni ostatnich lat znacząco zwiększył zapotrzebowanie na nowe technologie wspomagające proces produkcji oraz wydajność aplikacji. Również technologia wewnątrz podzespołów komputera uległa znaczącym zmianom. Kolejne lata przyniosły znaczący wzrost mocy obliczeniowej procesora podczas gdy wydajność pamięci komputera wzrosła bardzo niewiele (Rys. 1). Programowanie równoległe może jeszcze zwiększyć narzut czasu dostępu do pamięci na wydajność programu. W wielu gałęziach programowania projektowanie oparte na danych (DOD) wykorzystywane jest od lat w celu zwiększenia wydajności. Dodatkowym problemem wiążącym się z rozwojem branży gier jest rosnący rozmiar nowych projektów zwiększający wymagania na nowe technologie przyśpieszające tworzenie gry, zapewniając dodatkowo większe bezpieczeństwo oraz zmniejszające koszty produkcji.

Celem niniejszego artykułu jest przedstawienie i przetestowanie rozwiązania Entity-component-system (ECS) opartego o DOD. Omówione zostaną podstawowe korzyści oraz ograniczenia płynące z wykorzystania podejścia, a także zostanie dokonane porównane go z aktualnie najpopularniejszym programowaniem zorientowanym obiektowo. Badania przedstawione w artykule oparte zostały o autorskie rozwiązania. Wykorzystany został język D będący statycznie typowanym językiem ogólnego przeznaczenia [1].



Rys. 1. Zestawienie wydajności procesora z wydajnością pamięci RAM w kolejnych latach [2]

2. Programowanie obiektowe

Programowanie obiektowe (ang. object-oriented programming, OOP) to paradygmat programowania, w którym aplikacje opisuje się za pomocą obiektów. Obiekt

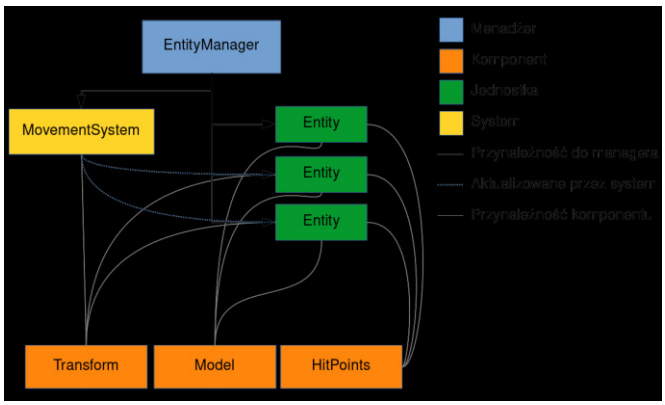
to połączenie danych oraz metod. Zazwyczaj obiekty tworzone są jako abstrakcja rzeczywistego obiektu klasyfikując dane oraz funkcje w sposób logiczny dla człowieka [3]. W parze z programowaniem obiektowym idzie projektowanie zorientowane obiektowo (ang. object-oriented design, OOD). Choć programowanie obiektowe jest jedynie wzorcem, nienarzucającym wykorzystywanych w programie technik, jest wysoce niekompatybilne z projektowaniem zorientowanym na dane.

3. Entity-component-system

ECS jest wzorcem architektonicznym używanym przede wszystkim przy tworzeniu gier komputerowych [4]. Jest zgodny z zasadą "kompozycja ponad dziedziczenie" minimalizując redundancję danych. Wzorzec składa się z trzech głównych elementów:

- Komponent (ang. component) – surowe dane opisujące pojedynczy aspekt obiektu oraz jego interakcję ze światem. Komponent implementowany jest zazwyczaj za pomocą struktur danych. Działa jak etykieta opisująca właściwość obiektu
- Jednostka (ang. entity) – obiekt ogólnego przeznaczenia. Zazwyczaj składa się jedynie z unikalnego identyfikatora, implementowanego jako liczba całkowita. Jednostka składa się z komponentów, które jednak nie muszą znajdować się w pamięci tam gdzie jednostka. Dostęp do komponentów odbywa się przy pomocy identyfikatora. Komponenty zawarte w jednostce określają jej typ
- System – zawiera logikę programu. Wykonuje akcje dla każdej jednostki posiadającej typ o tym samym aspekcie co system. Systemy działają niezależnie od siebie.

Wzorzec eliminuje problem programowania obiektowego związanego z głębokimi oraz szerokimi hierarchiami dziedziczenia. Zazwyczaj wszystkie kluczowe elementy tego rozwiązania spajane są za pomocą menadżera jednostek zarządzającego jednostkami oraz systemami. Na rysunku 2 przedstawiono przykładową strukturę aplikacji. System aktualizuje jedynie te jednostki, które posiadają komponenty przypisane również do systemu.



Rys. 2. Przykładowa struktura aplikacji wykorzystująca wzorzec ECS

Tworzenie aplikacji ogranicza się do dodawania nowych komponentów oraz programowania logiki systemów. ECS jest wysoce kompatybilny z projektowaniem opartym na danych.

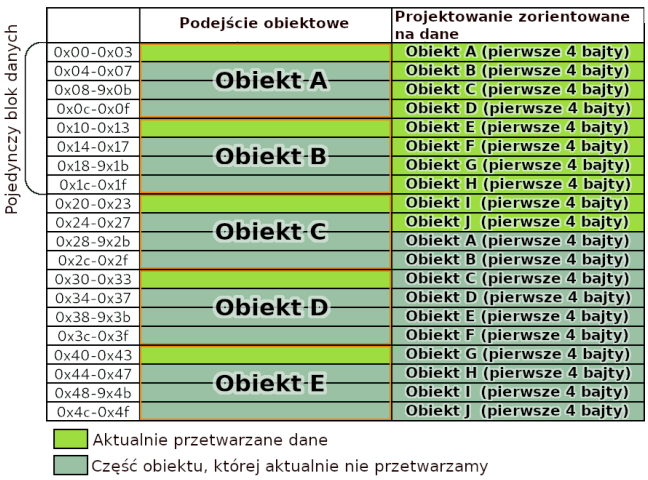
4. Data oriented design

Projektowanie zorientowane na dane ma na celu zwiększyć wydajność programu poprzez zmniejszenie czasu oczekiwania na pamięć. Ponieważ procesor pobiera dane blokowo, gdzie rozmiar bloku to zazwyczaj 32B bądź 64B, wykorzystanie pojedynczej zmiennej spowoduje wczytanie redundantnych danych [5]. Czas oczekiwania od momentu wysłania informacji o zapotrzebowaniu na dane do pamięci RAM do ich dotarcia jest kilka rzędów wyższy niż standardowe operacje [6]. Celem zmniejszenia wyżej wymienionego problemu w procesorach montowana jest wielopoziomowa pamięć podręczna [7]. Każdy kolejny poziom cechuje się większym rozmiarem oraz większym czasem dostępu. Najniższy poziom posiada zazwyczaj jedynie kilkukrotnie dłuższy czas dostępu niż rejestry procesora. Ponieważ jednak programy wykorzystują ogromne ilości danych, prawdopodobieństwo, że potrzebne dane nie znajdują się na niższych poziomach pamięci podręcznej jest wysokie.

Procesory posiadają jeszcze jeden istotny mechanizm mający na celu zminimalizowanie wpływu narzutu dostępu do pamięci na aplikację. Technologia przewidywania kolejnych instrukcji programu pozwala wysyłać do pamięci żądanie jeszcze przed instrukcją, która będzie ich potrzebować. Aby jednak technika ta działała poprawnie wymaga nieskomplikowanego schematu wczytywania kolejnych danych, tak aby procesor mógł przewidzieć, które dane powinien wczytać. Zazwyczaj proces optymalizacji pod kątem tej funkcjonalności sprowadza się do umieszczania danych w pamięci ciągłej, gdzie procesor wykorzystuje wszystkie informacje jedna po drugiej [8].

Projektowanie zorientowane na dane opiera się więc głównie na dwóch założeniach.

1) Należy maksymalnie zmniejszyć redundancję danych tak, aby wszystkie dane wewnątrz bloku wczytanego przez procesor były wykorzystywane w aktualnych obliczeniach
2) Należy umieścić dane jedne obok drugich posortowane w kolejności w jakiej będą wykorzystane.



Rys. 3. Porównanie układu pamięci w programowaniu obiektowym oraz z użyciem techniki DOD

Spełnienie obu wyżej wymienionych założeń zapewni aplikacji najbardziej optymalne wykorzystanie pamięci. Rysunek 3 przedstawia porównanie układu pamięci rozwiązania DOD oraz programowania obiektowego.

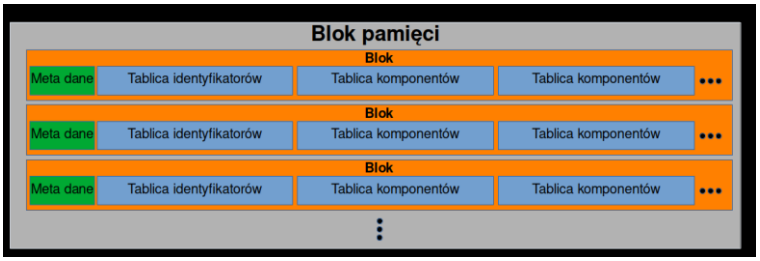
5. Aplikacja testowa

Na potrzeby testów przygotowana została aplikacja testowa oparta o autorskie rozwiązania, w skład których wchodzą silnik graficzny oraz biblioteka DECS realizująca wzorzec ECS. Oprogramowanie zawiera różne testy zaprojektowane w czterech wariantach:

- ECS – podejście oparte na wzorcu ECS. Posiada układ pamięci podobny jak przy programowaniu obiektowym grupując dane obiektów
- Liniowy ECS – rozwiązanie wykorzystujące wzorzec ECS z liniowym układem pamięci zapewniając, że pamięć identycznych komponentów zawsze grupowana jest w pamięci RAM minimalizując czas oczekiwania na dostęp do danych
- Programowanie obiektowe z dziedziczeniem - podejście dla każdego testu zawiera jedną klasę składającą się z danych oraz funkcji aktualizacji
- Programowanie obiektowe z grupowaniem obiektów – rozwiązanie, w którym pamięć obiektów przechowywana jest w ciągłej pamięci (jeden obiekt za drugim) dzięki czemu procesor odwołuje się do elementów pamięci znajdujących się blisko siebie.

Główną metodą omawianą w artykule jest ECS z liniowym układem pamięci. Schemat układu pamięci w tym rozwiązaniu przedstawia rysunek 4.

Wszystkie testy posiadają identyczną strukturę. Pierwszym etapem jest inicjalizacja aplikacji, podczas której alokowane są zasoby oraz tworzone obiekty. Każdy test poza trzecim, który tworzy pół miliona obiektów, wykonywany jest dla miliona obiektów. Kolejnym krokiem jest proces aktualizacji polegającej na wykonaniu pewnej grupy obliczeń na danych wszystkich utworzonych obiektów. Aktualizacja przeprowadzana jest sześćset razy i wyliczany jest średni czas przeprowadzenia jednego takiego procesu. Ostatnim etapem jest finalizacja, podczas której zwalniane są zasoby aplikacji.



Rys. 4. Schemat układu pamięci w rozwiązaniu ECS z liniowym układem pamięci

Pierwsze dwa testy zawierają prostą logikę aktualizacji, której jedynym elementem jest przemieszczanie obiektów. W każdym kroku symulacji obiekty przemieszczane są o stałą wartość. Test drugi zawiera większą ilość redundantnych danych wewnątrz obiektów. Przykłady 1 oraz 2 przedstawiają funkcje aktualizacji kolejno systemu architektury ECS oraz obiektu w podejściu obiektowym.

Przykład 1. Kod aktualizacji systemu w architekturze ECS

```
void update()
{
 position.y += -0.001 * app_state.delta_time;
 if(position.y < -10)
 {
  position.y = 10;
 }
}
```

Przykład 2. Kod aktualizacji obiektu

```
void update()
{
 position.y += -0.001 * app_state.delta_time;
 if(position.y < -10)
 {
  position.y = 10;
 }
}
```

Test trzeci różni się od dwóch poprzednich jedynie bardziej skomplikowanymi obliczeniami wewnątrz funkcji aktualizacji. Przemieszczanie obiektów wykorzystuje funkcje trygonometryczne oraz funkcje wygładzające.

Czwarty test nie zawiera funkcji aktualizacji, ale dużo zewnętrznych odwołań do danych obiektów. Celem testu jest zbadanie narzutu odwołania przy użyciu identyfikatora na wydajność aplikacji.

Ostatnie dwa testy przedstawiają różne rozwiązania identycznego problemu dla architektury ECS. Zdarza się tak, że pewien obiekt w trakcie działania aplikacji zmienia swoje zachowanie. W programowaniu obiektowym uzyskuje się to zazwyczaj instrukcją warunkową wewnątrz metody obiektu, bądź za pomocą listy obiektów podlegających pewnym obliczeniom. Architektura ECS pozwala dodatkowo zmieniać zachowanie obiektu poprzez dodanie bądź usunięcie komponentu do jednostki. Testy badają wydajność kolejno instrukcji warunkowej oraz komponentu etykiety do zmiany zachowania obiektu.

6. Porównanie wydajności

Testy przeprowadzone zostały na komputerze stacjonarnym o specyfikacji przedstawionej w tabeli 1.

Tabela 1. Specyfikacja sprzętu wykorzystywanego w testach

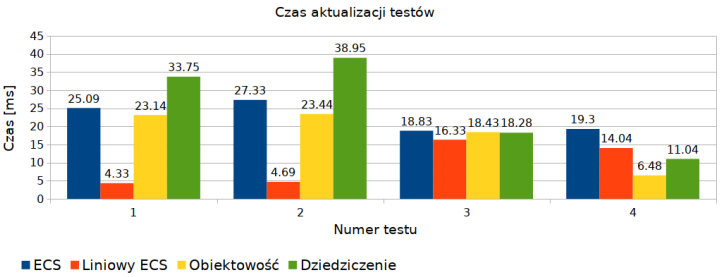
System operacyjny	Linux Arch
Wersja jądra	5.2.arch2
Procesor	AMD FX-8320e
Liczba rdzeni	8
Liczba wątków	8
Pamięć RAM	8GB DDR3 1600MHz

W tabeli 2 przedstawiony są wyniki testu pierwszego zawierające czas inicjalizacji programu, średni czas aktualizacji, czas zakończenia programu i zwalniania zasobów.

Tabela 2. Wyniki testu pierwszego dla wszystkich testowanych rozwiązań

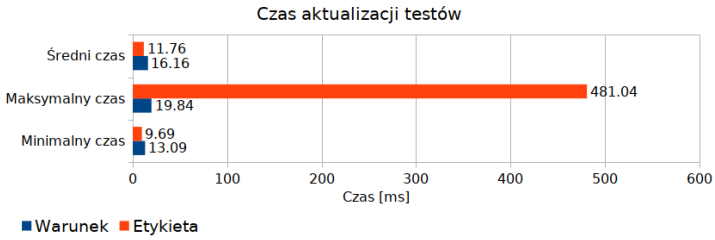
Rozwiązanie	Inicjalizacja [ms]	Aktualizacja [ms]	Zakończenie [ms]
ECS	173.34	25.09	7.68
Liniowy ECS	245.56	4.33	9.76
Programowanie obiektowe z grupowaniem obiektów	112.76	23.14	6.78
Programowanie obiektowe z dziedziczeniem	210.43	33.75	93

Rysunek 5 zawiera diagram czasu aktualizacji dla pierwszych czterech testów.



Rys. 5. Zestawienie czasu aktualizacji pierwszych czterech testów dla wszystkich testowanych rozwiązań

Na rysunku 6 ukazany jest wykres czasu trwania procesu aktualizacji dwóch ostatnich testów z uwzględnieniem najkrótszego oraz najdłuższego czasu aktualizacji.



Rys. 6. Porównanie czasów trwania etapu aktualizacji przy wykorzystaniu komponentu jako etykiety oraz instrukcji warunkowej wewnątrz logiki systemu

7. Podsumowanie

Czas finalizacji oraz inicjalizacji standardowego obiektowego podejścia obiektowego z dziedziczeniem jest wysoki, ponieważ pamięć każdego obiektu alokowana jest osobno. Czas inicjalizacji rozwiązania ECS z liniowym układem pamięci jest wysoki ze względu na konieczność kopiowania danych w różne miejsca w pamięci w celu zgrupowania komponentów.

Podejście ECS z liniowym układem pamięci pozwala na uzyskanie dużo wyższej wydajności. W pierwszych dwóch przypadkach gdy wąskim gardłem aplikacji był czas dostępu do pamięci wydajność tego rozwiązania była nawet 8x większa od standardowego podejścia obiektowego z dziedziczeniem. Wydajność architektury ECS w porównaniu do podejścia obiektowego z grupowaniem obiektów posiadającego identyczny układ pamięci jest podobna, a więc wpływ architektury na wydajność aplikacji jest względnie niewielki. W teście trzecim zauważyć można, że mimo dużo bardziej skomplikowanych obliczeń architektura ECS z liniowym układem pamięci jest nieco bardziej wydajna od pozostałych. Dostęp do pamięci przy użyciu identyfikatora jest nieco wolniejszy niż standardowy dostęp z wykorzystaniem adresu, jednak różnica nie jest tak duża. Najwydajniej prezentuje się rozwiązanie obiektowe z grupowaniem obiektów, jest to jednak rozwiązanie niepraktyczne, mające jedynie pokazać jaką wydajność można uzyskać poprzez idealne ułożenie pamięci zawierające jednocześnie redundantne dane.

Wykorzystanie komponentów jako etykiety warunkującej wykonanie obliczeń pozawala na uzyskanie niewielkiego wzrostu wydajności zakładając, że zmiana zachowania obiektu następuje względnie rzadko.

Przechodząc od wyników do porównania metod pod względem łatwości tworzenia aplikacji warto zauważyć, że architektura ECS stworzona została dla uproszczenia procesu tworzenia aplikacji. Pierwszą jej zaletą jest zmniejszenie redundancji kodu. W architekturze zorientowanej obiektowo klasy często składają się z częściowo identycznych danych oraz funkcji. Można oczywiście wykorzystać dziedziczenie, jednak dalej nie rozwiąże to problemu całkowicie. W architekturze ECS natomiast komponenty oraz systemy są współdzielone pomiędzy jednostkami w zależności od potrzeb. Każdy obiekt niezależnie od jego typu może uzyskać nowy komponent, automatycznie zmieniając jego zachowanie. Ponieważ logika przypisana jest do systemów oraz zawartych w jednostce komponentów, zamiast pisać kod dla każdego obiektu, tworzy się zachowania powiązane z komponentami. Podejście to znacząco zwiększa elastyczność projektowania aplikacji, pozwalając dodatkowo zmieniać zasadę działania programu nawet na zaawansowanym poziomie prac. Problemem projektowania zorientowanego obiektowo jest fakt, że gdy pewna funkcjonalność nie zostanie przemyślana na poziomie projektowania aplikacji, często dodanie jej wymaga przebudowy znacznej części programu. Oczywiście mowa tutaj to o rozwiązaniach opartych na rozbudowanych relacjach pomiędzy obiektami. Paradygmat programowania obiektowego pozwala na budowę dobrej architektury aplikacji, takiej, w której zmiany kodu nie wymuszają zmian całej jego struktury. Jednak tak jak projektowanie zorientowane obiektowo naturalnie prowadzi do redundancji oraz zależności pomiędzy elementami kodu, tak wzorzec ECS minimalizuje ten problem jednocześnie odseparowując funkcjonalności aplikacji. Zaletą architektury ECS jest także podział kodu oraz danych. Systemy nie posiadają informacji o typach obiektów jakie przetwarzają, ani o innych systemach. Dzięki temu podziałowi łatwo jest aplikację dzielić na wiele bibliotek, bądź zaprojektować rozwiązanie Hot-reload pozwalające na zmianę działania programu w czasie jego działania.

Do wad prezentowanej architektury należałoby zaliczyć przede wszystkim trudność zrozumienia metodyki programowania. OOD jest dużo bardziej przystępnym wzorcem projektowania aplikacji dla początkujących programistów. Dodatkowo dostęp do danych komponentów spoza funkcji systemu zawiera narzut obliczeniowy zmuszając programistę do minimalizacji zewnętrznych odwołań do obiektu. Ostatnią istotną wadą architektury ECS jest fakt, że rozwiązanie opiera się na grupowaniu danych w komponentach. Jeżeli aplikacja zawiera niewielką liczbę obiektów, dodatkowo niewspółdzielących funkcjonalności oraz danych, programowanie obiektowe czy inne podejścia mogą okazać się znacznie lepszym rozwiązaniem.

8. Wnioski

Prezentowana architektura jest doskonałym podejściem wykorzystywanym w procesie tworzenia gier. Wykorzystanie tego rozwiązania do innych dziedzin programowania niż tworzenie gier będzie prawdopodobnie złym pomysłem. Podejście pozwala jednak uzyskać znacznie wyższą wydajność oferując jednocześnie bardziej elastyczny proces tworzenia aplikacji oraz ułatwiając wprowadzanie wielu nowych technik. Ponieważ program posiada wszystkie dane systemów oraz komponentów, można w łatwy sposób zbudować algorytm automatycznego generowania zadań równoległych oraz ich zależności. Synchronizacja sieciowa może opierać się na zautomatyzowanej synchronizacji wybranych komponentów. Logika debugowania aplikacji może zawierać się w systemach, które włączane będą w zależności od potrzeb. Rozszerzanie funkcjonalności programu może odbywać się poprzez dodawanie dynamicznych bibliotek zawierających kod nowych systemów. Podział kodu na systemy pozwala w prosty sposób zmienić założenia projektowe na zaawansowanym poziomie prac.

Literatura

[1] Strona główna języka D https://dlang.org/ [10.09.2019]

[2] Opis działania pamięci podręcznej procesora https://www.extremetech.com/extreme/188776-how-l1-and-l2-cpu-caches-work-and-why-theyre-an-essential-part-of-modern-chips [10.09.2019]

[3] Opis programowania zorientowanego obiektowego https://en.wikipedia.org/wiki/Object-oriented_programming [10.09.2019]

[4] Opis wzorca Entity-component-system https://en.wikipedia.org/wiki/Entity_component_system [10.09.2019]

[5] Fabian R., Data-Oriented Design: Software Engineering for Limited Resources and Short Schedules, DataOrientedDesign.com, 2018

[6] Albrecht T., Pitfalls of Object Oriented Programming, na: Proceedings of Game Connect: Asia Pacific (GCAP) 2009, Melbourne, Australia, 2009

[7] Kowarschik M., Weiß C., An Overview of Cache Optimization Techniques and Cache-Aware Numerical Algorithms, Springer-Verlag, Berlin, Heidelberg, 2003

[8] Nystrom R., Game Programming Patterns, gameprogrammingpatterns.com, 2014

JOURNAL of COMPUTER SCIENCES INSTITUTE

JCSI 13 (2019) 354-358
Wysłane: 2019-09-14
Przyjęte: 2019-09-22

Analiza porównawcza języków Kotlin i Java używanych do tworzenia aplikacji na system Android

Daniel Sulowski*, Grzegorz Kozieł
Politechnika Lubelska, Instytut Informatyki, Nadbystrzycka 36B, 20-618 Lublin, Polska

Streszczenie. Niniejsza publikacja przedstawia rezultaty analizy porównawczej języków programowania Java i Kotlin, wykorzystywanych do tworzenia aplikacji mobilnych przeznaczonych do pracy pod kontrolą systemu Android. Analiza obejmuje aspekty związane z wydajnością, takie jak obciążenie procesora, obciążenie pamięci RAM, a także czasy kompilacji i wykonania programu. Pod uwagę wzięto również strukturę kodu, dostępność bibliotek, obsługiwane bazy danych, popularność oraz wsparcie społeczności.

Słowa kluczowe: Android; Java; Kotlin; wydajność

*Autor do korespondencji.

Adres e-mail: daniel.sulowski95@gmail.com

Comparative analysis of Kotlin and Java languages used to create applications for the Android system

Daniel Sulowski*, Grzegorz Kozieł
Institute of Computer Science, Lublin University of Technology, Nadbystrzycka 36B, 20-618 Lublin, Poland

Abstract. This publication presents the results of a comparative analysis of Java and Kotlin programming languages used to create mobile applications for system Android. The analysis covers performance aspects such as CPU load, RAM load, as well as the compilation and execution times. Aspects such as code structure, availability of libraries, supported databases, popularity and community support were taken under consolidation.

Keywords: Android; Java; Kotlin; performance

*Corresponding author.

E-mail address: daniel.sulowski95@gmail.com

1. Wstęp

W październiku 2008 r. wydano pierwszą wersję systemu Android. Od tego momentu aż do 2017 r. Java była jedynym językiem programowania wykorzystywanym do tworzenia aplikacji mobilnych przeznaczonych do pracy pod kontrolą tego systemu [1]. Na konferencji Google I/O w 2017 roku ogłoszono bowiem oficjalne wsparcie dla języka Kotlin przy programowaniu aplikacji mobilnych. Głównym powodem stworzenia tego języka była chęć eliminacji błędów powstałych już w czasie projektowania języka Java. Kotlin miał za zadanie wyeliminowanie sytuacji mogących wywołać nieoczekiwane zamknięcie aplikacji (*null-pointer safety*), a także ułatwienie procesu implementacji aplikacji mobilnych. Istotnym aspektem była również łatwa możliwość migracji kodu z Javy do Kotlina. Ponadto, języki te pozostają do siebie bardzo podobne i generują zbliżony kod bajtowy [2].

Pomimo podobnej struktury kodu, ta sama aplikacja napisana w obydwu językach może w inny sposób wpływać na podzespoły urządzenia na którym pracuje. Z samej dokumentacji Kotlina wynika, że mogą wystąpić różnice w wykorzystaniu pamięci RAM. Autorzy tej dokumentacji podają również, że aplikacje pisane w Kotlinie cechują się krótszym czasem wykonania [2]. Ważne są również różnice związane z procesem projektowania i implementacji. Zmieniona struktura kodu oraz wprowadzenie nowej funkcjonalności może wpływać na czas i koszt tworzenia oprogramowania. Nie bez znaczenia pozostaje trudność danej technologii. Mimo podobnych zastosowań jeden język może być znacznie prostszy w użyciu aniżeli drugi.

2. Cel badań

Niniejszy artykuł ma na celu przeprowadzenie analizy porównawczej języków programowania Java i Kotlin. Skupiono się przede wszystkim na wydajności tworzonych aplikacji. Zbadano takie parametry jak obciążenie procesora, obciążenie pamięci RAM oraz czasy kompilacji i wykonania programu. Pod uwagę wzięto także budowę kodu, dostępność bibliotek, obsługiwane bazy danych oraz popularność i wsparcie społeczności.

3. Metoda badań

W celu zbadania wydajności języków zaimplementowano specjalną aplikację mobilną, działającą w identyczny sposób zarówno w Javie, jak i Kotlinie. Jej działanie polega na generowaniu liczb i sortowaniu dużych zbiorów danych z wykorzystaniem kilku algorytmów oraz wyświetlaniu prostej animacji.

Badania dotyczące wydajności przeprowadzono na 3 różnych urządzeniach:

- Xiaomi Redmi 4A,
- Xiaomi Redmi 7,
- LG K350N.

Odizolowano aplikację od czynników zewnętrznych mogących wpłynąć na wynik pomiaru. W tym celu wyłączono połączenie z internetem, GPS, moduł Bluetooth, a także zamknięto aplikacje działające w tle. Uruchomiono również tryb samolotowy i ustawiono maksymalną jasność ekranu. Wyniki badań odczytano za pomocą środowiska programistycznego Android Studio. Do odczytu obciążenia procesora i pamięci RAM wykorzystano specjalny moduł Android Profiler. Łącznie dokonano 54 pomiarów każdego parametru na wszystkich urządzeniach.

Na podstawie literatury oraz doświadczeń związanych z programowaniem dedykowanej aplikacji porównano strukturę kodu badanych języków. Zbadano popularność oraz wsparcie społeczności w oparciu o dane zebrane z takich serwisów jak Stackoverflow czy Github. Sprawdzono również dostępność bibliotek oraz obsługiwane bazy danych.

4. Analiza literatury

Pierwszym istotnym źródłem wiedzy są oficjalne dokumentacje. Już z samej dokumentacji Kotlina można się dowiedzieć o różnicach w składni oraz bezpieczeństwie tworzonych aplikacji. Wprowadzono wiele udogodnień dla zaawansowanych programistów oraz znacznie zredukowano rozmiar kodu. Przekłada się to na krótszy czas potrzebny do implementacji programu. Skrócony kod jest jednak mniej intuicyjny, zwłaszcza dla niedoświadczonych programistów. Ograniczono również możliwość wystąpienia nieoczekiwanego zamknięcia aplikacji. Ponadto, dokumentacja Kotlina sugeruje, że programy napisane w tym języku działają szybciej niż programy napisane w Javie. Występują również różnice w czasie kompilacji. Kotlin osiąga lepszy czas w przypadku kompilacji przyrostowej, z kolei Java szybciej kompiluje program od zera [2].

Pracę o podobnej tematyce opublikował w maju 2018 Patrick Schwermer z Królewskiego Instytutu Technicznego w Sztokholmie. Autor dowodzi, że programy napisane w Javie działają nieznacznie szybciej niż programy napisane w Kotlinie, co jest sprzeczne z oficjalną dokumentacją Kotlina. Ponadto, programy napisane w Kotlinie mają zużywać więcej pamięci, ale lepiej funkcjonuje mechanizm odpowiedzialny za jej oczyszczanie [3]. Autorzy pracy „A comparative Study: Java vs Kotlin Programming in Android Application Development" wydanej w czerwcu 2018 podają, że Java jest lepszym wyborem dla początkujących programistów, ponieważ cechuje się prostszą składnią i większym wsparciem społeczności. Kotlin z kolei zdaje się być lepszym narzędziem w rękach doświadczonego programisty, ponieważ pisany kod jest krótszy oraz trudniej jest w nim popełnić błędy [4]. Do podobnych wniosków doszli autorzy pracy „Are you still smelling it?: A comparative study between Java and Kotlin language". Zbadano zapach kodu na przykładzie 50 projektów Java oraz 50 projektów Kotlin dostępnych na repozytorium Github. Autorzy udowadniają, że to Kotlin cechuje się mniejszym zapachem kodu [5].

Pozostałe prace są w zasadzie podsumowaniem informacji zawartych w dokumentacji Kotlina i nie wnoszą niczego nowego. Kotlin jest oficjalnie wspierany przez Androida od niespełna 2 lat, zatem temat ten nie jest jeszcze dostatecznie przebadany.

5.1. Analiza porównawcza cech badanych języków

W tabeli 1 zestawiono największe różnice pomiędzy strukturą kodu Javy i Kotlina.

Tabela 1. Różnice w budowie kodu pomiędzy Javą i Kotlinem

Java	Kotlin
Wyrażenia lambda wprowadzono w wersji 8, która nie jest jeszcze w pełni wspierana przez Androida [6]	Wprowadzono wyrażenia lambda używane do przekazywania bloku kodu jako argumentu funkcji
Brak funkcji jednoliniowych (*inline functions*)	Wprowadzenie funkcji jednoliniowych
Konieczność deklaracji każdego elementu interfejsu	Możliwość bezpośredniego odwołania się do każdego elementu interfejsu
Brak mechanizmów chroniących przed wyjątkiem *NullPointerException*	Wprowadzenie operatorów eliminujących wyjątek *NullPointerException* [7]
Obowiązek sprawdzenia wystąpienia wyjątków [6]	Wystąpienie wyjątku nie musi być sprawdzane
Konieczność deklaracji typów danych	Zwolnienie programisty w obowiązku deklaracji typów danych [8]
Istnienie prymitywnych typów danych (np. *int*, *char*)	Wszystkie zmienne są obiektami
Istnienie modyfikatora *static*	Usunięto modyfikator *static*. Wprowadzenie tzw. obiektów towarzyszących (*companion objects*) [9]
Dobrej jakości kod powinien zawierać metody *get* i *set*	Metody *get* i *set* generowane są automatycznie i pozostają niewidoczne w kodzie źródłowym
Średnik występuje na końcu każdej instrukcji	Brak średnika na końcu instrukcji [9]

Odświeżona składnia języka Kotlin pozwala pisać zdecydowanie krótszy kod. Przekłada się to na czas pisania programu. W przypadku niektórych aplikacji, Kotlin może zredukować rozmiar kodu nawet o kilkaset linijek, jeżeli weźmiemy pod uwagę brak konieczności sprawdzania wystąpienia wyjątków czy pisania metod *get* i *set*. Poniższe listingi przedstawiają tę samą klasę napisaną w Javie oraz w Kotlinie. Kilkanaście linijek kodu Javy odpowiada jednej linijce kodu w Kotlinie.

Przykład 1. Klasa Student w Javie

```
class Student {
   private String name;
   private int age;

   public String getName() {
      return name;
   }
```

```
    public void setName(String name) {
        this.name = name;
    }
    public int getAge() {
        return age;
    }
    public void setAge(int age) {
        this.age = age;
    }
}
```

Przykład 2. Klasa Student w Kotlinie

```
class Student(var name: String, var age : Int)
```

Ponadto, Kotlin oferuje nową funkcjonalność taką jak np. wyrażenia lambda czy funkcje jednoliniowe. Powoduje to, że będzie on zdecydowanie lepszym narzędziem w rękach doświadczonego programisty. Korzystając z Kotlina, można uzyskać zamierzony cel, wykorzystując znacznie mniej kodu i czasu niż w przypadku Javy. Osoba początkująca powinna jednak rozpocząć swoją naukę od Javy. Język ten cechuje się bowiem bardziej intuicyjnym i klarownym kodem. Warto dodać, że w internecie znaleźć można zdecydowanie więcej materiałów do nauki Javy, niż do nauki Kotlina. Jest to oczywiście związane z tym, że Kotlin jest językiem stosunkowo młodym, natomiast Java istnieje na rynku od ponad 20 lat. W tym czasie wydano wiele książek do nauki Javy, w praktycznie wszystkich językach. Kotlin oferuje w zamian narzędzie online zwane Kotlin Koans. Jest to specjalny moduł oferujący interaktywną naukę tego języka bez potrzeby pobierania dodatkowego oprogramowania. Jego wadą jest stosunkowo wysoki poziom trudności, jeżeli dana osoba nie ma doświadczenia z programowaniem.

Java cieszy się również znacznie większą popularnością. We wrześniu 2019 r. w serwisie Github znajdowało się ponad milion repozytoriów pod tagiem „Java" i nieco ponad 67 tysięcy pod tagiem „Kotlin". Natomiast w serwisie Stackoverflow zadanych zostało do tej pory aż 1,5 miliona pytań dotyczących Javy i nieco ponad 70 tysięcy pytań dotyczących Kotlina. W przyszłości jednak trend ten może się odwrócić i to Kotlin może stać się liderem, jeżeli chodzi o programowanie aplikacji mobilnych. Aktualnie można zaobserwować dużą migrację projektów z Javy do Kotlina. Zdecydowały się na to takie serwisy jak Twitter, Pinterest czy Netflix. Nawet twórcy Kotlina przewidzieli taką możliwość i udostępnili mechanizm pozwalający na konwersje kodu z Javy do Kotlina. Obydwa języki są interoperacyjne i kompilują się do podobnego kodu bajtowego [10]. Sprawia to, że języki te mogą być mieszane w dowolnym projekcie bez obaw o komplikacje z kompatybilnością. Tyczy się to również bibliotek i baz danych. Wszystkie narzędzia obsługiwane przez jeden język będą również obsługiwane przez drugi.

5.2. Analiza porównawcza wydajności badanych języków

Na podstawie wyników badań obliczono średnią, medianę, wartość minimalną oraz wartość maksymalną każdego badanego parametru. Uzyskane rezultaty zestawiono w formie tabeli.

5.2.1. Obciążenie procesora

Poniższe tabele zawierają dane statystyczne dotyczące obciążenia procesora. Tabela 2 odnosi się do Javy, z kolei tabela 3 do Kotlina.

Tabela 2. Obciążenie procesora - Java

Urządzenie	LG K350N	Xiaomi Redmi 4A	Xiaomi Redmi 7
Wartość minimalna	44,5%	40,9%	14,9%
Wartość maksymalna	53,5%	45,9%	18%
Średnia	48,94%	43,41%	16,37%
Mediana	49,4%	43,8%	16%

Tabela 3. Obciążenie procesora – Kotlin

Urządzenie	LG K350N	Xiaomi Redmi 4A	Xiaomi Redmi 7
Wartość minimalna	44,5%	40,8%	15%
Wartość maksymalna	52%	46%	18,8%
Średnia	48,77%	43,49%	16,3%
Mediana	49,3%	43,75%	16%

Poszczególne wartości w powyższych tabelach są do siebie bardzo zbliżone, a różnice nie przekraczają 1%. Zatem wybór technologii pomiędzy Javą a Kotlinem nie wpływa znacząco na obciążenie procesora.

5.2.2. Obciążenie pamięci RAM

W tabelach poniżej zaprezentowano jak kształtowało się obciążenie pamięci RAM na badanych urządzeniach. Tabela 4 dotyczy Javy, natomiast tabela 5 Kotlina.

Tabela 4. Obciążenie pamięci RAM - Java

Urządzenie	LG K350N	Xiaomi Redmi 4A	Xiaomi Redmi 7
Wartość minimalna	266,4 MB	263,6 MB	266,5 MB
Wartość maksymalna	275,3 MB	293,5 MB	278,9 MB
Średnia	268,8 MB	269,9 MB	269 MB
Mediana	268,5 MB	265,2 MB	268,9 MB

Tabela 5. Obciążenie pamięci RAM – Kotlin

Urządzenie	LG K350N	Xiaomi Redmi 4A	Xiaomi Redmi 7
Wartość minimalna	268,1 MB	263,4 MB	267,1 MB
Wartość maksymalna	277,2 MB	292,1 MB	279,3 MB
Średnia	269,6 MB	270,6 MB	270,4 MB
Mediana	269,5 MB	266,4 MB	269,8 MB

Na podstawie mediany i średniej można stwierdzić, że to Kotlin zużywa więcej pamięci. Różnica ta jest jednak bardzo mała i wynosi tylko 1 MB, jednak może się ona pogłębiać w przypadku mocno rozbudowanych programów. Różnice pomiędzy wartościami minimalnymi i maksymalnymi wynoszące do 30 MB mogą być spowodowane tym, że aplikacja generuje losowe obiekty i zbiory danych, które z każdym uruchomieniem zajmują inny obszar pamięci. Ponadto, pamięć RAM jest wykorzystywana w procesach systemowych, co również mogło mieć wpływ na wyniki badań.

5.2.3. Czas kompilacji

Jak podaje oficjalna dokumentacja Kotlina, aplikacje napisane w Javie powinny kompilować się ok. 15% szybciej niż aplikacje napisane w Kotlinie [2]. Tabela 6 przedstawia czas kompilacji w Javie, a tabela 7 czas kompilacji w Kotlinie. Czas ten został odczytany za pomocą środowiska programistycznego Android Studio.

Tabela 6. Czas kompilacji – Java

Urządzenie	LG K350N	Xiaomi Redmi 4A	Xiaomi Redmi 7
Wartość minimalna	899ms	802ms	838ms
Wartość maksymalna	1383ms	1336ms	1321ms
Średnia	1043ms	966ms	1043ms
Mediana	1020ms	943ms	1036ms

Tabela 7. Czas kompilacji – Kotlin

Urządzenie	LG K350N	Xiaomi Redmi 4A	Xiaomi Redmi 7
Wartość minimalna	879ms	900ms	895ms
Wartość maksymalna	1172ms	1812ms	2179ms
Średnia	999ms	1102ms	1193ms
Mediana	993ms	1057ms	1178ms

Rezultaty badań przeprowadzonych na telefonach Xiaomi są zgodne z oficjalną dokumentacją Kotlina. Java kompiluje programy o ok. 15% szybciej niż Kotlin. Niestety, sprzeczne wnioski dają badania przeprowadzone na LG K350N. Wynika z nich, że to Kotlin szybciej kompiluje programy (o ok. 3%). Trudno jest zatem na tej podstawie wysunąć jednoznaczne wnioski, ponieważ czas kompilacji oprócz technologii, jest też mocno uzależniony od samego urządzenia.

5.2.4. Czas wykonania

Czas wykonania programu jest w dużej mierze zależny od podzespołów, w które wyposażone jest badane urządzenie. Lepszy procesor i pamięć RAM o większej pojemności pozwalają osiągnąć krótszy czas wykonania. W tabeli 8 zestawiono czasy wykonania aplikacji Java, z kolei w tabeli 9 czasy wykonania aplikacji Kotlin. Czasy te zostały zmierzone programistycznie.

Tabela 8. Czas wykonania – Java

Urządzenie	LG K350N	Xiaomi Redmi 4A	Xiaomi Redmi 7
Wartość minimalna	43549ms	4046ms	2319ms
Wartość maksymalna	47212ms	4153ms	2401ms
Średnia	44301ms	4077ms	2338ms
Mediana	44081ms	4073ms	2335ms

Tabela 9. Czas wykonania – Kotlin

Urządzenie	LG K350N	Xiaomi Redmi 4A	Xiaomi Redmi 7
Wartość minimalna	40339ms	4627ms	2498ms
Wartość maksymalna	41078ms	4856ms	2540ms
Średnia	40686ms	4656ms	2518ms
Mediana	40694ms	4653ms	2518ms

Badania przeprowadzone na telefonach marki Xiaomi sugerują, że aplikacje napisane w Javie są szybsze od aplikacji napisanych w Kotlinie. Natomiast w przypadku LG K350N to Kotlin osiąga lepszy czas. Podobnie jak w przypadku czasu kompilacji, trudno jest jednoznacznie określić który język pozwala pisać szybsze programy.

6. Wnioski

W niniejszej publikacji dokonano analizy porównawczej języków programowania Java i Kotlin wykorzystywanych do tworzenia aplikacji mobilnych przeznaczonych do pracy pod kontrolą systemu Android. Na podstawie wyników badań można stwierdzić, że wybór języka programowania pomiędzy Javą a Kotlinem nie wpływa na obciążenie procesora. Aplikacje napisane w Kotlinie mogą jednak zużywać więcej pamięci. W przypadku prostych aplikacji, różnica powinna być niezauważalna. Może się ona jednak znacznie pogłębiać przy rozbudowanych projektach. Nie udało się sformułować jednoznacznych wniosków dotyczących czasów kompilacji i wykonania. Telefony marki Xiaomi osiągają lepsze czasy dla aplikacji napisanych w Javie. Natomiast w przypadku LG K350N to Kotlin pozwala szybciej kompilować i wykonywać programy.

Ponadto, Java zdaje się być lepszym wyborem dla poczatkujących programistów ze względu na większy zasób materiałów pomocniczych oraz większe wsparcie społeczności. Cechuje się też ona łatwiejszym do zrozumienia kodem. Kotlin jest natomiast językiem stosunkowo młodym, skierowanym do osób które mają już doświadczenie w programowaniu aplikacji mobilnych. Twórcy postanowili znacząco zredukować rozmiar kodu. Jest on również mniej intuicyjny, a część jego funkcjonalności pozostaje niewidoczna nawet dla programisty. Dlatego też będzie on lepszym wyborem dla doświadczonych osób poszukujących nowych, lepszych rozwiązań. Niemniej jednak ostatecznie, wybór powinien zależeć od indywidualnych preferencji programisty.

Języki Java i Kotlin są interoperacyjne, dlatego też nic nie stoi na przeszkodzie, aby dowolnie je mieszać w jednym projekcie. Mogą bez przeszkód korzystać z tych samych bibliotek i łączyć się z tymi samymi bazami danych. Z tego powodu następuje duża migrację projektów z Javy do Kotlina. Jako nowa alternatywa, Kotlin dostarcza wiele ciekawych i wygodnych rozwiązań, a jednocześnie pozostaje bardzo podobny do Javy. Można zatem się spodziewać, że w przyszłości zastąpi jej miejsce i stanie się najważniejszym językiem używanym do tworzenia aplikacji mobilnych. Trudno jest jednak oszacować, kiedy to nastąpi.

Literatura

[1] T. McDonnell, B. Ray, M. Kim: *An Empirical Study of API Stability and Adoption in the Android Ecosystem*, Texas 2013.

[2] Oficjalna dokumentacja języka Kotlin, https://kotlinlang.org/docs/reference/, Sierpień 2019

[3] P. Schwermer, *Performance Evaluation of Kotlin and Java on Android Runtime*, Sztokholm, Maj 2018.

[4] S. Bose, M. Mukherjee, A. Kundu i M. Banerjee, *A comparative Study: Java vs Kotlin Programming in Android Application Development*, International Journal of Advanced Research in Computer Science, Tom 9, Numer 3, Czerwiec 2018.

[5] M. Flauzino i inni: *Are you still smelling it?: A comparative study between Java and Kotlin language*, XII Sympozjum Brazylijskie dotyczące komponentów oprogramowania, architektury i ponownego użycia, s. 23-32, São Carlos, Wrzesień 2018.

[6] Oficjalna dokumentacja języka Java, https://docs.oracle.com/javase/8/docs, Kwiecień 2019.

[7] I. Kucherenko, A. Khan, *Hands-On Object-Oriented Programming with Kotlin*, Packt Publishing, Październik 2018.

[8] M. Devcic, *Kotlin Quick Start Guide: Core Features to Get You Ready for Developing Applications*, Packt Publishing, Sierpień 2018.

[9] K. Raghavendra Rao, *Kotlin for Enterprise Applications using Java EE: Develop, test, and troubleshoot enterprise applications and microservices with Kotlin and Java EE*, Packt Publishing, Listopad 2018.

[10] Oficjalna dokumentacja języka Android, https://developer.android.com/docs/, Kwiecień 2019.

JOURNAL of COMPUTER SCIENCES INSTITUTE

JCSI 13 (2019) 359-365
Wysłane: 2019-09-16
Przyjęte: 2019-09-23

Porównanie wydajności wybranych algorytmów oczyszczania pamięci w Wirtualnej Maszynie Javy

Igor Kopeć*, Jakub Smołka
Politechnika Lubelska, Instytut Informatyki, Nadbystrzycka 36B, 20-618 Lublin, Polska

Streszczenie. W językach z automatycznym zarządzaniem pamięcią ważną rolę pełni odśmiecacz pamięci - mechanizm odpowiedzialny za usuwanie nieużywanych obiektów z pamięci. Algorytmy odzyskiwania pamięci są rozwijane od wielu lat i dążą do zmaksymalizowania wydajności aplikacji. W niniejszym artykule przedstawiono i porównano wydajność pięciu algorytmów automatycznego zwalniania pamięci występujących w Javie w wersji 12 na trzech aplikacjach o różnym czasie życia obiektów. Analizie została poddana szybkość aplikacji, narzut pracy odzyskiwaczy pamięci oraz przepustowość aplikacji przy dużym obciążeniu.

Słowa kluczowe: odzyskiwanie pamięci; Wirtualna Maszyna Javy; wydajność aplikacji

*Autor do korespondencji.

Adres e-mail: igor.kopec@pollub.edu.pl

A performance comparison of garbage collector algorithms in Java Virtual Machine

Igor Kopeć*, Jakub Smołka
Institute of Computer Science, Lublin University of Technology, Nadbystrzycka 36B, 20-618 Lublin, Poland

Abstract. In programming languages with automatic memory management garbage collection plays an important role of cleaning unused memory. Garbage collection algorithms have been developed for many years and aim to maximize the application's performance. This paper presents and compares a performance of five garbage collection algorithms present in current version of Java 12 in three applications with different object lifetime span. The analysis covered the system responsiveness, garbage collector workload and application throughput at high application load.

Keywords: garbage collecting; Java Virtual Machine; application performance

*Corresponding author.

E-mail address: igor.kopec@pollub.edu.pl

1. Wstęp

Języki programowania stale się rozwijają i stawiają sobie za cel szybsze i efektywniejsze wytwarzanie oprogramowania. W coraz większym stopniu programiści mogą skupić się wyłącznie na dostarczaniu właściwych rozwiązań, gdyż niskopoziomowymi aspektami technicznymi zajmuje się niezawodny silnik aplikacyjny lub sam język. W językach z automatycznym zarządzaniem pamięcią ważną rolę pełni odśmiecacz pamięci - mechanizm odpowiedzialny za zapełnianie oraz usuwanie obiektów z pamięci. Duże ilości pamięci operacyjnej i coraz wydajniejsze procesory powodują, że programiści nie skupiają się na pisaniu wydajnego kodu do momentu, kiedy system zaczyna działać w nieoczekiwany sposób. Wtedy niezbędne staje się zrozumienie mechanizmów stojących za automatycznym zarządzaniem pamięcią jak i samych mechanizmów odzyskiwania pamięci.

Ze względu na pojawienie się nowej wersji Javy a w niej nowego odśmiecacza pamięci nasuwa się pytanie o ich wydajność, różnice w ich działaniu a także ich wpływ na aplikacje. W ramach szerszego porównania w niniejszym artykule zbadano wszystkie algorytmy odzyskiwania pamięci dostępne w Javie w wersji 12.

Artykuł ma za zadanie porównać wydajność i skalowalność pięciu odśmiecaczy pamięci na wirtualnej maszynie HotSpot na przykładzie trzech aplikacji o różnej budowie.

2. Przegląd literatury

Istnieje wiele badań poświęconych odzyskiwaniu pamięci w językach z automatycznym zarządzaniem pamięcią. Dużą część z nich stanowią prace skierowane w język Java i Wirtualną Maszynę Javy skupiające się na działaniu i wydajności zawartych w niej algorytmów odzyskiwania pamięci. Największą część badań skierowanych w problematykę odśmiecania pamięci były prace porównawcze. W składzie literatury, która pozwoliła przyjrzeć się bliżej problematyce odzyskiwania pamięci znalazły się prace porównujące wydajność konkretnych algorytmów i ich wpływ na działanie aplikacji bazując na predefiniowanych testach wzorcowych [2, 4, 5, 6, 7, 8, 9]. Przyjrzano się także pracom autorów implementacji odzyskiwaczy pamięci,

w których porównywano ich osiągnięcia względem istniejących już algorytmów [3, 11]. W kilku pracach badano zgodność algorytmów z teorią generacyjną [1], skalowalność poszczególnych algorytmów [4] a także badania skupiające się na wydajności odzyskiwania w procesach całkowitego odzyskiwania pamięci [6, 7, 10]. Żadna z prac nie uwzględniała wydajności algorytmów względem rozkładu czasu życia obiektów. Nie zbadano również zachowania algorytmów, które działają w aplikacjach działających wbrew teorii generacyjnej [12]. Temat skalowalności algorytmów także nie został dokładnie zbadany biorąc pod uwagę tylko liczbę dostępnych rdzeni procesora bez uwzględnienia zwiększania nakładu pracy.

3. Cel badań

Celem badania było zmierzenie wydajności pięciu algorytmów odzyskiwania pamięci w Javie w wersji 12 na podstawie trzech aplikacji: aplikacji tworzącej obiekty, które będą znajdować się w pamięci przez krótką chwilę, aplikacji tworzącej obiekty, z których większość nigdy nie zostanie usunięta z pamięci oraz aplikacji tworzącej obiekty, których „długość życia" – czyli liczba przetrwanych cykli odzyskiwania – będzie oscylować w granicach nie pozwalających na przeniesienie danego obiektu do innej części pamięci.

Kolejnym celem było zmierzenie skalowalności algorytmu. Na podstawie wyżej wymienionych aplikacji zbadano, jak zmienia się wydajność algorytmu odzyskiwania pamięci w przypadku stopniowego zwiększania nakładu pracy.

Porównanie miało także na celu wskazać, które algorytmy będą najbardziej odpowiednie do zastosowania w aplikacjach o konkretnym sposobie działania.

4. Pamięć w Wirtualnej Maszynie Javy

Pamięć w Wirtualnej Maszynie Javy podzielona jest na stertę (ang. *heap*) oraz na pamięć natywną (ang. *off-heap*). Pierwsza przechowuje obiekty i jest w zasięgu działania odzyskiwaczy pamięci. Jest tworzona przy starcie Wirtualnej Maszyny Javy i rośnie wraz z liczbą utworzonych obiektów. Sterta jest głównym przedmiotem zainteresowań programistów, gdyż to ona ma największy wpływ na wydajność. Druga zaś przechowuje dane wspomagające pracę całej aplikacji nie będących obiektami i jest poza zasięgiem automatycznego odzyskiwania. Do pamięci natywnej zaliczamy tzw. Meta Space, schowek kompilatora JIT, stosy wątków oraz współdzielone biblioteki.

Sterta dzielona jest na przestrzeń młodą (ang. *young*) i starą (ang. *old*) i jest skutkiem zastosowania hipotezy generacyjnej [1]. Dalej pamięć młoda podzielona została na eden i dwie przestrzenie przejściowe (nazywane przetrwalnikowymi). Eden to miejsce, w którym umieszczane są obiekty nowo utworzone. Nowe obiekty pozostają w przestrzeni eden do momentu pierwszego cyklu odzyskiwania pamięci. Obiekty które po pierwszym cyklu nie zostały usunięte, zostają przeniesione do kolejnego segmentu pamięci młodej - pierwszej przestrzeni przetrwalnikowej (ang. *survivor 0*). Przetrwanie obiektu w kolejnych cyklach czyszczenia pamięci skutkuje jego przeniesieniem do kolejnych przestrzeni przetrwalnikowych aż do momentu w którym obiekt jest uznawany za obiekt stary, po czym zostaje wypromowany do starszej generacji. Przestrzeń starej generacji (zwana także przestrzenią *tenured*) jest przestrzenią ciągłą, w której znajdują się najdłużej żyjące obiekty aplikacji.

Pamięć natywna jest częścią pamięci zarządzaną głównie przez wirtualną maszynę. Dalsza dekompozycja pamięci natywnej stanowi rozdzielenie na obszary wspólne dla całej wirtualnej maszyny oraz prywatne obszary pojedynczych wątków. Pierwszą przestrzenią jest *Meta Space*, gdzie znajdują się podprzestrzenie zawierające definicje klas, metod i pól, literały znakowe, stałe i referencje oraz kody klas. Nie posiada górnej granicy wielkości pamięci - ogranicza ją tylko pamięć systemu operacyjnego. Kolejną, ważną z punktu widzenia wydajności jest przestrzeń schowka kompilatora JIT , która to stanowi kod zoptymalizowany przez kompilator JIT. Ostatnią częścią pamięci natywnej jest przestrzeń wykorzystywana przez poszczególne wątki, która składa się ze stosu (ang. *stack*) oraz z licznika operacji (ang. *program counter*). Stosem jest kolejka LIFO zawierająca ramki - tablicę zmiennych lokalnych oraz wartości zwracane przez wywoływane metody. Licznik operacji natomiast posiada referencję do aktualnie wykonywanej instrukcji. Każdy wątek posiada swój stos wywołań metod.

Empiryczne badania [1] dowiodły, iż najwięcej jest obiektów najkrócej żyjących, natomiast odwołania do starych obiektów są stosunkowo rzadkie. Obserwacje te są określane mianem hipotezy generacyjnej i implikują podział pamięci na starą i młodą generację ze względu na ich odmienną charakterystykę. Segmentacja pamięci pozwala na zastosowanie różnych algorytmów odzyskiwania pamięci do różnych generacji. Każdy algorytm jest zoptymalizowany pod konkretne właściwości danej generacji. Podział pozwala także na selektywne uruchamianie sprzątania dla określonych segmentów.

5. Algorytmy odzyskiwania pamięci

Według oficjalnej dokumentacji w Javie 12 istnieje pięć odśmiecaczy pamięci - Szeregowy (ang. *Serial*), Równoległy (ang *Parallel*), CMS (*Concurrent Mark Sweep*), G1 (*Garbage First)* i Shenandoah.

Algorytm szeregowy

Algorytm szeregowy (ang. *Serial*) jest najprostszym algorytmem działającym tylko w jednym wątku. Nie uzyskuje lepszej wydajności na maszynach o większej liczbie rdzeni procesora, ale jest dobrym wyborem w środowisku jednowątkowym oraz wtedy, gdy na jednym komputerze liczba działających Wirtualnych Maszyn jest równa lub większa liczbie dostępnych wątków procesora. Używając tego algorytmu w środowisku wielowątkowym blokujemy pracę innych wątków. Ten odzyskiwacz pamięci zatrzymuje całą aplikację na czas swojego działania [14].

Algorytm równoległy

Algorytm równoległy (ang. *Parallel*) jest podobny do algorytmu szeregowego ale różni go to, że do odzyskiwania używa wielu wątków. Zarówno młoda i stara generacja czyszczona jest równolegle. Pomimo tego, iż algorytm na czas działania zatrzymuje całą aplikację zrównoleglenie pracy powoduje, że przerwy są proporcjonalnie krótsze. Mając dostępnych X rdzeni, odśmiecacz użyje X wątków. Do najefektywniejszej pracy kolektor potrzebuje przynajmniej trzech wątków - wtedy dopiero zauważalna jest różnica pomiędzy algorytmem szeregowym [14].

Algorytm Concurrent Mark Sweep

Algorytm Concurrent Mark Sweep (CMS) jest algorytmem współbieżnym i próbuje wykonywać większą część swojej pracy nie przerywając działania aplikacji. Osiąga to przez to, iż w starej generacji stara się nie używać częstego kopiowania i kompaktowania obiektów, tylko coraz bardziej zwiększa dostępną pamięć dla aplikacji. W przypadku gdy ilość zajmowanego miejsca jest dostatecznie duża wykonywane jest całkowite czyszczenie kopiująco-kompaktujące. Algorytm większość pracy wykonuje równolegle z działaniem aplikacji. Niestety wykonując odzyskiwanie współbieżnie, wątki algorytmu zabierają część czasu procesora wątkom aplikacji, więc wydajność aplikacji używającej tego algorytmu zmniejsza się. Stanowi to dobry kompromis, jeśli aplikacja nie powinna mieć większych przestojów [14].

Algorytm G1

Algorytm G1 (ang. *Garbage first*) działa na podobnej zasadzie co algorytm CMS lecz jest jeszcze bardziej współbieżny i stosuje odmienny podział generacyjny sterty niż wcześniej wymienione algorytmy. Pozwala programiście samemu określić czas pauzy w każdym cyklu odzyskiwania. Sterta dla tego algorytmu nie jest już dzielona na młodą i starą generację lecz na regiony o takiej samej wielkości. W zależności od wielkości dostępnej pamięci wielkość regionu waha się od 1MB do 32MB po to, aby liczba regionów wynosiła 2048. Wyróżnia się pięć rodzajów regionów - *eden, przetrwalnikowy, ogromny, stara generacja* i *nieużywany*. Podobnie jak w innych algorytmach, nowe obiekty alokowane są w edenie. Gdy się starzeją przechodzą do regionu przetrwalnikowego, a następnie do regionu starej generacji. Obiekty, których wielkość przekracza 50% miejsca regionu alokowane są w regionie ogromnym - jest to sytuacja dość rzadka. Regiony nieużywane czekają na przydzielenie im jednej z pozostałych ról oddając nieużywaną pamięć systemowi operacyjnemu [14].

Algorytm Shenandoah

Algorytm Shenandoah jest algorytmem redukującym czas pauz do minimum poprzez wykonywanie większości swojej pracy współbieżnie do działania aplikacji. Zrównolegleniu uległa tutaj również faza kompaktowania obiektów, wskutek czego czasy pauz nie zależą już dłużej od wielkości dostępnej pamięci. Algorytm stworzono z myślą o stertach osiągających wielkość powyżej 100GB. Jednym z założeń tego algorytmu jest osiąganie czasu trwania wszystkich pauz odzyskiwania nie większych niż 10 ms. dla małych oraz dla dużych obszarów pamięci. Co za tym idzie, Shenandoah może być używany tam, gdzie najczęściej używane dziś algorytmy (CMS, G1) po prostu nie są wystarczająco wydajne. Z drugiej strony Shenandoah jest mało wydajny dla małych aplikacji [15].

Podział pamięci na regiony wygląda podobnie jak w algorytmie G1. Zasada działania jest również podobna. Różnicą jest to, że Shenandoah nie przestrzega zasady, iż większość obiektów umiera szybko, ponieważ nie wszystkie aplikacje działają zgodnie z tym założeniem. Algorytm nie dzieli stosu na młodą i starą generacje. Region może zawierać zarówno młode jak i stare obiekty. W związku z tym proces odzyskiwania jest zawsze procesem czyszczenia całego stosu (ang *Full garbage collection*) [15].

6. Metoda badawcza

W celu przeprowadzenia badań stworzono trzy aplikacje, które różniły się rozkładem czasu trwania obiektów w pamięci. Każda aplikacja została przetestowana używając wszystkich algorytmów oczyszczania pamięci wspomnianych w rozdziale piątym w ich podstawowej konfiguracji. Aplikację testowano pod kątem czasu wykonanego testu, oraz *przepustowości* aplikacji - stosunku czasu działania aplikacji do czasu pauzy systemu związanej z pracą odzyskiwacza pamięci. Prześledzono także działania algorytmu odzyskiwania pamięci. Ponadto zbadano skalowalność każdej aplikacji dla poszczególnych algorytmów oczyszczania stopniowo zwiększając nakład pracy. Wszystkie testy poprzedzono pięcioma iteracjami próbnymi w celu „rozgrzania" Wirtualnej Maszyny Javy. W celu uzyskania poprawnego wyniku, każdy przypadek testowy został wykonany dziesięć razy. Następnie wyniki zostały uśrednione.

Wyniki uzyskano analizując zapis plików dziennika generowanych przez algorytm odzyskiwania pamięci oraz poprzez dostarczany wraz z Javą program *jStat* [13] służący do monitorowania stanu pamięci aplikacji.

6.1. Opis aplikacji testowych

Pierwszą aplikację zaprojektowano tak, aby tworzyła bardzo dużo małych obiektów. Aplikacja sprawdza, jak algorytm poradzi sobie z odzyskiwaniem dużej liczby obiektów w młodej generacji.

Listing nr 1 przedstawia test aplikacji tworzącej obiekty krótko żyjące. Test dodaje do listy podaną liczbę obiektów klasy akumulator. Klasa akumulator posiada zmienną typu Long i metodę dodającą do akumulatora losową wartość liczbową. Po wyjściu z funkcji zmienna *rand* jest uznawana jako obiekt martwy. W następnym kroku, lista akumulatorów jest iterowana a na każdym z nich wykonywana jest funkcja dodająca losową wartość. W taki sposób test tworzy 1,5 miliona małych obiektów. Następnie wartości wszystkich obiektów są sumowane do zmiennej *sum*. Na końcu test usuwa wszystkie elementy z listy.

Przykład 1. Listing przedstawiający kod testu pierwszej aplikacji

```
public void startTest() {
  List<Accumulator> accumulators = new LinkedList<>();
  //dodanie obiektów Akumulator do listy
  for (int i = 0; i < numberOfAccumulators; i++) {
     accumulators.add(new Accumulator());
  }

  //wywołanie 10000 razy funkcji addValue() akumulatora
  AtomicLong tempSum = new AtomicLong();
  accumulators.forEach(accumulator -> {
     //3000 wywołań funkcji
     for (int i = 0; i < numberOfAddedValues; i++) {
        tempSum.addAndGet(accumulator.addValue());
     }
  });

    long sum = accumulators.stream()
        .mapToLong(Accumulator::getAccumulatedAmount)
        .sum();
    BigInteger bigInteger = new
    BigInteger(String.valueOf(sum));
    Iterator<Accumulator> it = accumulators.iterator();
    while (it.hasNext()) {
      Accumulator next = it.next();
      it.remove();
  }
}
```

Drugą aplikację zbudowano tak aby przechowywała dużo obiektów w pamięci jak najdłużej po to, aby wymusić promocję obiektów do starej generacji.

Test aplikacji tworzącej obiekty długożyjące przedstawiono na listingu nr 2. Celem testu jest stworzenie określonej liczby węzłów w drzewie. W pętli na przemian dodawanych jest 50 dużych obiektów do binarnego drzewa. Następnie usuwany jest losowo jeden węzeł. Usunięcie węzła powoduje, że jego potomkowie także są usunięci z węzła. Wymusza to proces odzyskiwania pamięci. Elementy które są najbliżej korzenia drzewa mają dużą szansę na przetrwanie dostatecznie długo, aby zostać zakwalifikowane do przeniesienia do starej generacji. Ciągłe tworzenie obiektów i alokacja w starej generacji wymusza proces oczyszczania nie tylko młodej ale i starej generacji stosu.

Przykład 2. Listing przedstawiający kod testu drugiej aplikacji

```
public void startTest() {
  try {
     while (nodeCounter.get() < maxNumberOfNodes) {
        //dodanie elementów
        int randomInt =
    ThreadLocalRandom.current().nextInt(50);
    for (int i = 0; i < randomInt; i++) {
    Node node = new
    Node(nodeCounter.getAndIncrement());
      tree.insert(node);
      overallSum += (Long) node.getResultValue();
  }
        //usunięcie węzła
  Iterator iterator = tree.inorderIterator();
  ArrayList<Node> elements = new ArrayList<>();
  while (iterator.hasNext()) {
     elements.add((Node) iterator.next());
  }
  double rand = ThreadLocalRandom.current().nextDouble(0,
1.01);
  Node element;
  if (rand < 0.05) {
     element = chooseElement(elements, lastXPercent(0.4));
  } else if (rand < 0.10) {
     element = chooseElement(elements, lastXPercent(0.2));
  } else {
     element = chooseElement(elements,
this::cut100OrTenPercent);
  }
  tree.delete(element);

     }
  } catch (InterruptedException e) {
     e.printStackTrace();
  } finally {
     log.debug("Sum: {}", overallSum);
  }
}
```

Trzecią aplikacje skonstruowano tak, aby pracę wykonywała na wielu wątkach w celu zwielokrotnienia liczby wykonywanych operacji oraz liczby alokowanych obiektów o średnim czasie życia.

Aplikacja tworzy obiekty które od razu kwalifikują się do odzyskania przez algorytm odzyskiwania pamięci oraz takie, które w tej pamięci pozostają na pewien czas. Test został skonstruowany tak, aby długość życia danego obiektu oscylowała w granicach progu zakwalifikowania obiektu do przeniesienia do starej generacji. W praktyce heurystyki odśmiecacza pamięci nie są w stanie jednoznacznie stwierdzić czy obiekty będą przeniesione do starej generacji pamięci, więc algorytm nie może zastosować odpowiednich optymalizacji. Test ma za zadanie sprawdzić zachowanie algorytmu w aplikacji działającej wbrew teorii generacyjnej. Każdy wątek wykonuje takie samo zadanie przedstawione w listingu nr 3. Badanie wykonano na czterech wątkach - taką liczbę wątków posiadał komputer na którym przeprowadzono badania.

Przykład 3. Listing przedstawiający kod testu trzeciej aplikacji

```
public Object doOperation() {
   char[] chars = getCharArray(objectSize);
   List<String> liveList = new
ArrayList<String>(myNumLive);

  // Utworzenie listy
  for (int i = 0; i < myNumLive; i++) {
     liveList.add(new String(chars));
  }
  for (int j = 0; j < 5_000_000; j++) {
     // Tworzymy dużą liczbę obiektów
     for (int i = 0; i < fractionLive; i++) {
        String garbageObject = new String(chars);
     }
     // W losowym miejscu zamieniamy obiekty w liście
     // na inne
     int index = (int) (Math.random() * myNumLive);
     liveList.set(index, new String(chars));
  }
  return liveList;
}
```

Dla powyższych aplikacji wykonano podane scenariusze w dwóch przypadkach. Pierwszy zakładał użycie stałych parametrów dla wszystkich algorytmów w celu zbadania wydajności. Były to odpowiednio:

- 500 000 obiektów akumulator i 3 000 operacji na każdym obiekcie,

- 100 000 dodanych węzłów do drzewa. 50 obiektów dodawanych w każdym cyklu,
- Cztery wątki aplikacyjne.

Drugi przypadek badał skalowalność aplikacji i zakładał użycie zmiennych parametrów odpowiednio:

- 10 000 - 12 400 000 obiektów akumulator i 3 000 operacji na każdym obiekcie,
- 100 000 dodanych węzłów do drzewa i 100 - 50 000 obiektów dodawanych w każdym cyklu,
- 1 - 8 wątków aplikacyjnych.

7. Analiza porównawcza

W analizie porównano:

- Całkowity czas testu i przepustowość aplikacji,
- Narzut czasu odzyskiwania i liczbę procesów odzyskiwania dla generacji młodej i starej,
- Średni i maksymalny czas pauz systemu,
- Skalowalność poszczególnych algorytmów.

W tabeli 1, 2 i 3 został przedstawiony średni czas wykonania całego testu wydajnościowego na trzech aplikacjach a także przepustowość aplikacji dla poszczególnych algorytmów.

Tabela 1. Czas wykonania testu i przepustowość aplikacji dla poszczególnych algorytmów dla pierwszej aplikacji

Algorytm	Czas testu [s]	Przepustowość
Szeregowy	43,83	96,66%
Równoległy	46,82	99,28%
CMS	44,68	98,38%
G1	53,74	98,13%
Shenandoah	48,33	98,72%

Tabela 2. Czas wykonania testu i przepustowość aplikacji dla poszczególnych algorytmów dla drugiej aplikacji

Algorytm	Czas testu [s]	Przepustowość
Szeregowy	51,87	96,74%
Równoległy	52,02	97,67%
CMS	49,09	98,40%
G1	51,87	98,99%
Shenandoah	58,77	88,14%

Tabela 3. Czas wykonania testu i przepustowość aplikacji dla poszczególnych algorytmów dla trzeciej aplikacji

Algorytm	Czas testu [s]	Przepustowość
Szeregowy	38,2	21,51%
Równoległy	35,8	40,65%
CMS	25,0	62,70%
G1	24,9	77,76%
Shenandoah	90,4	21,27%

Tabele 4, 5 i 6 przedstawiają całkowity czas poświęcony na odzyskiwanie pamięci oraz liczbę procesów odzyskiwania pamięci. Poniższe wyniki zostały rozdzielone na młodą i starą generację.

Tabela 7, 8 i 9 przedstawia średni czas pauz występujących w danych testach wydajnościowych oraz maksymalny zarejestrowany czas pauzy w młodej i starej generacji pamięci.

Tabela 4. Narzut czasu i liczba odzyskiwań dla poszczególnych algorytmów dla pierwszej aplikacji

Algorytm	Narzut czasu odzyskiwania [s]		Liczba procesów odzyskiwania	
	młoda gen.	stara gen.	młoda gen.	stara gen.
Szeregowy	1,929	0,000	124,30	0,0
Równoległy	0,446	0,000	103,90	0,0
CMS	1,095	0,000	127,4	0,0
G1	1,717	0,000	113,60	0,0
Shenandoah	0,000	10,216	0,0	404,20

Tabela 5. Narzut czasu i liczba odzyskiwań dla poszczególnych algorytmów dla drugiej aplikacji

Algorytm	Narzut czasu odzyskiwania [s]		Liczba procesów odzyskiwania	
	młoda gen.	stara gen.	młoda gen.	stara gen.
Szeregowy	3,293	1,845	93,7	15,4
Równoległy	2,969	1,850	128,6	17,4
CMS	3,007	0,493	473,4	54,5
G1	2,530	0,000	328,9	0,0
Shenandoah	0,000	2,835	0,0	153,3

Tabela 6. Narzut czasu i liczba odzyskiwań dla poszczególnych algorytmów dla trzeciej aplikacji

Algorytm	Narzut czasu odzyskiwania [s]		Liczba procesów odzyskiwania	
	młoda gen.	stara gen.	młoda gen.	stara gen.
Szeregowy	26,688	7,199	66,6	4,2
Równoległy	20,181	3,472	287,1	866,2
CMS	11,415	0,989	227,7	4,1
G1	16,303	9,687	61,0	1,3
Shenandoah	0,000	36,116	0,0	489,7

Tabela 7. Maksymalny i średni czas pauz dla poszczególnych algorytmów dla pierwszej aplikacji

Algorytm	Średni czas pauz [ms]	Maksymalny czas pauzy [ms]	
		młoda gen.	stara gen.
Szeregowy	15,29	212,7	0,0
Równoległy	0,45	471,8	0,0
CMS	8,58	619	0,0
G1	15,26	198,3	0,0
Shenandoah	0,80	0,0	4,41

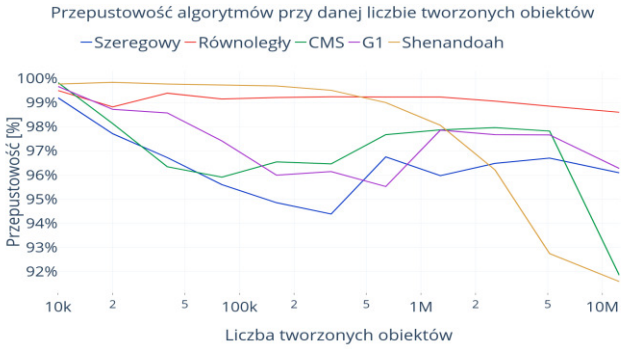
Tabela 8. Maksymalny i średni czas pauz dla poszczególnych algorytmów dla pierwszej aplikacji

Algorytm	Średni czas pauz [ms]	Maksymalny czas pauzy [ms]	
		młoda gen.	stara gen.
Szeregowy	56,0	128,7	334,2
Równoległy	33,9	100,1	247,9
CMS	7,9	49,8	241,2
G1	6,8	69,2	0,0
Shenandoah	1,0	0,0	6,2

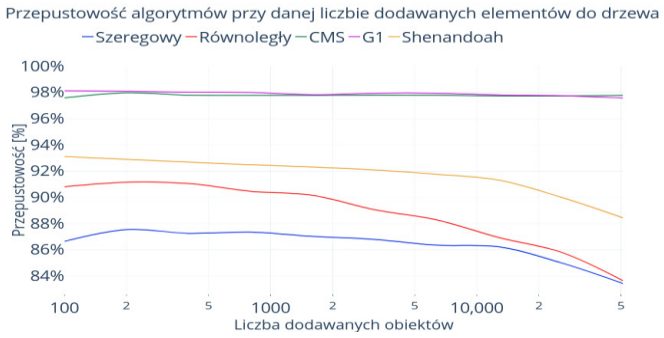
Tabela 9. Maksymalny i średni czas pauz dla poszczególnych algorytmów dla pierwszej aplikacji

Algorytm	Średni czas pauz [ms]	Maksymalny czas pauzy [ms]	
		młoda gen.	stara gen.
Szeregowy	480,0	4502,2	4252,7
Równoległy	320,0	501,4	1524,4
CMS	49,4	1361,0	58,1
G1	106,3	185,2	0,0
Shenandoah	5,2	0,0	25,8

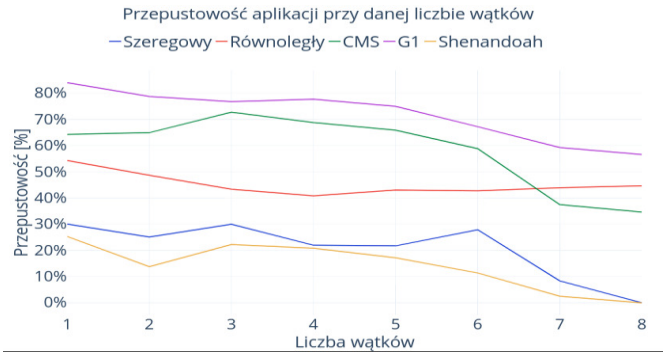
W ramach testów wydajnościowych trzech aplikacji, zbadano także zmianę przepustowości algorytmów na zwiększające się obciążenie pracą. Wyniki testów skalowalności przedstawia rysunek 1, 2 i 3.



Rys. 1. Wykres przedstawiający zmieniającą się przepustowość przy danej liczbie tworzonych obiektów w aplikacji pierwszej.



Rys. 2. Wykres przedstawiający zmieniającą się przepustowość przy danej liczbie dodawanych elementów w aplikacji drugiej.



Rys. 3. Wykres przedstawiający zmieniającą się przepustowość przy danej liczbie wątków roboczych w aplikacji trzeciej.

8. Wnioski

Wyniki pokazują, jak bardzo wykonanie testu różniło się w przypadku zastosowania różnych algorytmów odzyskiwania pamięci. Algorytmy współbieżne CMS i G1 charakteryzują się najwyższą przepustowością i dobrą skalowalnością i warto z nich korzystać niemal we wszystkich typach aplikacji. We wszystkich przypadkach algorytmy współbieżne osiągały bardzo krótkie pauzy, zazwyczaj o długości niezauważalnej przez użytkownika aplikacji. Ich doskonały wynik zawdzięczany jest wysokiej współbieżności, co pozwalała im na działanie wraz z działającą aplikacją. Cechują ich także niższe maksymalne pauzy i liczba procesów odzyskiwania pamięci. Algorytmy CMS i G1 najlepiej poradziły sobie z testem aplikacji drugiej i trzeciej. W pierwszej natomiast ustąpiły algorytmowi równoległemu. Algorytmy współbieżne przystosowane są do efektywnego odzyskiwania starej generacji. Ze względu, że wyniki obu algorytmów są podobne, ciężko jest wskazać który algorytm sprawdza się lepiej. CMS osiągał mniejsze czasy pauz, natomiast w przypadku użycia algorytmu G1 przepustowość aplikacji była wyższa.

Najmniejsze czasy pauz osiągał najnowszy algorytm Shenandoah, który niezależnie od rodzaju aplikacji czy obciążenia potrafił osiągać stałe wartości pauz nie przekraczające średniej wartości 10 ms. Najdłuższa odnotowana pauza nie przekraczała 25 ms., kosztem wolniejszego wykonania testu i zmniejszonej przepustowości.

Algorytmy nie współbieżne takie jak szeregowy czy równoległy mogą natomiast z powodzeniem być wykorzystywane w aplikacjach w których nie przechowuje się sporej liczby obiektów w pamięci na długi czas, a także w środowiskach jednoprocesorowych. Sporą wadą tych algorytmów jest to, że powodują one nieuniknione długie pauzy systemu najbardziej zauważalne przy odzyskiwaniu starej generacji. Z wyników można odczytać, że algorytm równoległy jako algorytm odzyskujący pamięć w wielu wątkach, powinien być lepszym wyborem od algorytmu szeregowego w każdym przypadku.

Analizując wyniki trzech pierwszych aplikacji stwierdzono, że algorytmy działają poprawnie w aplikacjach, które podążają za teorią generacyjną - większość obiektów jest szybko niepotrzebna a obiektów znajdujących się w pamięci dłużej jest mniej. Dobrze to widać na przykładzie aplikacji trzeciej, gdzie czas życia obiektów został uśredniony. Wyniki trzeciej aplikacji pokazują, że heurystyki algorytmów nie są dobrze zoptymalizowane do tego typu aplikacji.

Literatura

[1] Appel A. W.: Simple generational garbage collection and fast allocation. Software: Practice and Experience, nr 2/1989, Tom 19, s. 171-183.

[2] Carpen-Amarie M., Marlier P., Felber P., Thomas G.: A Performance Study of Java Garbage Collectors on Multicore Architectures, Proceedings of the Sixth International Workshop on Programming Models and Applications for Multicores and Manycores, 2015, s. 20-29.

[3] Detlefs D., Flood C., Heller S., Printezis T.: Garbage-first garbage collection. In Proceedings of the 4th international symposium on Memory management. ACM. 2004, s. 37-48.

[4] Gidra L., Thomas G., Sopena J., Shapiro M.: Assessing the scalability of garbage collectors on many cores. In Proceedings of the 6th Workshop on Programming Languages and Operating Systems, ACM, 2011, s. 7.

[5] Grgic H., Mihaljevic B., Radovan A.: Comparison of garbage collectors in Java programming language, 41st International Convention on Information and Communication Technology, Electronics and Microelectronics, 2018.

[6] Li H., Wu M., Chen H.: Analysis and Optimizations of Java Full Garbage Collection, Proceedings of the 9th Asia-Pacific Workshop on Systems, 2018, s. 18.

[7] Li H., Wu M., Zang B., Chen H.: ScissorGC: Scalable and Efficient Compaction for Java Full Garbage Collection, 2019.

[8] Suo K., Rao J., Jiang H., Srisa-an W.: Characterizing and optimizing hotspot parallel garbage collection on multicore systems. EuroSys, 2018, s. 35-1.

[9] Tauro C., Prabhu M., Saldanha V.: CMS and G1 Collector in Java 7 Hotspot: Overview, Comparisons and Performance Metrics. International Journal of Computer Applications, 43(11), 2012.

[10] Yu Y., Lei T., Zhang W., Chen H., Zang B.: Performance analysis and optimization of full garbage collection in memory-hungry environments. In ACM SIGPLAN Notices, ACM. Nr 7/2016, Tom 51, s. 123-130.

[11] Flood C. H., Kennke R., Dinn A., Haley A., Westrelin R.: Shenandoah: An open-source concurrent compacting garbage collector for openjdk. In Proceedings of the 13th International Conference on Principles and Practices of Programming on the Java Platform: Virtual Machines, Languages, and Tools, ACM, 2016, s. 13.

[12] Struktura pamięci Wirtualnej Maszyny Java , https://www.oracle.com/technetwork/java/javase/memorymanagement-whitepaper-150215.pdf [24.03.2019].

[13] Dokumentacja programu jStat, https://docs.oracle.com/javase/7/docs/technotes/tools/share/jstat.html [24.03.2019].

[14] Hunt C., Beckwith M., Parhar P., Rutisson B.: Java Performance Companion, Addison-Wesley Professional, 2016.

[15] Opis algorytmu Shenandoah https://wiki.openjdk.java.net/display/shenandoah/Main 24.03.2019.

JOURNAL of COMPUTER SCIENCES INSTITUTE

JCSI 13 (2019) 366-370
Wysłane: 2019-12-06
Przyjęte: 2019-12-30

Innovative applications of digital solutions and tools in IT education

Michalina Gryniewicz-Jaworska
Department of Computer Science, Lublin University of Technology, Nadbystrzycka 36B, 20-618 Lublin, Poland

Abstract. Currently, digital skills have become an important factor for the development and active participation in today's information society. The article describes innovative IT methods and tools used in the education process. New technologies and new methods of conducting classes form the basis of today's education. Traditional methods have been replaced by digital tools that are perfect at the stage of educating school students in IT profiles, preparing them for vocational exams. The article compares the results of school students from vocational exams at the Technical College of Lublin with the results from the entire Lubelskie Voivodeship.

Keywords: Information and Communication Technology, digital tools, IT education

E-mail address: m.jaworska@pollub.pl

Innowacyjne zastosowania rozwiązań i narzędzi cyfrowych w kształceniu uczniów szkół informatycznych

Michalina Gryniewicz-Jaworska
Politechnika Lubelska, Katedra Informatyki, Nadbystrzycka 36B, 20-618 Lublin, Polska

Streszczenie. Aktualnie cyfrowe umiejętności stały się istotnym czynnikiem rozwoju i aktywnego uczestnictwa w dzisiejszym społeczeństwie informacyjnym. W artykule opisano innowacyjne metody oraz narzędzia informatyczne wykorzystywane w procesie kształcenia. Nowe technologie oraz nowe metody prowadzenia zajęć lekcyjnych stanowią podstawę dzisiejszej edukacji. Tradycyjne metody zostały zastąpione cyfrowymi narzędziami, które doskonale sprawdzają się na etapie kształcenia młodzieży szkolnej w szkołach o profilu informatycznym, przygotowujących młodzież do egzaminów zawodowych. W artykule porównano wyniki egzaminów zawodowych uczniów Technikum Informatycznego w Lublinie z wynikami uczniów z całego województwa lubelskiego.

Słowa kluczowe: Technologie informacyjne i komunikacyjne, narzędzia cyfrowe, edukacja informatyczna

Adres e-mail: m.jaworska@pollub.pl

1. Introduction

It is very likely that within a dozen or so years, information technology will become an integral part of both private and professional life for most people. Without the skills related to use of this technology, it will soon be impossible to perform most lucrative professions, replenish one's qualifications or acquire new ones. Therefore the school must prepare its graduates for the proficient use of computer and telecommunication hardware and software. The article currently describes the use of IT tools in education in both high schools and colleges. The author further describes the methods he used in working with students to prepare them for qualification exams. The previously used traditional teaching methods resulted in unsatisfactory results. The implementation of new tools and methods of working with students confirms their beneficial impact on the achieved exam results. A modern school should make information technology a means to support the education of young people in the subjects of teaching. The tools offered by this technology should become typical didactic means. Currently, one of the many modern forms of education is e-learning. Many solutions derived from this form of education can successfully support traditional school lessons, one can safely state that people are dealing with blended learning, which combines the advantages of traditional teaching and e-learning. The key task of a modern teacher is the proper selection of work methods based on the use of digital tools[5]. To facilitate the task of educators, Ruben Puentedura developed the Substitution Augmentation Modification (SAMR) model that defines the various stages of implementing new technologies into the learning process. The proposed model describes how a teacher can change the course of an educational process by using new information technologies as a substitute for traditional methods.

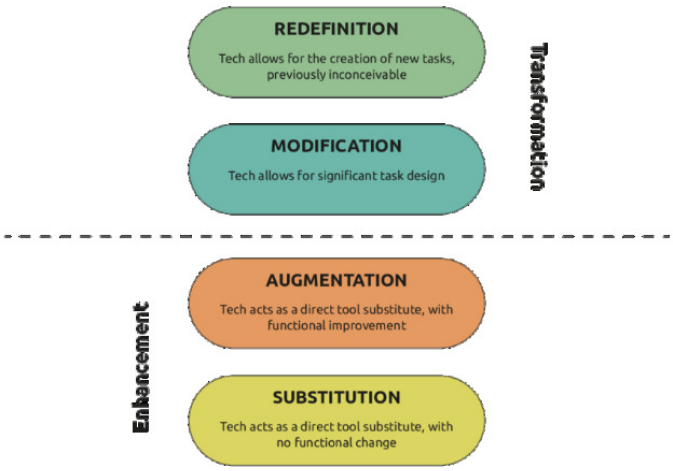


Fig. 1. Graphic presentation of the respective stages of implementing new technologies into the didactic process [5][5]

The first stage is substitution. At this stage new technologies are used for tasks that were previously performed without the presence of these tools. There is therefore no change in functionality. The next stage is augmentation. At this stage, we use IT tools (e. g. tests, educational games, quizzes to test the knowledge available on mobile devices) in a way that is attractive to the student. This method perfectly activates the student and speeds up the teaching process. The third stage is a modification. At this level of implementation, the use of new information technologies for educational purposes is becoming a necessary and irreplaceable element. A simple example is a creation by a student of a multimedia presentation on a topic specified by the teacher and sending it electronically. The last stage of implementing new technologies in the didactic process is called Redefinition. At this stage, we are dealing with a transformation of the education process, closely related to the degree to which new technologies have been implemented in the education process [11].

2. E-learning

The development of new technologies, and in particular, the dissemination of access to the Internet, is conducive to introducing new methods to the existing model of school education. Research conducted in schools shows that the activity of students in the classroom is usually limited to noting down information from the blackboard or textbook, listening to the teacher's long speeches or noting them down. It turns out that young people prefer to learn in groups by performing practical activities using a computer. New technologies can support the forms of education preferred by students, but traditional methods of education and development should not be abandoned[8]. According to the "E-learning trends 2019" report, e-learning is the most widely used educational tool in North America and Western Europe, less popular in the eastern countries of the European Union and Asia [2].

E-learning enables distance learning without having to leave home. It is enough to have a computer with Internet access and one can become a member of a virtual school. E-learning is today an educational norm, it fits in with the vision of an innovative school that meets the expectations of a student whose natural environment is the Internet. A feature that has significantly influenced the success of e-learning is the flexibility of teaching. It is possible to study at any place and time, to use up-to-date educational materials adapted to our own individual needs. It should be emphasized that e-education allows for equal educational opportunities for disabled students. The use of e-learning requires certain skills from the student, the student should be fluent in computer use and telecommunications techniques. In the case of e-learning, the student is forced to be more independent and disciplined at work, while the teacher has limited contact with a group of students. New technologies work well in primary schools, but traditional teaching should be continued in this stage of education [9].

3. M-learning

The key to modern education is not only the technology but also communication. According to research, people learn best in a group, a community, when they are connected in various ways. A community is created and strengthened when people cooperate, research and create together in a team. Therefore, any technological solution proposed to the university or school must include communication. If it does not, it becomes pedagogically useless. Currently, more and more is expected from educational methods using new technologies such as m-learning or e-learning. Mobile devices have become part of our everyday life, with their help one can learn and work. The dynamic development of mobile technologies means that these devices are increasingly becoming personal computers in terms of functionality and performance. With mobile devices, the necessary resources become available anywhere, allowing one to combine work with acquiring knowledge. It is possible to state unequivocally that the use of mobile devices in teaching responds to the educational needs of the modern world. The definition of mobile learning consists of all learning using mobile devices only, as well as learning on mobile devices in combination with other technologies. In m-learning, the most important aspect is learning in motion at any place and time. Unlike e-learning, m-learning is a dynamic way of learning. Mobile devices can be used literally anywhere: at home, at school, on public transport or on a walk. Bearing in mind that learning on a small smartphone screen can be exhausting after some time, it is necessary to develop teaching materials and content as concise and legible as possible so that the student can acquire the necessary knowledge quickly and efficiently. In m-learning, audio or video podcasts, quizzes, language learning applications, pdf files or audiobooks are most commonly used. Multifunctionality of mobile devices allows for communication and cooperation with other users, people learning through social networks, MMS or text messages. However, the most advanced form of m-learning is the use of functionalities such as mediascapes and QR codes to create context- and location-oriented training modules. QR codes, called Quick Response codes, are added to physical objects that are read by devices equipped with cameras. The developed teaching materials are made available through the automatic start of the web browser. Mediascapes are a form of media that creates interactive experiences by combining digital images, sounds, and interactions with the physical world. In education, however, m-learning can be treated as a complementary method of traditional teaching, but it works perfectly for learning specific competencies, as additional support for lessons, courses or large e-learning courses. However, it is necessary to remember that m-learning must be adapted to the capabilities of a mobile phone. It cannot take up too much memory and the courses should be divided into short thematic blocks. In Polish schools, m-learning is treated as a new source of supplementing and systematizing knowledge acquired in class.

Table 1 List of mobile devices used in m-learning[10],[11]

Device	Definition
PDA (personal data assistants)	PDA works as an individual digital secondary device which is generally small in size and able to play wide range of multimedia files
Cellular phone	It permits operators to converse with one another at anyplace and at any time. Access to the internet via WAP or GPRS technology may also be done through cellular phones
Smart phone	This device integrate mobile phone capabilities with the more common features of a handheld computer or PDA. It is furnished with internet usage and the capability to support multimedia files
3G phone	3rd generation of mobile phone that has the capability to transmit four times better than the normal cellular phones
4G phone	4G is the original and enhanced version of 3G. Internet speeds are five times faster, and the internet linking are and solid
Tablet Pc	Tablet Pc is the utmost common computer of our time which is a transportable personal computer classically smaller than a notepad
Notebooks	Notebook computers normally thinner design and weigh less than the laptop
Netbook	Is a device that may perform the most of the purposes of a desktop or laptop, It looks like tiny laptops, with screens infrequently beyond 10 or 12 inches
Laptop	The laptop is small and light sufficient to be used sitting in your lap
MP3 player	Is a digital audio player which plays music and audio files
iPod	A transportable media player that allows an operator to download materials such as: audio books, music, podcasts, and other video

The use of m-learning in education has several benefits for both teachers and students. Traditional classrooms are replaced by computers and mobile devices, so one can always have access to educational materials. A great advantage of m-learning is the individualization of the process of education of a very talented or dysfunctional student. It enables the education of students from less affluent or disabled families, for whom the access to school is a significant impediment. M-learning systems cannot replace traditional teaching methods, but they can be used as an additional tool in the educational process both in primary schools and universities [8],[10].

4. Analysis and evaluation of available electronic resources for m-learning

At present, there are many applications and services available on the market used in m-learning, which may cause some confusion when choosing a solution to achieve specific didactic goals. Based on research and analysis, an attempt was made to evaluate and classify the existing tools on the market, which allowed to identify more than 80 applications and services, whose functionalities and cases of use presented in the web and literature indicated their high usefulness for applications in the didactic process. The research focused mainly on the possibility of using applications and services in such aspects as developing didactic content and interactive way of presenting didactic materials, cooperation and sharing knowledge as well as developing the ability to search, select and comment on available resources and content [9].

5. Application of e-learning in primary and secondary schools

There are only a few primary and secondary schools in Poland that provide distance learning. Schools are implementing a project that focuses on full remote education. In practice, the student must work independently at home. The plan and the way of working for the whole week are written by the teachers. Tests and examinations at the end of the school year are held stationary in Poland in the school's headquarters or in examination centers in other countries, which are created after consultation with parents. The online school program is offered by private schools with therapeutic classes providing primary and secondary education. This project is aimed at children living in Poland, who have a certificate of psychological-educational counseling institutions because of autism disorders, as well as Polish children staying abroad with similar disorders, who want to pursue the Polish core curriculum. Therefore, this offer is dedicated mainly to children who, for health reasons, are unable to attend a stationary institution, also living abroad. Also parents who want to educate their children at the primary school level at home can benefit from the above mentioned educational project. Students have access to on-line lessons in the form of webinars, which are conducted by Polish teachers[7]. Remotely educating primary schools are usually bilingual in order to ensure a high level of the educational offer for Polish children living abroad. In the case of secondary schools, the distance education offer is much broader and focuses on general secondary schools. Secondary schools with modern methods of learning via the Internet mainly offer the preparation of adults to pass external exams. Many private schools provide e-learning services to people who cannot function in the school system, to people living outside the country and to people with disabilities. Secondary schools providing education via the Internet also provide opportunities for those who have decided to continue their interrupted education. They may also be used by people living in rural areas, where there are no appropriate schools or people involved in working and raising children. On-line schools provide students with access to teaching materials through various e-learning platforms, teaching aids in an interactive form make learning more enjoyable and faster [6].

6. Digital competence of teachers

Currently, teachers in various types of schools use new technologies to deliver lessons. Analyzing the survey data of general school teachers and technicians in total of 300 people has demonstrated that teachers most often use multimedia presentations, websites, and videos during lessons. Over 60% of the surveyed teachers use educational computer programs. Also e-learning platforms and electronic textbooks are the leading tools used by teachers to implement the core curriculum. Teachers are also increasingly beginning to use computer games in classroom activities, which until now have

been mainly used by students mainly outside school. Detailed information on the conducted surveys is shown in the figure below.

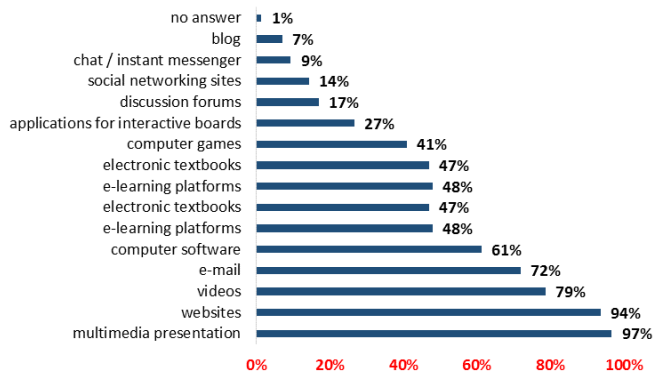


Fig. 2. Tools used in the education process [7]

7. New technologies in the process of education of IT schools students and the results of examinations confirming qualifications in the profession of IT technician

The legislation introducing changes in vocational education has entered into force in 2012. In the professions presented in the new classification, qualifications were distinguished. A qualification in the profession should be understood as a set of expected learning outcomes in a given profession, the achievement of which is confirmed by a certificate issued by the regional examination commission, after passing the exam confirming qualifications in the profession in the scope of one qualification. An examination confirming qualifications in a profession, also called a vocational examination, is a form of assessing the level of knowledge and skills within the scope of a given qualification established on the basis of curriculum of training in professions, achieved by a person passing the exam. The professional exam is an external exam and is assessed by external examiners[1]. In all vocational schools, teachers of vocational subjects prepare students for the exams of a given vocational qualification. In IT-oriented technical schools, three qualifications are tested:

- E.12. Installation and operation of personal computers and peripheral devices,
- E.13. Designing of local computer networks and network administration,
- E.14. Creating web applications and databases as well as database administration.

To prepare students for the above-mentioned professional qualifications teachers in class have been using new technologies to implement the core curriculum for many years. Traditional methods have been replaced by innovative methods of teaching lessons. The most frequently used educational methods and tools included online tests and examination tasks, multimedia presentations, instructional videos, google tools, educational blogs, and e-learning platforms.

The educational tools used by teachers allowed to obtain satisfactory exam results confirming qualifications in the profession of IT technician in three qualifications E.12, E.13, E14. The main reason for replacing traditional teaching methods with new IT tools was the rather poor results of in-school exams. Below are the results of the qualification exams for individual qualifications in 2017 and 2019 in the Lubelskie Voivodship and in the Technical College of Information Technology in Lublin[1].

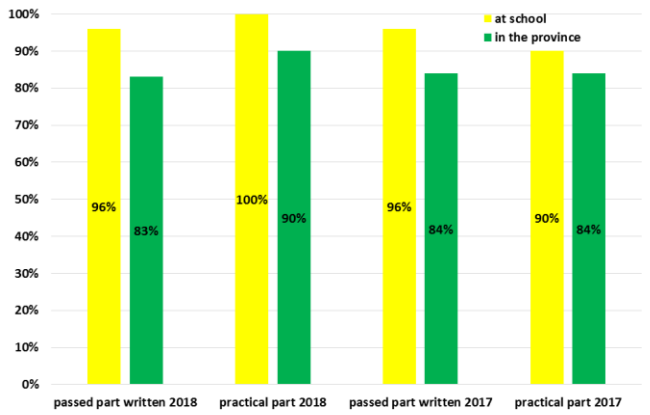


Fig. 3. Pass rate in the qualification 'Installation and operation of personal computers and peripheral devices in 2017 and 2018[1]

Figure 3 presents the percentage results of practical and written exams from E. 12 qualification. The graph compares the examination results from Lubelskie Voivodeship from the same qualification with the results of the Energy and Information Technology Technical School in Lublin. Analyzing the above results it is easy to notice that they are at a high level of pass rate in both the written and practical parts.

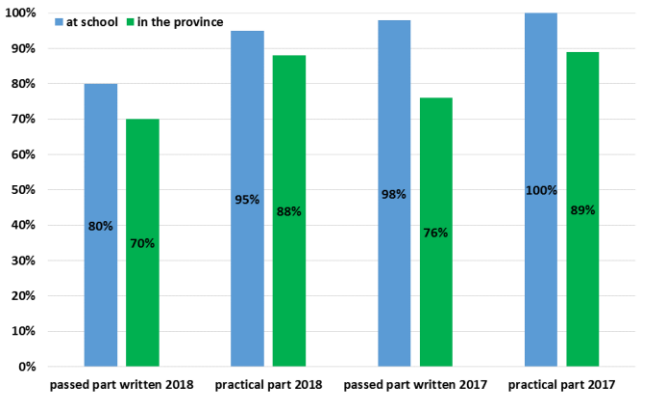


Fig. 4. Pass rate in the qualification 'Designing of local computer networks and administration of networks' in 2017 and 2018[1]

Figure 4 shows the percentage results of practical and written exams from E.13 qualifications. The graph compares the examination results from Lubelskie Voivodeship from the same qualification with the results of the Energy and Information Technology Technical School in Lublin. Analyzing the above results it is easy to notice that they are also at a fairly high level of pass rate in both the written and

practical parts. In this qualification, about 2017 was slightly better in comparison to 2018. This does not change the fact that the examination results obtained in the technical secondary school are at a higher level than in the whole Lublin voivodeship.

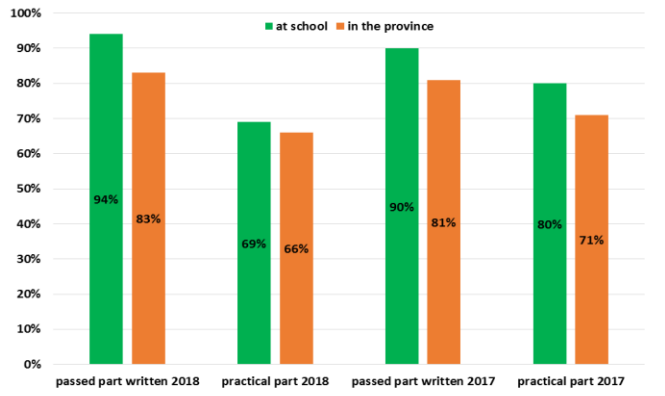


Fig. 5. Pass rate in the qualification 'Creating web applications and databases and administration of databases' in 2017 and 2018[1]

Figure 5 presents the percentage results of practical and written exams from E. 14 qualifications. The graph compares examination results from Lublin Province from the same qualification with the results of the Energy and Information Technology Technical School in Lublin. In this qualification, the written exam performed better than the practical exam in both 2017 and 2018[3].

Figure 6 presents the pass rate of exams in individual qualifications from both the practical and written part of the exams in a technical college. The presented results show that the written part is better than the practical part. Nevertheless, the results are at a rather high level. The annual qualification exams allow teachers to select appropriate methods and IT tools to work with students taking the exams in each qualification. The use of new technologies in the classroom allows mastering a wide range of materials in a relatively short period of time.

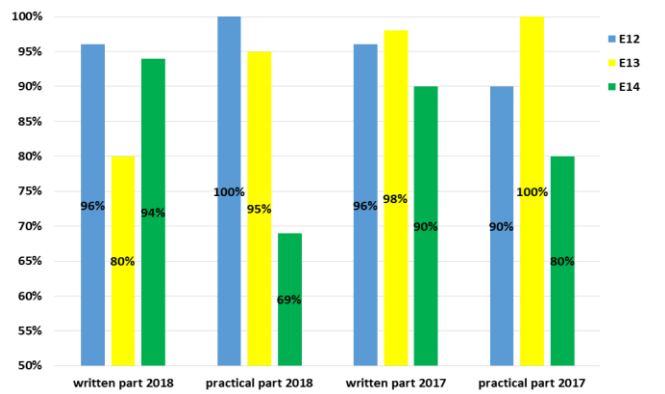


Fig. 6. School pass rate in 2017 and 2018[1]

8. Summary

The development of new media means that education is entering a new dimension. Knowledge began to be available to the public and transferred in an easily digestible form. Recent years have shown that content transmitted in the form of video lectures, online tests have become very popular among schoolchildren. Traditional methods of education are less effective and new technologies in the form of interactive educational platforms allow maximum facilitation of searching and assimilation of specialist knowledge from various fields of science. In-school examinations carried out by the author on various professional qualifications and their poor results were the main reason for the implementation of new technologies in education of young people. The new IT tools used by teachers, especially in IT classes, translated into good results of students taking qualification exams.

Sources

[1] https://cke.gov.pl/egzamin-zawodowy/egzamin-w-nowej-formule/wyniki-2/[23.11.2019]

[2] https://www.docebo.com/resource/report-elearning-trends-2019/[31.12.2019]

[3] https://www.edunews.pl/nowoczesna-edukacja/e-learning/246-e-learning-e-ksztalcenie-w-edukacji-szkolnej?showall=1&limitstart[22.11.2019]

[4] K. Mikołajczak, K. Pietraszek. Czy nauczyciele wykorzystują nowoczesne technologie informacyjno-komunikacyjne w kształceniu? Raport z badań [23.11.2019]

[5] K. Penny, P. Mileva, N. Tokmakov, D. Ruiz i M. Castro (2013). Metodyka prowadzenia szkoleń z wykorzystaniem urządzeń mobilnych dla e-biznesu v.2, http://pl.mtraining.eu/knowledge/methodology[12.11.2019]

[6] M. Kiliszewski. E-learning jako nowoczesny system zarządzania nauczaniem. Praktyka zarządzania nowoczesnym przedsiębiorstwem, pod red. M. Fertacha, S. Trzcielińskiego, Poznań 2003.

[7] M. Wrońska, Skuteczny nauczyciel czy skuteczne technologie informacyjne? Racjonalizm czy moda w edukacji?, „Gazeta IT", http://www.gazeta-it.pl, [27.07.2019]

[8] Modern Project Pedagogic assessment of most promising mobile and digital elearning resourcesfor european vocational training and higher education, http://www.modern.pm/resources/pedagogic-assessment [23.11.2019]

[9] P. Maczuga, M. Plewczyński, K. Sikorska, R. Fernandez, A. Jaruga, A. Świątecka. Modern Project (2016a). Audit of Digital and Interactive learning resources for European Vocational Training and Higher Education.

[10] P. Różewski, E. Kusztina i O. Zaikin (2008). Modele i metody zarzadzania procesem Otwartego nauczania zdalnego. Warszawa-Szczecin: Instytut Badań Systemowych PAN.

[11] Y. Sang., K. Chang i T. Liu (2016). The effects of integrating mobile devices with teaching and learning on students' learning performance: A meta-analysis and research synthesis. Computers & Education, 94,252–275.