Polska Akademia Nauk Oddział w Gdańsku
Komisja Informatyki

Metody Informatyki Stosowanej

Nr 5/2011 (30)

Szczecin 2011

Metody Informatyki Stosowanej
Kwartalnik Komisji Informatyki Polskiej Akademii Nauk Oddział w Gdańsku

Komitet Naukowy:

Przewodniczący:

prof. dr hab. inż. Henryk Krawczyk, czł. koresp. PAN, Politechnika Gdańska

Członkowie:

prof. dr hab. inż. Michał Białko, czł. rzecz. PAN, Politechnika Koszalińska
prof. dr hab. inż. Ludosław Drelichowski, Uniwersytet Technologiczno-Przyrodniczy w Bydgoszczy
prof. Constantin Gaindric, Corr. Memb. of Academy of Sciences of Moldova
prof. dr hab. inż. Janusz Kacprzyk, czł. rzecz. PAN, Instytut Badań Systemowych PAN
prof. dr hab. Jan Madey, Uniwersytet Warszawski
prof. Elisabeth Rakus-Andersson, Blekinge Institute of Technology, Karlskrona, Szwecja
prof. dr hab. inż. Leszek Rutkowski, czł. koresp. PAN, Politechnika Częstochowska
prof. dr hab. inż. Piotr Sienkiewicz, Akademia Obrony Narodowej
prof. dr inż. Jerzy Sołdek, Zachodniopomorska Szkoła Biznesu w Szczecinie
prof. dr hab. inż. Andrzej Straszak, Instytut Badań Systemowych PAN
prof. dr hab. Maciej M. Sysło, Uniwersytet Wrocławski

Recenzenci współpracujący z redakcją:

Marian Adamski, Andrzej Banachowicz, Zbigniew Banaszak, Joanna Banaś, Alexander Barkalov, Włodzimierz Bielecki, Piotr Błaszyński, Janusz Bobulski, Mariusz Borawski, Piotr Bubacz, Ryszard Budziński, Henryk Budzisz, Arkadiusz Bukowiec, Dariusz Burak, Tadeusz Czachórski, Jacek Czarnecki, Andrzej Czyżewski, Robert Dylewski, Piotr Dziurzański, Witold Dzwinel, Imed El Fray, Mykhaylo Fedorov, Paweł Forczmański, Dariusz Frejlichowski, Krzysztof Giaro, Zbigniew Gmyrek, Lesław Gniewek, Marek Gorgoń, Anna Grocholewska-Czuryło, Stanisław Grzegórski, Volodymyr Harbarchuk, Edward Hrynkiewicz, Zbigniew Huzar, Janusz Jabłoński, Andrzej Janczak, Jarosław Jankowski, Wojciech Jędruch, Szymon Kaczanowski, Tadeusz Kaczorek, Andrei Karatkevich, Andrzej Kasiński, Aleksander Katkow, Włodzimierz Khadzhynow, Przemysław Klęsk, Jacek Kluska, Leonid Kompanets, Józef Korbicz, Jerzy Korostil, Przemysław Korytkowski, Marcin Korzeń, Witold Kosiński, Magdalena Krakowiak, Krzysztof Kraska, Mariusz Kubanek, Georgy Kukharev, Eugeniusz Kuriata, Emma Kusztina, Piotr Lipiński, Małgorzata Łatuszyńska, Ewa Łukasik, Radosław Maciaszczyk, Wiesław Madej, Krzysztof Makles, Wojciech Maleika, Krzysztof Małecki, Radosław Mantiuk, Andrzej Marciniak, Oleg Maslennikow, Tomasz Mąka, Dariusz Mrozek, Piotr Mróz, Karol Myszkowski, Andrzej Napieralski, Evgeny Ochin, Krzysztof Okarma, Remigiusz Olejnik, Piotr Pechmann, Jerzy Pejaś, Andrzej Pieczyński, Andrzej Piegat Piotr Piela, Maria Pietruszka, Zbigniew Pietrzykowski, Mateusz Piwowarski, Marcin Pluciński, Anna Pławiak-Mowna, Orest Popov, Edward Półrolniczak, Artur Przelaskowski, Izabela Rejer, Valeriy Rogoza, Przemysław Różewski, Mirosław Ryba, Anna Samborska-Owczarek, Dominik Sankowski, Adam Słowik, Marek Stabrowski, Andrzej Stateczny, Jerzy Stefanowski, Janusz Stokłosa, Szymon Stoma, Jakub Swacha, Zdzisław Szczerbiński, Tomasz Szmuc, Marcin Szpyrka, Bożena Śmiałkowska, Ryszard Tadeusiewicz, Alexander Ţariov, Larysa Titarenko, Anna Tomaszewska, Leszek Trybus, Andrzej Tujaka, Zenon Ulman, Andrzej Walczak, Jarosław Wątróbski, Agnieszka Węgrzyn, Marek Węgrzyn, Sławomir Wiak, Waldemar Wolski, Waldemar Wójcik, Oleg Zaikin, Wojciech Zając, Danuta Zakrzewska, Zenon Zwierzewicz

Redakcja:

Antoni Wiliński (redaktor naczelny)
Dariusz Frejlichowski (zastępca redaktora naczelnego)
Piotr Czapiewski (sekretarz redakcji)

ISSN 1898-5297

Wydawnictwo:

Polska Akademia Nauk Oddział w Gdańsku, Komisja Informatyki
Adres kontaktowy: ul. Waryńskiego 17, 71-310 Szczecin, email: *redakcja.mis@gmail.com*
Wydano we współpracy ze Stowarzyszeniem Przyjaciół Wydziału Informatyki w Szczecinie

Spis treści

Yuri D. Grigoriev
PHENOMENON OF SOLUTIONS BRANCHING IN THE PROBLEMS OF THE OBJECTS PLACEMENT AND EXPERIMENTAL DESIGN 5

Arkadiusz Wrzosk
BADANIE EFEKTYWNOŚCI SYSTEMÓW REALIZOWANYCH W ARCHITEKTURZE SOA 21

Leszek Grocholski
SPECYFIKOWANIE WYMAGAŃ JAKO BARIERA WDRAŻANIA SYSTEMÓW KLASY B2B 35

Przemysław Wyszkowski
WYKORZYSTANIE SZYNY ESB DO EFEKTYWNEJ SYNCHRONIZACJI DANYCH POMIAROWYCH DUŻEJ OBJĘTOŚCI 49

Bogumiła Hnatkowska, Iwona Dubielewicz, Lech Tuzinkiewicz
WDRAŻANIE KURSU „ZESPOŁOWE PRZEDSIĘWZIĘCIE INŻYNIERSKIE" – DOŚWIADCZENIA, OCENA, ULEPSZANIE 57

Włodzimierz Dąbrowski, Agata Kosior, Andrzej Stasiak
STANOWISKA JĘZYKOWE DO BUDOWY SERWISÓW INFORMACYJNYCH. 67

Karol Domagała, Włodzimierz Dąbrowski
GRY SYMULACYJNE W ZARZĄDZANIU PROJEKTAMI INFORMATYCZNYMI. 79

Jan Trąbka
ZARZĄDZANIE PROJEKTEM WDROŻENIOWYM SYSTEMU KLASY ERP – AUTORSKA METODYKA. 91

Andrzej Grosser, Norbert Sczygiol
TWORZENIE JĘZYKÓW SPECYFIKACJI DLA ZAGADNIEŃ NUMERYCZNYCH 101

Marcin Deptuła, Julian Szymański, Henryk Krawczyk
INTERAKTYWNE WYSZUKIWANIE INFORMACJI W REPOZYTORIACH DANYCH TEKSTOWYCH . . 117

Marian Jureczko, Lech Madeyski
A REVIEW OF PROCESS METRICS IN DEFECT PREDICTION STUDIES 133

Sylwester Wosiak, Katarzyna Buda, Antoni Wiliński
POSZUKIWANIE TWARZY I ROZPOZNAWANIE EMOCJI W SCENACH ZMIENNYCH 147

Andrzej Ratkowski
WYKORZYSTANIE FORMALIZACJI JĘZYKA BPMN W BADANIU EWOLUCJI SYSTEMÓW SOA . . 159

Sławomir Chyłek
QEMU CPU TRACER – AN EXACT PROFILING TOOL. 167

Mariusz Dramski
SHORTEST PATH PROBLEM IN STATIC NAVIGATION SITUATIONS. 173

Larisa Globa, Rina Novogrudskay
ONTOLOGY BASED APPROACH TO ANTARCTIC KNOWLEDGE PORTAL DESIGNING 181

Leonid Kompanets, Piotr Milczarski
ON ASYMMETRY AND FUZZINESS OF COGNITIVE-MENTAL TRAITS OF PERSONALITY, POSSIBILITY OF THE MENTAL WORKER PRODUCTIVITY THEORY CREATION 189

Piotr Milczarski, Leonid Kompanets
ON HYPOTHESIS REALIZATION OF THE NEUTRAL-SIDED, RIGHT-SIDED, LEFT-SIDED PERSONALITY, AND CREATION NECESSITY OF THE PRECISE GENERATION FACE BASE, FOR MODERN BIOMETRICS 199

Zbigniew Pietrzykowski, Paweł Banaś, Anna Wójcik
COMPUTING WITH WORDS IN COMMUNICATION PROCESSES AT SEA 207

Anna Plichta, Szymon Szomiński
SECURITY MANAGEMENT IN DATABASE SERVERS 215

Andrzej Piegat
NONLINEAR MODEL OF A STRAIGHT LINE 221

Edward Półrolniczak, Michał Kramarczyk
AUTOMATYCZNE PRZETWARZANIE NAGRAŃ DŹWIĘKOWYCH DO CELÓW ANALIZY 233

Phenomenon of solutions branching in the problems of the objects placement and experimental design

Yuri D. Grigoriev

Department of Computer Science,
St.-Petersburg State Electrotechnical University

Abstract:

The examples of solutions branching are presented in the construction of D- and A-optimal placement of beacons in distance measurement problem of navigation and in the construction of Bayesian and maximin D-efficient designs of experiment..

Keywords:

Solutions branching, bifurcation point, beacon, minimal support, Bayesian and maximin D-efficient experimental designs.

1. Introduction

We understand a mathematical problem of solutions branching as a problem of solutions finding of nonlinear equation

$$F(y;\lambda)=0, \quad y=y(\lambda)\in R^1, \quad \lambda\in R^1. \tag{1.1}$$

where nonlinear operator F depending on parameter λ is applied to unknown function or vector $y = y(\lambda)$.

The basic question occurs whether the solution of equation (1.1) exists for a given value of λ or not. Then another problems arise, namely: how many solutions this equation has and how their number varies depending on λ. Phenomenon of branching presents a particular interest , when at λ, passing through critical value λ_0, named a point of branching or bifurcation, the solution of the equation (1.1) split into two or more solutions. From the mathematical point of view this is a classical problem on implicit functions, when the partial derivative or Jacobian $J = \partial F/\partial\lambda$ vanishes at the corresponding point.

The theory of branching of nonlinear equations ascenting to Newton is well studied presently both in theoretical and computational aspects, see, for example, Keller and Antman (1969), Vainberg and Trenogin (1974). The main problems solved within the framework of this theory are the following: determination of parameter values λ at which the number of solutions of the equation (1.1) changes (existence theory); determination of a number of solutions in neighborhood of bifurcation point λ_0 (multiplicity theory); behavior of solutions in neighborhood of λ_0 (spectral theory).

The aim of the present paper consists in the presentation of substantial examples of nonlinear equation solution brunching arising in the theory of object placement and in the optimal experimental design theory. The structure of the paper is the following. In Section 2 two elementary examples are considered illustrating the idea of solution branching. They are borrowed with insignificant supplements from Vainberg and Trenogin (1974) and Keller and Antman (1969). In Section 3 navigational problem of a vessel place determination with

the help of two beacons disposed on an ellipse is considered. It is shown, that the nonlinear equation (1.1) has bifurcation points at *D*- and *A*-optimal configurations of two beacons. In Section 4 some examples of solution branching are considered in the problems of Bayesian and maximin *D*-efficient experimental design. The numerical result of Bayesian experimental design for Mihaelis-Menten model obtained by Staroselsky (2008a) is defined more exactly. On the basis of results presented in Bräss and Dette (2007) an algorithm for calculation of the first bifurcation point of the maximin *D*-efficient designs in a case of an exponential model is discussed.

2. Elementary examples of the bifurcation points

Consider two examples in this section. The first one is the bifurcation point of algebraic curve of the third order (folium of Descartes). The second example is a boundary value problem for a thin rod with jammed ends.

2.1. Folium of Descartes

Show the phenomenon of solutions branching using the example of algebraic curve of the third order named by folium of Descartes

$$\lambda^3 + y^3 - 3\lambda y = 0\,. \tag{2.1}$$

Usual consideration of this curve from the view point of singular points leads to conclusion that the origin ($\lambda_0 = 0$) is a bifurcation point (see Fig. 1), in which the solution branching of the equations (2.1) occurs. From geometrical reasons we observe that there exist three such branches, and $y(\lambda) \to 0$ as $\lambda \to \lambda_0$.

Using a Newton's diagram method (see, for example, Vainberg and Trenogin (1974)), we obtain the following expansions of three branches of a Cartesian folium (2.1) in the origin:

$$y_1(\lambda) = \frac{1}{3}\lambda^2 + \frac{1}{3^4}\lambda^5 + \frac{1}{3^6}\lambda^8 + \frac{4}{3^9}\lambda^{11} + o(\lambda^{11}), \quad \lambda \to 0\,, \tag{2.2}$$

$$y_{2,3}(\lambda) = \pm\sqrt{3}\lambda^{1/2} - \frac{1}{6}\lambda^2 \mp \frac{\sqrt{3}}{72}\lambda^{7/2} - \frac{1}{162}\lambda^5 + o(\lambda^5), \quad \lambda \to 0, \tag{2.3}$$

In the given case we obtain answers to all the questions formulated above: the branching point is one, its order is equal to three, the behavior of solutions is presented by the equations (2.2), (2.3).

2.2. Thin rod with the jammed ends

In this section we follow Reiss (1969). Let us consider a thin rod with jammed ends lying in a plane *Oxz*. The movements of the rod ends along the axis *Ox* are given. The form of the rod is described by two functions $u(x)$ and $w(x)$. They are dimensionless movements along the axes *Ox* and *Oz*. These functions satisfy the following differential equations and boundary conditions:

$$w''(x) + \lambda w(x) = 0, \quad w(0) = w(1) = 1, \quad 0 \le x \le 1, \tag{2.4}$$

$$u'(x) + \frac{1}{2}[w'(x)]^2 = -\beta\lambda, \quad u(0) = -u(1) = c > 0, \quad 0 \le x \le 1, \tag{2.5}$$

The constant λ in (2.4) and (2.5) is proportional to axial strain in the rod. The constant $c > 0$ in (2.5) is proportional to the given end movement. A physical constant $\beta > 0$ is given.

The problem (2.4) – (2.5) is a nonlinear one, because of the presence of the term $(w')^2$ in equation (2.5). Let's set

$$\lambda_n = (n\pi)^2, \quad n = 1, 2,... \tag{2.6}$$

The solution of the problem (2.4) is

$$w(x) = \begin{cases} 0, & \text{if } \lambda \neq \lambda_n; \\ A_n w_n(x) = A_n \sin n\pi x, & \text{if } \lambda = \lambda_n. \end{cases} \tag{2.7}$$

Here A_n is a constant which is not defined yet. To receive $u(x)$ at $\lambda = \lambda_n$ according to (2.6) we substitute $w_n = A_n w_n(x)$ into (2.5). Next we integrate received expression by variable x using a condition $u(0) = c$. We obtain

$$u(x) = u_n(x) \equiv c - \beta\lambda_n \left(1 + \frac{A_n^2}{4\beta}\right) x - \frac{n\pi A_n^2}{8} \sin 2n\pi x. \tag{2.8}$$

On the other hand using a condition $u(1) = -c$, we derive

$$c = c_n \left(1 + \frac{A_n^2}{4\beta}\right). \tag{2.9}$$

The equation (2.9) expresses a relation between end contraction and amplitude and gives "a response" of the rod. On Fig. 2 the graphic image of equation (2.9) is shown. The real solutions of (2.9) for any n at given A_n can exist if and only if $c \geq c_n$. Solutions branch off from the not goggle forms $A_n = 0$ at the points c_n. Thus the solution of linear problem determines the solution bifurcation points of a nonlinear problem. For any c from interval $c_n \leq c \leq c_{n+1}$ there exist $2n + 1$ solutions. For $c < c_1$ the stability loss is impossible. From (2.9) we observe that $dc/dA_n = c_n A_n/2\beta$. Thus for fixed amplitude A the parabola on Fig. 2 branching off from c_n has a steeper inclination than parabola branching off from c_m in the case $m < n$. Therefore parabolas do not intersect.

3. Solution branching in navigational problems

Navigational problems of a vessel placement determination can be considered as the problems of the object placement studied first by Fermat. There exist different statements and methods of solutions to these problems (see Barabanov and Barabanova (2007), Barabanova (2008)). The distance measurement problem of navigation in elementary statement consists in solution of the system of n equations

$$| \beta - a_i | = D_i, \quad i = 1,..., n, \tag{3.1}$$

where $a_i \in R^m$ are the given support points or beacons, $\beta \in R^m$ is an unknown point, D_i are observed distances to the object. Let's consider the planar ($m = 2$) distance measurement problem in the case of measurement of two values $r_i = D_i^2$, i.e. two quadrates of distances from object to beacons. The measurements model has the form

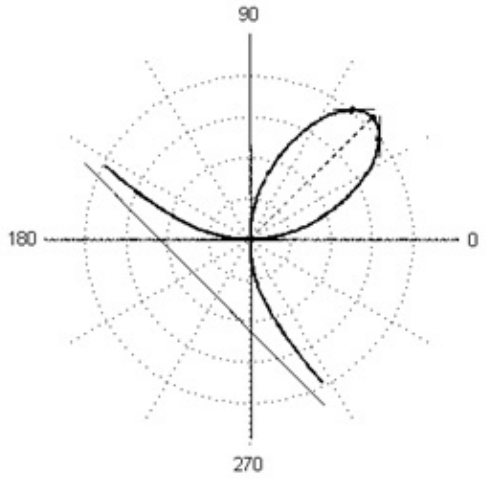


Figure 1. Folium of Descartes.

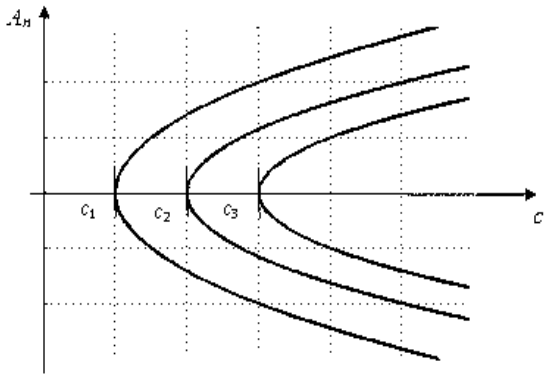


Figure 2. The solution branching: family of parabolas.

$$\mathrm{M}r_i = \eta(a_i, \beta) = (\beta_1 - x_i)^2 + (\beta_2 - y_i)^2, \quad a_i = (x_i, y_i), \quad i = 1, 2, \tag{3.2}$$

where $\eta(x, y; \beta)$ is a response function, $\mathrm{M}r_i = \sigma^2$ is a variance of measurements. We are interesting in such an placement

$$\xi = \{a_i = (x_i, y_i) \in R^2 : i = 1, 2\} \tag{3.3}$$

of beacons a_i at which it will be optimal in a certain statistical sense. More precisely our purpose is to show that there exists *D*- and *A*- optimal placements ξ^* with solution branching, i.e. they have bifurcation points.

3.1. Statement of the problem

In according to (3.2) and (3.3) we obtain

$$F(\xi; \beta) = \frac{\partial \eta(a, \beta)}{\partial \beta} = 2\begin{pmatrix} \beta_1 - x_1 & \beta_1 - y_1 \\ \beta_1 - x_2 & \beta_1 - y_2 \end{pmatrix}, \tag{3.4}$$

Thus, we define the Fisher's information $M(\xi, \beta) = F^T F$. Since any determinant is an oriented volume drawn on rows of the corresponding matrix, then $|M(\xi, \beta)|$ and $\mathrm{tr}M^{-1}(\xi, \beta)$ are functions of rows g_i of Jacobi matrix F and angle ω between them. Functionals

$$| M(\xi; \beta) | = \sigma^{-4} | F(\xi; \beta) |^2, \quad \mathrm{tr}M^{-1}(\xi; \beta) = \frac{| g_1 |^2 + | g_2 |^2}{\sigma^2 | M(\xi; \beta) |}, \tag{3.5}$$

correspond to criteria *D*- and *A*-optimality used in experimental design theory for construction of optimal experimental designs.

Suppose $\Omega \subset R^2$ is an area where beacons can be placed. Let Ω be an ellipse arc:

$$\Omega = \{(x, y) \in R^2 : x = a\cos\varphi, \, y = b\sin\varphi, \quad 0 \le \varphi \le 2\pi\}, \quad a \ge b > 0. \tag{3.6}$$

Denote by $\lambda = 1 - e^2$, where $e = \sqrt{1-(b/a)^2}$ is eccentricity of ellipse Ω. The first beacon we will place at the point $a_1 = (a; 0)$. It is required to determine placements ξ_D and ξ_A, for which

$$|M(\xi_D;\beta)| = \max, \quad \mathrm{tr}M^{-1}(\xi_A;\xi) = \min.$$

3.2. *D*-placements in a distance measurement problem

For placement $\xi \in \Omega$ defined by (3.5) we obtain

$$|M(\omega;\lambda)| = a^4(\cos^2\omega + \lambda\sin^2\omega)\sin^2\omega, \quad \lambda = e^2 \in [0,1], \tag{3.7}$$

and so

$$\frac{\partial|M(\omega;\lambda)|}{\partial\omega} = a^4\cos\omega\sin\omega(1-2\cos^2\omega-2\lambda\sin^2\omega). \tag{3.8}$$

The study of equation $\frac{\partial|M(\omega;\lambda)|}{\partial\omega} = 0$ shows that for $\lambda < \frac{1}{2}$ the values $\omega = 0, \frac{\pi}{2}, \pi$ give the minimum of functional $|M(\omega; \lambda)|$, and the third factor in expression (3.8) has two roots $\omega_1 = k_1\pi$, $\omega_2 = k_2\pi$ that are symmetrical to $\omega = \frac{\pi}{2}$. Precisely these roots corresponds to the maximum of $|M(\omega; \beta)|$ and here

$$k_1 + k_2 = 1, \quad |M(\omega_1;\lambda)| = |M(\omega_2;\lambda)|.$$

Really solving the equation

$$1 - 2\cos^2\omega - 2\lambda\sin^2\omega = 0, \tag{3.9}$$

for ω we find

$$\omega(\lambda) = \begin{cases} \omega_1 = \mathrm{arctg}\dfrac{1}{\sqrt{1-2\lambda}},\ \omega_2 = \pi - \omega_1, & \lambda < \dfrac{1}{2}, \\ \dfrac{\pi}{2}, & \lambda \ge \dfrac{1}{2}, \end{cases} \tag{3.10}$$

Thus in this case we get always two equivalent solutions.

If $\lambda \ge \frac{1}{2}$, value $\omega = \frac{\pi}{2}$ corresponds to an unique maximum of functional $\varphi(\omega;\lambda) = |M(\omega;\lambda)|$. So $\lambda = \frac{1}{2}$ is a branching point of multiplicity two of functions $\omega = \omega(\lambda)$, given implicitly by the branching equation

$$F(\omega;\lambda) := 1 - 2\cos^2\omega - 2\lambda\sin^2\omega = 0. \tag{3.11}$$

The given equation is solvable not only for ω, but also for $\lambda = \lambda(\omega)$, namely:

$$\lambda(\omega) = \frac{1}{2}(1 - \mathrm{ctg}^2\omega), \quad \frac{\pi}{4} < \omega < \frac{3\pi}{4}. \tag{3.12}$$

The function (3.12) is shown on Fig. 3. The function $\lambda(\omega)$ is a concave and unimodal one and reaches its maximum at the point $\omega = \frac{\pi}{2}$, thus $\lambda(\pi/2) = \frac{1}{2}$. Therefore if we take a concrete value $\lambda \in (0, \frac{1}{2})$, we easily obtain required values ω_1 and ω_2 both graphically and numerically.

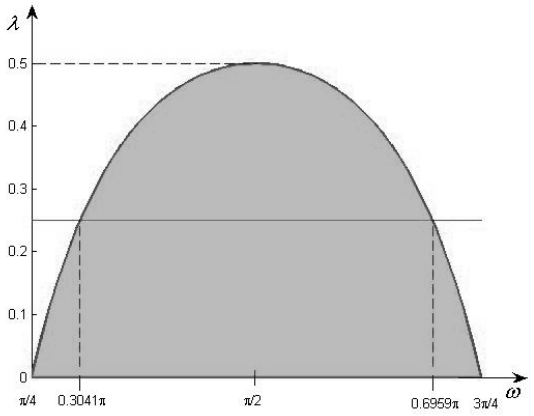


Figure 3. Criterion of *D*-optimality (3.12): function $\lambda(\omega)$.

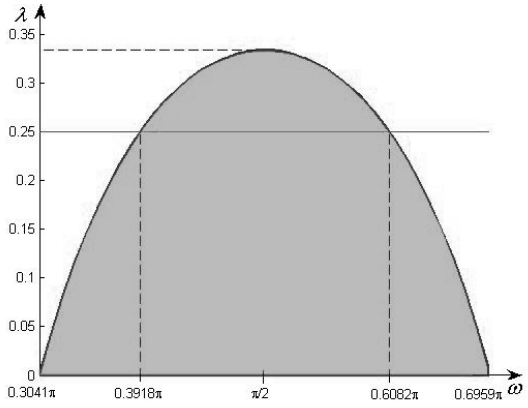


Figure 4. Criterion of *A*-optimality (3.16): function $\lambda(\omega)$.

3.3. *A*-placements in distance measurement problem

Let us consider now *A*-criterion. For the uniform plan ξ with two beacons we obtain

$$\mathrm{tr}M^{-1}(\omega;\lambda) = \frac{1+\cos^2\omega+\lambda\sin^2\omega}{a^2(\cos^2\omega+\lambda\sin^2\omega)\sin^2\omega}, \quad \lambda\in[0,1]. \tag{3.13}$$

Further we suppose $a = 1$.

Differentiating (3.13) by ω we arrive at implicit function $\omega = \omega(\lambda)$ given by numerator of the derivative $\frac{\partial \mathrm{tr}M^{-1}(\omega;\lambda)}{\partial\omega}$ in which non-vanishing factors are omitted:

$$F(\omega;\lambda) = \cos\omega(\cos^2\omega+\lambda\sin^2\omega-\sqrt{2}+1) = 0. \tag{3.14}$$

The analysis of equation (3.14) shows that for $\lambda < \lambda^* = \sqrt{2}-1$ the value $\omega = \frac{\pi}{2}$ corresponds to maximum of functional $\varphi(\omega;\lambda) = \mathrm{tr}M^{-1}(\omega;\lambda)$, and the second factor in (3.14) has two roots $\omega_1 = k_1\pi$ and $\omega_2 = k_2\pi$ that are symmetrical to $\omega = \frac{\pi}{2}$ and correspond to minimum $\mathrm{tr}M^{-1}(\omega;\beta)$. Moreover

$$k_1 + k_2 = 1, \quad \mathrm{tr}M^{-1}(\omega_1;\beta) = \mathrm{tr}M^{-1}(\omega_2;\beta).$$

Consequently as in the case of *D*-optimality, we have again two equivalent solutions.

If $\lambda \geq \lambda^*$ the value $\omega = \frac{\pi}{2}$ corresponds to unique maximum of functional $\varphi(\omega;\lambda) = \mathrm{tr}M^{-1}(\omega;\lambda)$. Thus $\lambda^* = \sqrt{2} - 1$ is a branching point of a function $\omega = \omega(\lambda)$ given implicitly by equation

$$F(\omega;\lambda) := \cos^2\omega + \lambda \sin^2\omega - \lambda^* = 0. \tag{3.15}$$

From here we derive

$$\lambda(\omega) = \lambda^* - (1-\lambda^*)\mathrm{ctg}^2\omega, \quad \omega_1 \leq \omega < \omega_2, \tag{3.16}$$

where

$$\omega_i = \begin{cases} \arccos(\sqrt{2}-1) = 0.8716, & i = 1, \\ \pi - \arccos(\sqrt{2}-1) = 2.2699, & i = 2. \end{cases} \tag{3.17}$$

The graph of function (3.16) is presented on Fig. 4. This function is concave and unimodal and reaches its maximum at the point $\omega = \frac{\pi}{2}$. So $\lambda(\pi/2) = \sqrt{2} - 1 = 0.4142$. Therefore choosing specific values $\lambda \in [0, \lambda^*]$ we easily obtain numerically desired values of k_1 and k_2.

3.4. Basic result

Uniting results on the beacons placement for *D*- and *A*-optimality criteria we conclude that the following theorem takes place.

Theorem 1. *Let the following conditions hold*:

(1) $\lambda^* \in (0,1)$ *is the branching point given by the expression*

$$\lambda^* = \begin{cases} \frac{1}{2}, & \textit{if } D-\textit{criterion is used}; \\ \sqrt{2}-1 = 0.4142, & \textit{if } A-\textit{criterion is used}; \end{cases}$$

(2) $\lambda = 1 - e^2$, $e = \sqrt{1-(b/a)^2}$ *is* eccentricity *of ellipse* (3.6).

Then

(1) *optimal mark angle* $\omega(\lambda)$ *is defined by expression*

$$\omega(\lambda) = \begin{cases} \omega_1 = arctg\sqrt{\frac{1-\lambda^*}{\lambda^* - \lambda}}, \ \omega_2 = \pi - \omega_1, & \lambda < \lambda^*, \\ \frac{\pi}{2}, & \lambda \geq \lambda^*. \end{cases} \tag{3.18}$$

(2) *inverse function* $\lambda(\omega) = \omega^{-1}(\lambda)$ *is defined by expression*

$$\lambda(\omega) = c - (1-c)\mathrm{ctg}^2(\omega), \quad \omega_- \leq \omega \leq \omega_+, \tag{3.19}$$

where

$$\omega_{\mp}(D) = \begin{cases} \pi/4, \\ 3\pi/4, \end{cases} \qquad \omega_{\mp}(A) = \begin{cases} \arccos(\sqrt{2}-1), \\ \pi - \arccos(\sqrt{2}-1). \end{cases} \tag{3.20}$$

Example 3.1. In Tables 1 and 2 *D*- and *A*-optimal mark angles $\omega(\lambda)$ are presented for ellipses with various values of $\lambda = 1 - e^2$.

Table 1. *D*-optimal placement designs for ellipse, $\omega_i = k_i\pi$, $\|a_1\| = 1$.

	Angles $\omega_i = k_i\pi$, $i = 1, 2$, [radian]		Angles ω_i, $i = 1, 2$, [degree]		Criterion
λ	k_1	k_2	ω_1	ω_2	$\lvert M(\omega^*;\lambda)\rvert$
0.01	0.2516	0.7884	45.2880°	134.7120°	0.2525
0.04	0.2566	0.7434	46.1880°	133.8120°	0.2604
0.09	0.2658	0.7342	47.8440°	132.1560°	0.2747
0.16	0.2805	0.7195	50.4900°	129.5100°	0.2976
0.25	0.3041	0.6959	54.7380°	125.2620°	0.3334
0.36	0.3451	0.6549	62.1180°	117.8820°	0.3906
0.49	0.4553	0.5447	81.9540°	98.0460°	0.4902
$\lambda = \frac{1}{2}$	0.5	0.5	90°	90°	0.5000
$\lambda > \frac{1}{2}$	0.5	0.5	90°	90°	λ

Table 2. *A*-optimal placement designs for ellipse, $\omega_i = k_i\pi$, $i = 1, 2$, $\|a_1\| = 1$, $\lambda^* = \sqrt{2} - 1$.

	Angles $\omega = k_i\pi$, $i = 1, 2$, [radian]		Angles $\omega = k_i\pi$, $i = 1, 2$, [degree]		Criterion
λ	k_1	k_2	ω_1	ω_2	$\mathrm{tr}M^{-1}(\omega^*;\beta)$
0.01	0.2794	0.7206	50.2842°	129.7158°	5.7701
0.04	0.2854	0.7146	51.3659°	128.6341°	5.5953
0.09	0.2964	0.7036	53.3525°	126.6475°	5.3039
0.16	0.3146	0.6854	56.6246°	123.3754°	4.8959
0.25	0.3450	0.6550	62.1004°	117.8996°	4.3713
$\lambda = \lambda^*$	0.5	0.5	90°	90°	3.4142
$\lambda > \lambda^*$	0.5	0.5	90°	90°	$1 + \lambda^{-1}$

4. Solution branching in experimental design

In the papers on experimental design a lot of authors in multiple empirical examples noticed that with growth of uncertainty of a priori information (in particular, with variance of a priori distribution or a priori interval of uncertainty growth) the number of support points for corresponding optimal design also increases. Strictly speaking the given effect differs from examples considered above. In our case the matter is that at bifurcation point not various new branches of solutions appear, but the structure of the solution changes on the whole. Thus the question is about peculiar branching in the space of probability measures but not in some functional space.

For the first time phenomenon of solution branching in the problem of Bayesian experimental design has been considered by Chaloner and Larntz (1989). Among the consequent

research, including maximin criteria, we could mention the papers of Dette and Neugebauer (1996), Melas and Staroselsky (2007), Staroselsky (2008a), (2008b), etc.

The theoretical explanation of the given phenomenon is presented in the paper of Bräss and Dette (2007). Beside it is explained there why maximin *D*-efficient designs contain greater number of points than Bayesian *D*-efficient designs.

The Bayesian and maximin criteria of optimality come from possibility of representations of a priori information on parameters of regression model. We consider the corresponding optimal designs for nonlinear regression models using specific examples borrowed with insignificant sharpenings from Bräss and Dette (2007) and Staroselsky (2008b).

4.1. Statement of the problem

Consider the nonlinear in parameter θ regression model

$$\mathrm{M}(y/x) = \eta(x,\theta), \quad x \in X, \quad \theta \in \Theta \subset R^m. \tag{4.1}$$

We are interested in the estimation of the unknown parameter θ by results of experiment. The quality of estimate $\hat{\theta}$ is characterized by value of some functional Φ of covariance matrix $D(\xi,\theta) = M^{-1}(\xi,\theta)$, where ξ is an experimental design, $M(\xi, \theta)$ is a Fisher's information matrix:

$$M(\theta,\varepsilon) = \int_X f(x,\theta) f^T(x,\theta)\xi(dx), \tag{4.2}$$

with

$$f(x,\theta) = \frac{\partial\eta(x,\theta)}{\partial\theta} = \left(\frac{\partial\eta(x,\theta)}{\partial\theta_1},\ldots,\frac{\partial\eta(x,\theta)}{\partial\theta_m}\right)^T, \quad \theta \in \Theta \subset R^m,$$

being the vector of basic regression functions that are linear independent and continuous on a compact space *X*. In the case of nonlinear parametrization of response function (4.1) matrix $M(\xi,\varepsilon)$, and, hence, functional $\Phi[M(\xi, \theta)]$, depend on true value of parameter $\theta_0 \in \Theta \subset R^m$.

Currently the great attention is given to Bayesian criterion of *D*-optimality

$$\Phi(\xi) = -\mathrm{M}_\pi \log|M(\xi,\theta)| = -\int_\Theta \log|M(\xi,\theta)|\,\pi(d\theta). \tag{4.3}$$

The design being optimal with respect to this criterion approximately maximizes an expected Shannon information supplied by experiment (see, for example, Chaloner and Larntz (1989)). The experimental design

$$\xi_B = -\arg\min_{\xi\in\Xi}\int_\Theta \log|M(\xi,\theta)|\,d\pi(\theta) \tag{4.4}$$

is called Bayesian *D*-optimal with respect to prior probability measure $\pi(\theta)$.

The most important tool for optimality testing of Bayesian designs is the theory of equivalence that generalizes the result of linear theory. In the case of criterion (4.3) it is possible to show (see Läuter (1974)) that the Bayesian design ξ_B is *D*-optimal if and only if the inequality

$$\int_\Theta d(x,\xi_B,\theta)\pi(d\theta) \le m \tag{4.5}$$

holds for any $x \in X$, where m is a number of unknown parameters $\theta \in \Theta \subset R^m$ and

$$d(x, \xi_B, \theta) = f^T(x, \theta) M^{-1}(\xi_B, \theta) f(x, \theta) \tag{4.6}$$

is a function of prediction variance. Therefore the algorithms for construction of Bayesian designs (4.4) do not differ from the algorithms used for construction of locally *D*-optimal designs.

Another kind of Bayesian approach to experimental design is Bayesian *D*-efficient approach. We show a phenomenon of solution branching on the examples using this approach. Introduce the quantity

$$\text{eff}(\xi, \theta) = \left(\frac{|M(\xi, \theta)|}{|M(\xi_0, \theta)|} \right)^{1/m}, \quad \theta \in \Theta \subset R^m. \tag{4.7}$$

The quantity (4.7) is said to be efficiency of the design ξ relative to locally *D*-optimal design ξ_0. Let $\pi(\theta)$ be a a priori distribution of θ and

$$Q(\xi, \theta) = \text{eff}^{-1}(\xi, \theta) = \left(\frac{|M(\xi_0, \theta)|}{|M(\xi, \theta)|} \right)^{1/m}, \quad \theta \in \Theta \subset R^m. \tag{4.8}$$

The experimental design

$$\xi_B = \arg \min_{\xi \in \Xi} \int_\Omega Q(\xi, \theta) \pi(d\theta), \tag{4.9}$$

minimizing the quantity $Q(\xi, \theta)$ is called Bayesian *D*-efficient design with respect to given probability measure $\pi(\theta)$.

Similarly if a a priori information on parameter θ is given only as an a a priori region Θ, it is possible to consider maximin *D*-efficient designs with the help of the same quantity $Q(\xi,\theta)$:

$$\xi_M = \arg \max_{\xi \in \Xi} \min_{\theta \in \Theta} Q(\xi, \theta). \tag{4.10}$$

As examples the ξ_B- and ξ_M-efficient designs branching in the following sections are considered for specific regression models.

4.2. Bayesian *D*-efficient designs

For testing of experimental design ξ on the Bayesian *D*-efficiency the special optimality conditions are used being generalization of a classical equivalence theorem from Kiefer and Wolfowitz (1960). We represent it according to Staroselsky (2008) in the following form. Let

$$\eta(x, \theta) = \sum_{i=1}^{k} \mu_i \omega_i(x, \lambda), \tag{4.11}$$

where $\mu = (\mu_1, ..., \mu_k) \in R^k$ are linear parameters, $\lambda = (\lambda_1, ..., \lambda_{m-k}) \in \Omega \subseteq R^{m-k}$ are nonlinear parameters, $\theta = (\mu, \lambda) \in \Theta \subset R^m$, $\omega_i(x, \lambda)$ are functions depending from nonlinear parameters λ only. Suppose also that $\Omega = supp\ \pi(\lambda)$ is a support of a a priori distribution of

parameters λ, ξ_0 is locally *D*-optimal design for $\lambda_0 \in \Omega$. The following assertion is valid: the Bayesian experimental design ξ is *D*-efficient relative to distribution $\pi(\lambda)$ if and only if

$$d(x,\xi,\Omega) = \int_\Omega (f^T(x,\lambda)M^{-1}(\xi,\lambda)f(x,\lambda) - m)Q(\xi,\lambda)d\pi(\lambda) \le 0. \quad (4.12)$$

The function $d(x, \xi, \Omega)$ is said to be extreme.

Let us consider a problem of construction of design ξ_B for rational model of Michaelis-Menten

$$\eta(t;\mu,\lambda) = \mu\omega(t;b) = \frac{\mu t}{\lambda + t}, \quad \mu,\lambda > 0, \quad 0 \le t \le T. \quad (4.13)$$

The design

$$\xi_0 = \left\{ t_1 = \frac{\lambda T}{2\lambda + T}, \quad t_2 = T \right\}. \quad (4.14)$$

is locally *D*-optimal for the Michaelis-Menten model. Therefore for the uniform design $\xi = \{t_1 = t; t_2 = T\}$ with the minimal support and free support point $t_1 = t$ one has

$$Q(\xi,\lambda) = \frac{T^2}{4}\frac{(\lambda+t)^2}{\lambda t(T+\lambda)(T-t)}, \quad 0 < a \le \lambda < b < \infty.$$

Using Melas parametrization we define the region

$$\Omega(z) = [a,b] = \{\lambda : 0 < \lambda_0(1-z) \le \lambda \le \lambda_0/(1-z)\}, \quad 0 \le z < 1,$$

of prior uniform probability measure

$$\pi_\lambda(x) := \mathrm{P}\{\lambda < x\} = \begin{cases} (x-a)/(b-a), & x \in \Omega, \\ 0, & x \notin \Omega. \end{cases} \quad (4.15)$$

Thus, $a = \lambda_0(1 - z)$, $b = \lambda_0/(1 - z)$ are the boundaries of Ω. The region $\Omega(0) = \{\lambda_0\}$ corresponds to degenerate distribution of π_λ and leads to locally optimal design. The region $\Omega(1) = (0,\infty)$ corresponds to a limiting as $z \to \infty$ singular distribution π_λ.

Let us denote

$$S = \sqrt{(b-a) - T\log\frac{b+T}{a+T}}, \quad U = \sqrt{(b-a) + T\log\frac{b}{a}}.$$

Theorem 2. *Let the following conditions hold*:

(1) *regression function* $\eta(x, \theta)$ *is given by* (4.14);

(2) $\Omega = [a, b]$ *is support of measure* (4.15);

(3) $t_2 = T$ *is a fixed point of design* $\xi_B = \{t_1, t_2 = T\}$ *with minimal support.*

Then

$$t_1(z) = \frac{T}{1 + U/S}. \quad (4.16)$$

is the point of Bayesian D-efficient design ξ_B *with minimal support.*

The proof of Theorem 2 can be received by immediate minimization with respect to *t* of the functional

$$\Phi(t) = \int_{\Omega} Q(\xi, \lambda)\pi(d\lambda)$$
$$= \frac{T}{4t(T-t)(b-a)}\left\{(b-a)T - (T-t)^2 \log\frac{b+T}{a+T} + t^2 \log\frac{b}{a}\right\}$$

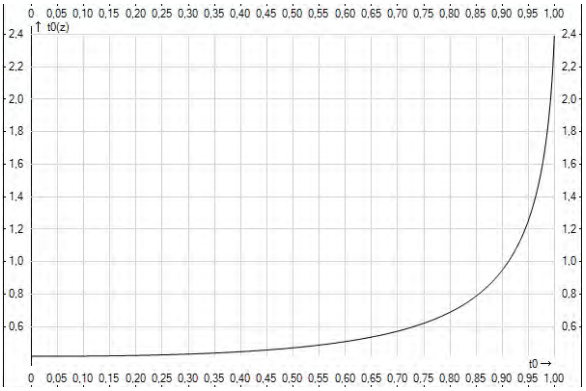


Figure 5. The point t_1 of Bayesian *D*-efficient designs with the minimal support as a function of *z*.

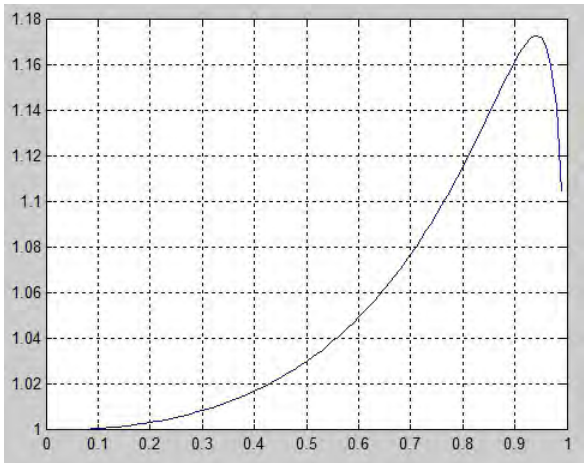


Figure 6. *D*-efficiency of Bayesian *D*-efficient designs with the minimal support as a function of *z*.

Since the condition $(\partial/\partial t)\Phi(t) = 0$ leads to quadratic equation with respect to t the required result follows directly. Notice also that as $\lim_{z\to 1} U/S = 1$, then $\lim_{z\to 1} t_1(z) = T/2$. The support point t_1 and a functional Φ as functions of $z \in [0, 1]$ are shown in Fig. 5 and Fig. 6.

Example 4.1. Let $T = 5$, $\lambda_0 = 0.5$. The points $t_1(z)$, the values of functional $\Phi(t_1)$ and variance $D\pi_\lambda$, as a functions of *z*, are shown in the Table 3.

Table 3. Bayesian *D*-efficient experimental design $\xi_B(z; 2) = \{t_1(z), t_2 = T\}$ with the minimal support, $\lambda_0 = 0.5$ is true value of λ, $T = 5$ is an upper boundary of λ, $t_1(0) = \lambda_0 T/(2\,\lambda_0 + T) = 0.4166$ is the support point of locally *D*-optimal design, $\Phi(z; 2)$ is the value of functional, $D\pi_\lambda = (b - a)^2/12$ is the variance of a prior probability measure π_λ.

z	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
$t_1(z)$	0.4179	0.4221	0.4307	0.4453	0.4691	0.5076	0.5718	0.6885	0.9487
$\Phi(z; 2)$	1.0008	1.0034	1.0085	1.0171	1.0302	1.0495	1.0771	1.1153	1.1617
$D\pi_\lambda$	0.0009	0.0042	0.0110	0.0237	0.0469	0.0919	0.1917	0.4800	2.0419

The specific behavior of functional $\Phi(\xi_B)$ which begins to decrease abruptly as $z \to 1$ indicates that as $z \to 1$ the point $t_1(z)$ of experimental design ξ_B with the minimal support should have the branching.

Example 4.2. *Michaelis-Menten model*. The calculation of bifurcation points z_k can be fulfilled with the help of necessary and sufficient condition of optimality (4.12). We obtain numerically the first point $z_1 = 0.835$. Thus, for the design $\xi_1 = \xi_B(z_1; 2)$ we have $\Phi(\xi_1) = 1.1313$. For $z_1 < z \le z_2$ the *D*-efficient design $\xi_B(z;N)$ should contain three points. In particular for $z = 0.9 \in (0.835, z_2]$ we obtain

$$\xi_B(0.9; 3) = \begin{pmatrix} t_1 = 0.162, & t_2 = 1.055, & t_3 = 5.000 \\ p_1 = 0.028, & p_2 = 0.472, & p_3 = 0.500 \end{pmatrix}, \quad \Phi(0.9; 3) = 1.150754.$$

For the design $\xi_B(0.9; 2)$ with the minimal support we have $\Phi(0.9; 2) = 1.1617$, i.e. the design $\xi_B(0.9; 3)$ with $\Phi(0.9; 3) = 1.150754$ is better.

Note that pointed in Staroselsky (2008b) a bifurcation point $z_1 = 0.67$ has not been defined precisely. Therefore the corresponding three-point design

$$\xi_B(0.8; 3) = \begin{pmatrix} t_1 = 0.18, & t_2 = 1.04, & t_3 = 5.000 \\ p_1 = 0.02, & p_2 = 0.48, & p_3 = 0.500 \end{pmatrix}, \quad \Phi(0.8; 3) = 1.1276,$$

is not optimal.

Really according to Table 3 we have $\Phi(0.8; 2) = 1.1153 < 1.1276 = \Phi(0.8; 3)$. On the other hand in the same paper the following empirical fact is marked: by the branching of the support points their common mass remains the same, i. e. all subsequent generations of support points of the experimental design keep mass of the points which have generated them. In the considered example this phenomenon proves to be true.

4.3. Maximin *D*-efficient designs

Let ξ be an experimental design given on the region $X \subset R^1$ and p_k are the masses of its support points x_k, $k = 1,\ldots, n$. Then the information matrix of design ξ is given by expression

$$M(\xi, \lambda) = \sum_{k=1}^{n} p_k M(x_k, \lambda),$$

where $M(x, \lambda)$ is an one-point information about λ at the point $x \in \{x_1,..., x_n\}$. The value

$$Q(\xi, \lambda) = \left(\frac{|M(\xi(\tilde{\lambda}), \lambda)|}{|M(\xi^*(\lambda), \lambda)|} \right)^{1/m}, \quad \lambda \in \Lambda \subset R^m, \tag{4.17}$$

exhibits the loss of information in the case if $\xi^*(\lambda)$ is locally *D*-optimal design for true value of parameter λ, while the experimenter uses the design $\xi := \xi(\tilde{\lambda})$ for supposed value $\tilde{\lambda}$. Thus $Q(\xi, \lambda)$:= eff(ξ) is *D*-efficiency of ξ similar to (4.7).

The design ξ_M is said to be maximin *D*-efficient if

$$\min_{\lambda \in \Lambda} Q(\xi_M, \lambda) = \max_{\xi \in \Xi} \min_{\lambda \in \Lambda} Q(\xi, \lambda). \tag{4.18}$$

It is shown by Bräss and Dette (2007) that for given criterion the number of support points of design ξ_M can be arbitrary large. The optimality of the *n*-point designs $\xi_M(n)$ can be tested with the help of equivalence theorem formulated by Wong (1992).

We will illustrate the idea of solution branching using the example of one-parameter exponential model

$$\eta(x, \lambda) = e^{-\lambda x}, \quad \lambda \in [1, \infty), \quad x \in [0, 1], \tag{4.19}$$

considered by Bräss and Dette (2007), where *n*-point designs $\xi_M(n)$ for $n = 2, 3, 4, 5, 6$ have been obtained but bifurcation points Λ_k, $k = 1, 2,\ldots$, in which the number of support points of maximin *D*-efficient design increases by one have not been specified. Our aim is to estimate the upper bound for the first bifurcation point Λ_1 using the design $\xi_M(2)$ of Bräss and Dette (2007) for $\lambda \in [0, 10]$.

Since the information about λ at the point x for model (4.19) is given by

$$M(x,\lambda)=\frac{\partial\eta(x,\lambda)}{\partial\lambda}=x^2e^{-2\lambda x}, \tag{4.20}$$

we obtain that the locally *D*-optimal design ξ^* is concentrated in a point $x^*=\lambda^{-1}$. Thus we have $M(x^*,\lambda)=(\lambda e)^{-2}$. Therefore according to (4.17) we obtain for the arbitrary one-point design $\xi=\{x\}$

$$Q(\xi,\lambda)=(\lambda x)^2e^{2(1-\lambda x)},\quad x\in[0,1]. \tag{4.21}$$

Let Λ_1, Λ_2 are the bifurcation points of design ξ_M for $\lambda\in[\Lambda_0,\Lambda]\subset[1,\Lambda_2]$. Set also

$$Q(\xi,\lambda_{k-1})=\min_{\lambda\in|\Lambda_{k-1},\Lambda_k|}Q(\xi,\lambda),\quad k=1,2.$$

Then

$$Q_{k-1}=\max_{\xi\in|\Xi|}Q(\xi,\lambda_{k-1})=Q(\xi_M(k),\lambda_{k-1}),\quad k=1,2,$$

is *D*-efficiency of design $\xi_M(k)$.

Theorem 3. *Let the following conditions are valid*:

(1) *regression function is given by* $\eta(x,\lambda)=e^{-\lambda x}$, $\lambda\in[\Lambda_0,\Lambda]\subset[1,\Lambda_2]$, $x\in[0,1]$;

(2) $\xi_M(1)=\{x_0\}$ *is one-point design*;

(3) Q_1 is *D-efficiency of two-point design* $\xi_M(2)$.

Then the following assertions are true:

(1) *if* $\lambda\in[\lambda_1,\lambda_2]\subset[\Lambda_0,\Lambda_1]$, *then one-point design* $\xi_M(1)=\{x_0\}$ *is given by*

$$x_0=\max_{0\le x\le1}Q(x,\lambda_0)=\frac{1}{\lambda_2-\lambda_1}\log\left(\frac{\lambda_2}{\lambda_1}\right),\quad \lambda_0\in\{\lambda_1,\lambda_2\}; \tag{4.22}$$

(2) *if* $\lambda\in[\Lambda_0,\Lambda]$, $Q_1=Q(\xi_M(2),\lambda_1)$, *then the best one-point design* $\xi_M(1)=\{x_0\}$ *is the point* $\{x_0\}$ *satisfying the equation*

$$x_0=\frac{1}{\lambda_1}\left(1+\log\frac{\lambda_1}{\sqrt{Q_1}}+\log x_0\right); \tag{4.23}$$

(3) *if* $\lambda\in[\Lambda_0,\Lambda]$, $Q_1:=Q(\xi_M(2),\lambda_1)$ *and* x_0 *is the root of equation* (4.23), *then the upper bound for the point of branching* Λ_1 *is the root of equation*

$$\Lambda_1=\frac{1}{x_0}\left(1+\log\frac{x_0}{\sqrt{Q_1}}+\log\Lambda_1\right). \tag{4.24}$$

The proof of the assertion (1) follows from the fact that for one-point design the efficiency $Q(x,\lambda)$ is an unimodal function of $x\in[0,1]$. The assertion (2) is obvious. The assertion (3) implies from following argument.

Let Λ be the upper bound for λ for which two-point design $\xi_M=\{(x_1,p_1),(x_2,p_2)\}$ is found, Q_1 is *D*-efficiency of this design. If $\Lambda_1<\Lambda$, then for values $\lambda\in[\Lambda_1,\Lambda]$ one- point design $\{x_0\}$ will be worse than two-point one, i.e. $Q_0(x_0,\lambda)=:Q_0<Q_1$. In the case $\Lambda<\Lambda_1$ the design $\{x_0\}$ will be better than any two-point design. Due to this reasoning the estima-

tion algorithm of branching point Λ_1 is built and, hence, the subsequent course of reasoning from which the assertion (3) implies.

Example 4.3. *Exponential model*. In the paper of Bräss and Dette (2007) the two-point design

$$\xi_M(2) = \begin{pmatrix} x_1 = 0.142, & x_2 = 0.771 \\ p_1 = 0.553, & p_2 = 0.447 \end{pmatrix}, \quad \lambda \in [1, 10],$$

have been obtained. Let $\lambda_1 = 1$, $\lambda_2 = 5$. According to (4.22) we obtain $\xi_M(1) = \{x_0 = 0.4023\}$, $Q_0(x_0, \lambda_0) = 0.5349$, where $\lambda_0 = 1$ or 5. Whether the one-point design $\xi_M(1) = \{x_0 = 0.4023\}$ is the best? In other words, whether the inequality $\Lambda_1 > 5$ is true or for $\lambda \in [1, 5]$ will be found better two-point design $\xi_M(2)$ than $\xi_M(1)$? In the last case this fact would mean that inequality $\Lambda_1 < 5$ takes place.

For the given design $\xi_M(2)$ we have $Q_1 = 0.4821$. From here according to (4.23) we obtain $x_0 = 0.2558$, $Q_0 = 0.2899$. The inequality $Q_0 < Q_1$ says that design $\xi_M(1)$ at the point $x_0 = 0.2558$ is less efficient than the design $\xi_M(2)$. Hence $\Lambda_1 < 10$. Solving the equation (4.24), we find $\Lambda_1 = 5.7154$. Thus one-point designs $\xi_M(1)$ are *D*-efficient at $\lambda \in [1, 5]$.

The value Λ_1 received by means of given algorithm is an estimate of true value Λ_1 only. Using this value as initial, we can define "exact" value of a branching point Λ_1 by straightforward calculation of efficiencies of one- and two-point designs $\xi_M(1)$ and ξ_M. The resultant value of the first bifurcation point $\Lambda_1 = 5.1355$. Thus, $\xi_M(1) = \{x_0 = 0.3956\}$ and

$$Q(\xi_M(1), 5.1355) = Q(\xi_M(2), 5.1355) = 0.5242.$$

However having no exact value of Λ_2 we do not obtain the true value Λ_1. The last value $\Lambda_1 = 5.1355$ again is only an estimate.

5. Conclusion

In the presented paper the short review of solution branching examples arising in the placement and experimental design theories have been given. A field of similar examples in the considered research regions can be essentially expanded. However computational difficulties of branching points search are not resolvable yet. Therefore it should be expected that any effort in development of the branching theory in placement theory and experimental design theory will be applied precisely in this direction.

References

[1] Barabanova L. P. (2008) *Optimization of configurations of four beacons in a difference range finding navigation system within the visibility cone* (Original Russian Text: Barabanova L. P. published in Izvestiya Akademii Nauk. Teoriya i Sistemy Upravleniya, 2008, No. 4, pp. 90–96.

[2] Barabanov O. O., Barabanova L. P. *Mathematical Problems of Distance Measuring Navigation*, Moscow, 2007. (Original Russian Text: Barabanov O. O., Barabanova, L. P. *Matematicheskie problemy dalnomernoj navigatsii*, Moskva, Fizmatlit, 2007, 272s.).

[3] Bräss D., Dette H. *On the number of support points of maximin and Bayesian D-optimal designs in nonlinear regression models*. Annals of Statistics, **35**(2), 772-792.

[4] Chaloner K., Larntz K. *Optimal Bayesian experimental design applied to logistic regression experiments*. J. Statist. Plann. Infer., 1989, **21**, 191-208.

[5] Dette H., Neugebauer H. M. *Bayesian optimal one point designs for one parameter nonlinear models*. J. Statist. Plann. Inference, 1996, **52**, 17-31.

[6] Keller J. B., Antman S. *Bifurcation Theory and Nonlinear Eigenvalue problems* / Edited by J.-B.- Keller and S.-Antman, W. A. Benjamin, Inc., New York, Amsterdam, 1969.

[7] Kiefer J., Wolfowitz J. *The equivalence of two extremum problems*. Can. J. Math., **12**, 129-132.

[8] Läuter Å. *Design of experiment method by non-linear parametrization*. Math. Operationsforsch. Statist., Ser. Statist., **5**, No. 7/8, 1974, 625-636 (in Russian).

[9] Melas V. B., Staroselsky Yu. M. *Studying maximin efficient designs for the Michaelis-Menten model*. Procced. St.-Petersburg University, 2007, Series 1, No. 2, 41-50. (Original Russian Text: Melas, V. B., Staroselsky Yu. M. *Issledovanie maximin-effektivnykh planov dlya modely Mikhaelisa-Menten*. Vestnik Sankt-Peterburgskogo universiteta, 2007, Seriya 1, Vipusk 2, 41-50).

[10] Reiss E. L. *Column buckling – an elementary example of bifurcation*.–In: Bifurcation theory and nonlinear eigenvalue problems / Edited by J. B. Keller and S. Antman, W. A. Benjamin, Inc., New York, 1969, Amsterdam.

[11] Staroselsky Yu. M. *Investigation of D-optimal designs for linear-fractional models.* Procced. S.-Petersburg University, 2008, Series 10, No. 3, 98-105. (Original Russian Text: Staroselsky Yu. M. Issledovanie bayesovskikh D-optimaljnikh planov dlya drobno-linejnikh modelej. Vestnik Sankt-Peterburgskogo universiteta, 2008, Seriya 10, Vypusk 3, 98-105).

[12] Staroselsky Yu. M. *Investigation of optimal experimental designs for nonlinear regression models*. Ph. D. Thesis. St.-Petersburg State University, 2008. (Original Russian Text: Staroselsky Yu. M. *Issledovanie optimalynikh planov eksperimenta dlya nelinejnykh po parametram rergessionnykh modelej*. St.-Petersburg State University.

[13] Vainberg M. M., Trenogin V. A. *Theory of Branching of Solutions of Non-linear Equations*. Monographs and Textbooks on Pure and Applied Mathematics, Noordhoff International Publishing, Leyden, 1974, xxvi+485 pp.

[14] Wong W. K. *A unified approach to the construction of minimax designs*. Biometrika, **79**, 611-619, 1992.

Badanie efektywności systemów realizowanych w architekturze SOA

Arkadiusz Wrzosk

Wojskowa Akademia Techniczna, Wydział Cybernetyki

Abstrakt:

Cechą charakterystyczną systemów Internetu Przyszłości i Internetu Usług jest ich heterogeniczność. Niejednolitość systemów ma duży wpływ na mechanizmy integracji, która może okazać się bardzo kosztowna. Architektura zorientowana na usługi (SOA) jest koncepcją wspierającą elastyczność, interoperacyjność i skalowalność systemów. Niestety zbiór zasad SOA ma również negatywny wpływ na wydajność systemów, a w konsekwencji na wspierane przez nie procesy. Usuwanie błędów na późnych etapach wytwarzania oprogramowania może być bardzo kosztowne. W pracy opisano zastosowanie narzędzi, metod i technik umożliwiających badanie wydajności systemów budowanych zgodnie z regułami SOA na etapie ich projektowania. Na podstawie modeli systemu, udokumentowanych z wykorzystaniem notacji UML, generowany jest model wydajnościowy oparty o Layered Queuing Network (LQN). Wyniki badania modelu mogą być wykorzystane w procesie doskonalenia architektury systemu.

Słowa kluczowe:

SOA, SOMA, LQN, UML, MARTE, wydajność, projektowanie

1. Wprowadzenie

Systemy Internetu Przyszłości i Internetu Usług, szczególnie te budowane na potrzeby inteligentnych miast, są systemami przetwarzającymi informacje w czasie rzeczywistym. Inną istotną cechą charakterystyczną takich systemów jest ich heterogeniczność. Jej konsekwencją są problemy związane z wymianą informacji między poszczególnymi systemami.

Koncepcją wspierającą interoperacyjność rozproszonych systemów jest architektura zorientowana na usługi (Service Oriented Architecture - SOA). Wdrożenie strategii SOA jest przedsięwzięciem długotrwałym i kosztownym, a systemy mogą być bardzo złożone. Modyfikacja późno wykrytych błędów w architekturze może być bardzo kosztowna. Dlatego ważne jest aby tak, jak w przypadku każdego systemu informatycznego, tak i systemy oparte o architekturę SOA spełniały istotne wymagania pozafunkcjonalne. Zagadnienie efektywności nabiera szczególnego znaczenia w sytuacji, gdy koncepcja SOA wykorzystana jest do budowy systemu, który wspiera procesy realizowane w ograniczeniach czasowych (np. procesy reagowania kryzysowego).

Ze względu na złożoność systemów opartych na paradygmacie SOA oraz koszt budowy takich rozwiązań, istnieje potrzeba badania ich efektywności na możliwie najwcześniejszych etapach wytwarzania. W pracy zaproponowano zastosowanie narzędzi, metod i technik umożliwiających badanie efektywności systemów opartych o architekturę SOA na etapie ich projektowania.

Podstawowym sposobem opisu architektury systemu jest przedstawianie jej za pomocą modeli, obrazujących zarówno aspekty strukturalne jak i behawioralne. Notacją wybraną do dokumentacji architektur opartych o koncepcję SOA jest Unified Modeling Language (UML).

Jest to język znormalizowany, stosowany do obrazowania, specyfikowania, tworzenia i dokumentowania artefaktów powstałych podczas procesu budowy systemu informatycznego. Jest on powszechnie akceptowany i stosowany w środowisku inżynierów. Dodatkowo UML dostarcza wygodnych mechanizmów, umożliwiających rozszerzenie jego semantyki, co jest niezbędne do opisu systemu z perspektywy specyfiki SOA oraz jego wydajności.

Na podstawie udokumentowanej architektury budowany jest model wydajnościowy systemu. Model ten może być wykorzystany do analizy efektywności różnych rozwiązań architektonicznych, a jej wyniki mogą być wykorzystane w procesie doskonalenia projektu systemu. Takie podejście pozwala na wczesne wykrycie błędów w architekturze oraz zmniejszenie kosztów wynikających z wprowadzania zmian na późnych etapach wytwarzania oprogramowania.

Struktura pracy jest następująca. Rozdział 2 opisuje prace związane z oceną wydajności oprogramowania na wczesnych etapach wytwarzania oprogramowania. Rozdział 3 opisuje podstawowe koncepcje takie jak: architektura zorientowana na usługi, model oprogramowania oraz model wydajnościowy. Rozdział 4 opisuje algorytm transformacji modelu oprogramowania na model wydajnościowy. W rozdziale 5 zaprezentowano przykład, natomiast w rozdziale 6 opisano wnioski.

2. Powiązane prace

W pracy [1] znajduje się opis metody umożliwiającej ocenę wymagań pozafunkcjonalnych monolitycznego systemu, którego architektura została udokumentowana z wykorzystaniem notacji UML. Na podstawie diagramów UML, adnotowanych z wykorzystaniem rozszerzeń zagregowanych w profilu „UML profile for Schedulability, Performance and Time" (SPT) [6], generowany jest model wydajnościowy oparty o warstwowe sieci kolejkowe (Layered Queuing Network - LQN), który następnie może być odczytany przez solwer LQN. Podobne podejście zostało opisane w [2]. Model LQN generowany jest na podstawie diagramów UML z adnotacjami z profilu MARTE [7].

Praca [5] prezentuje metodę generacji modeli LQN na podstawie modeli specyficznych dla architektury SOA. Bazą są modele UML zawierające opis procesów, komponentów realizujących usługi, scenariuszy, oraz rozmieszczenia oprogramowania na zasobach. Diagramy UML adnotowane są z wykorzystaniem profilu MARTE.

W [3] opisano narzędzie umożliwiające badanie wydajności i skalowalności architektur z wykorzystaniem symulacji dyskretno zdarzeniowej. Podobnie jak w innych metodach wykorzystywane są istniejące artefakty UML. Modele mogą być uruchomione równolegle, co pozwala na analizę alternatywnych architektur. Porównywane mogą być również wyniki symulacji pojedynczego modelu, sparametryzowanego w różny sposób (np. różne obciążenie).

Autorzy prac [8-11] koncentrują się na wydajności usług złożonych (wykorzystujących usługi wewnętrzne lub zewnętrzne dostarczone przez współpracujące organizacje) realizujących proces biznesowy. Pozwala to na porównanie różnych konfiguracji usługi realizującej proces i uzyskanie optymalnej z perspektywy wydajności.

3. Podstawowe koncepcje

Wydajność systemów uzależniona jest od tego, w jaki sposób system wykorzystuje zasoby. Różne implementacje tych samych funkcjonalności mogą charakteryzować się różną złożonością obliczeniową, czy pamięciową. W domenie wydajności można wyróżnić trzy podstawowe koncepcje: obciążenie systemu, zasoby i scenariusze. Scenariusze opisują zachowanie systemu z uwzględnieniem zapotrzebowania na zasoby. Każdy scenariusz uru-

chamiany jest z różną częstotliwością zdefiniowaną przez obciążenie systemu. Zasoby natomiast mogą reprezentować oprogramowanie (zasoby pasywne – np. bufor, sekcja krytyczna) lub platformę sprzętową (zasoby aktywne – np. procesor, dysk, sieć).

W celu wykonania badań wydajnościowych należy zastanowić się nad notacją, w której system będzie udokumentowany oraz nad modelem wydajnościowym. W kolejnych sekcjach opisano koncepcję architektury SOA, z uwzględnieniem bazowych elementów wymagających dokumentowania w celu przeprowadzenia badań wydajności. Opisano sposób dokumentowania architektury oraz model wydajnościowy wykorzystywany do badania wydajności architektur SOA.

3.1. Architektura zorientowana na usługi

Architektura zorientowana na usługi (Service Oriented Architecture – SOA) jest koncepcją architektury wspierającą interoperacyjność, skalowalność i elastyczność w heterogenicznym środowisku. Definiuje ona pojęcie usługi, która reprezentuje fragment funkcjonalności biznesowej. Wprowadza to dodatkową abstrakcyjną warstwę między logiką biznesową, a systemami działającymi w organizacji, zwiększając w ten sposób między nimi spójność. Przekształca ona styl architektoniczny aplikacji w zbiór powiązanych usług, które mogą być wykorzystywane wtedy, gdy istnieje taka potrzeba. Architekturę SOA można przedstawić jako architekturę warstwową. Rysunek 1 przedstawia wysokopoziomowy widok warstw [12].

Patrząc na architekturę warstwową od góry, najistotniejsza jest identyfikacja procesów i usług biznesowych wykorzystywanych przez użytkowników. Proces biznesowy opisuje w jaki sposób organizacja realizuje swoje cele. Warstwa procesów pozwala zatem zrozumieć domenę biznesową, która jest wspierana przez system w architekturze SOA. Ze względu na to, że procesy powinny być traktowane jako kompozycje innych procesów biznesowych i usług, powinny być dekomponowane na podrzędne podprocesy i kooperujące usługi.

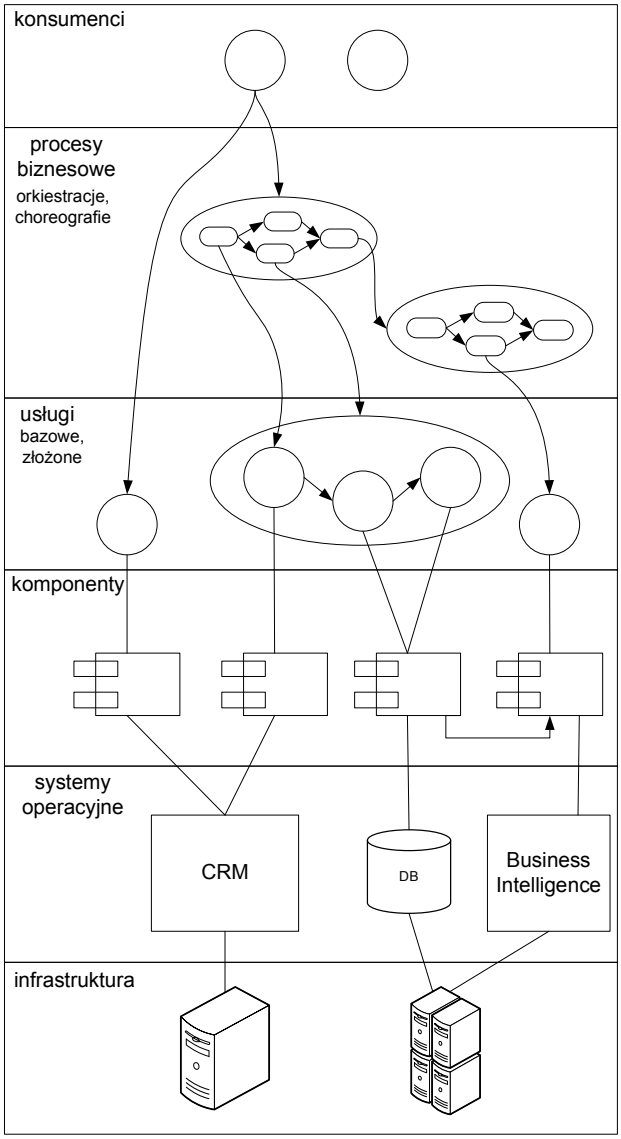
Usługi są opisane przez dobrze zdefiniowane interfejsy. Opis interfejsu jest niezależny od infrastruktury, czy systemu operacyjnego, na którym jest uruchomiona implementacja usługi. Pozwala to usługom, zbudowanym na heterogenicznych systemach, na interakcję w jednolity i uniwersalny sposób. Możemy wyróżnić dwie podstawowe kategorie usług:

- bazowe - dostarczają podstawowej funkcjonalności związanej z przetwarzaniem danych w nowych lub istniejących systemach. Z reguły tworzą one fundamentalną warstwę biznesową dla specyficznego systemu lub domeny problemowej. Przykładowymi usługami mogą być: zapis danych o kliencie itp.

- złożone - operują one na wyższym poziomie od usług bazowych. Usługi złożone maja z reguły dostęp do wielu systemów i są złożone z wielu usług bazowych (pełnią rolę kontrolera). Przykładową usługą biznesową może być usługa, która aktualizuje redundantne dane o kliencie w kilku systemach.

Funkcjonalność usług i procesów jest implementowana przez komponenty, które są odpowiedzialne również za ich jakość. Mogą one implementować nową logikę, jak również stanowić adapter do istniejących systemów działających w organizacji (np. klasy CRM)..

Wynikowy zbiór definicji procesów, usług, komponentów oraz istniejących systemów organizacji jest uruchomiony na infrastrukturze organizacji. Obejmuje ona zasoby sprzętowe, systemy operacyjne i wirtualne środowiska tworzące środowisko uruchomieniowe systemu.



Rysunek 1. Warstwy SOA

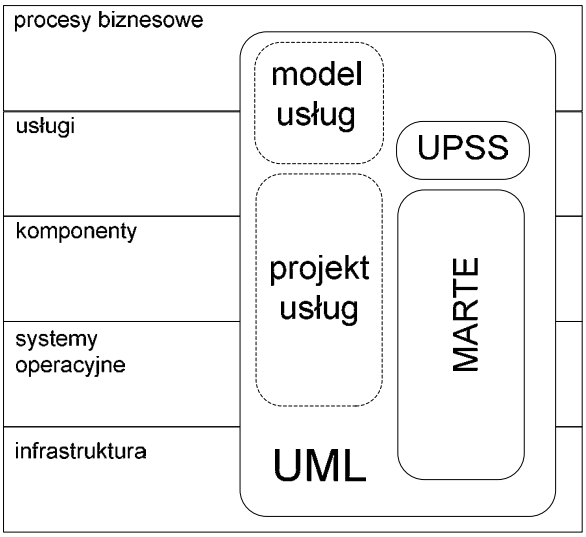
3.2. Model oprogramowania

Istotnym elementem badań wydajnościowych jest sposób dokumentacji architektury. Metodą, która definiuje cykl modelowania SOA, jest „RUP for Service-Oriented Modeling and Architecture" (SOMA). Opisuje ona trzy fazy projektowe: identyfikacja usług, specyfikacja usług oraz realizacja usług. Faza identyfikacji usług koncentruje się na wyszczególnieniu usług na podstawie modeli biznesowych. Faza specyfikacji usług skupia się na opisie usług z perspektywy tego, co usługa oferuje, czego wymaga i w jaki sposób jest udostępniona. Dodatkowo opisane są zależności od innych usług, kompozycje usług oraz wiadomości wymieniane przez usługę. Ostatnia faza koncentruje się na sposobie realizacji usług.

Z fazami specyfikacji i realizacji usług związane są odpowiednio modele: model usług i projekt usług. Według SOMA model usług jest abstrakcyjnym opisem usług zaimplementowanych w organizacji i wspierających implementację rozwiązań zorientowanych na usługi. Jest obszernym, złożonym produktem obejmującym wszystkie: usługi, dostawców, konsumentów, specyfikacje, wiadomości, kolaborację oraz relacje między usługami. Projekt

usług traktuje usługę jak „białą skrzynkę". Podstawowym elementem modelu jest komponent usługi. Według SOMA jest to artefakt wykorzystywany do opisu tego, jak specyfikacja usługi jest realizowana. Komponent, poprzez implementację wielu specyfikacji, może być realizacją więcej niż jednej usługi. Projekt usług jest produktem pośrednim pomiędzy modelem usług i implementacją.

Notacją wykorzystywaną do zapisu architektury jest „Unified Modeling Language" (UML) [13]. Jest on językiem znormalizowanym, służącym do dokumentowania projektu architektury. Może być stosowany do obrazowania, specyfikowania, tworzenia i dokumentowania artefaktów powstałych podczas procesu budowy systemu informatycznego. Jest powszechnie akceptowany i stosowany. Dodatkowo UML dostarcza wygodnych mechanizmów, umożliwiających rozszerzenie jego semantyki. Zbiór zdefiniowanych rozszerzeń jest agregowany w tzw. profilu. Rysunek 2 przedstawia notacje, profile oraz modele na tle warstw SOA.



Rysunek 2. Warstwy SOA

Do modelowania systemu wykorzystywano dwa profile UML: „UML 2.0 Profile for Software Services" (UPSS) [15] wspiera projektowanie rozwiązań w architekturze SOA natomiast profil „Modeling and Analysis of Real-Time and Embedded Systems" (MARTE) [14] wspomaga modelowanie systemów czasu rzeczywistego i systemów wbudowanych oraz jest zorientowany m.in. na analizę wydajności opartą na modelach.

3.2.1. UML 2.0 Profile for Software Services

Profil UPSS jest profilem wykorzystywanym na etapie identyfikacji i specyfikacji usług metody SOMA. Usługi dokumentowane są za pomocą modelu usług i traktowane są jak „czarne skrzynki". Poniżej omówiono wybrane stereotypy.

Punkt końcowy usługi modelowany jest z wykorzystaniem stereotypu *service*. Pozawala to oddzielić otoczenie usługi od sposobu jej realizacji. Jest używany do definiowania dwóch rodzajów interfejsów: wymaganych i dostarczanych.

Do specyfikacji usług wykorzystywany jest stereotyp *serviceSpecification*. Specyfikacja opisuje czego konsumenci mogą oczekiwać od usługi.

Innym istotnym stereotypem jest *serviceProvider*. Oznaczony nim komponent lub klasa reprezentuje element, który jest odpowiedzialny za realizację jednej lub kilku specyfikacji usług. Dostawca usług nie powinien eksponować swojej wewnętrznej struktury, ale powi-

nien wskazać publiczne porty, opisane stereotypem *service*, poprzez które specyfikacje usług są realizowane.

3.2.2. Modeling and Analysis of Real-Time and Embedded Systems

Profil MARTE umożliwia uzupełnienie projektu architektury o informacje z dziedziny wydajności. Pozwala na opisanie tego, w jaki sposób system wykorzystuje zasoby, posługując się trzema koncepcjami: obciążeniem systemu, zasobami i scenariuszami. Poniżej omówiono wybrane stereotypy.

Obciążenie systemu modelowane jest z wykorzystaniem strumieni zdarzeń inicjujących scenariusze *GaWorkloadEvent*. W strumieniu otwartym zdarzenia napływają do systemu z częstotliwością określoną pewnym wzorcem (np. periodycznie, z rozkładem Poissona). Strumień zamknięty definiuje strumień generowany przez określoną liczbę użytkowników, którzy pomiędzy kolejnymi wysyłanymi żądaniami, oczekują pewien czas poza systemem.

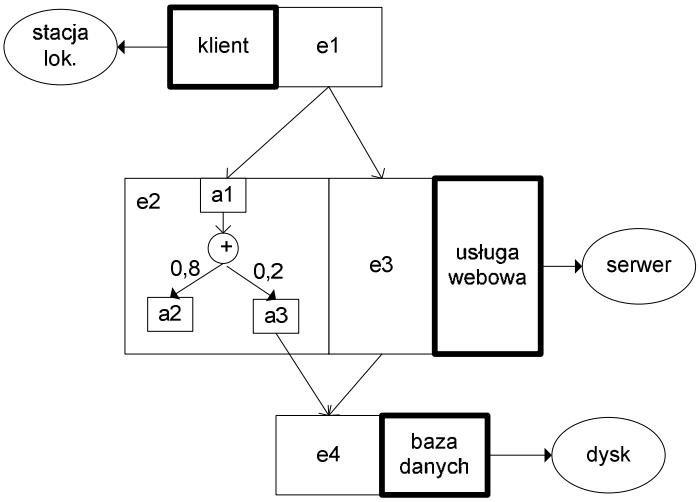
Zasoby modeluje się z wykorzystaniem klasy bazowej *Resource*, która pozwala opisać m.in. politykę szeregowania żądań, czy liczebność zasobu. Specjalizacje tej klasy pozwalają na modelowanie zasobów różnych typów. Zasoby aktywne wykorzystywane są do modelowania zasobów obliczeniowych (*GaExecHost*), czy elementów sieci (*GaCommHost*). Zasoby pasywne (*PaLogicalResource*) reprezentują zasoby współdzielone, które mogą być alokowane (np. pula połączeń do bazy danych).

Klasą opisująca scenariusze, powiązaną z diagramami definiującymi zachowanie systemu, jest *GaScenario*. Scenariusz jest sekwencją kroków (*GaStep*), które są w relacji poprzednik-następnik. Kroki reprezentują żądania dostępu do zasobów systemu, np. żądanie dostępu do procesora (*PaStep*), przesłanie komunikatu przez sieć (*PaCommStep*), czy akwizycja zasobów logicznych (*GaAcqStep*, *GaRelStep*).

3.3. Model wydajnościowy

Model wydajnościowy jest abstrakcyjną reprezentacją systemu zawierającą opis własności systemu związanych z dziedziną wydajności. Może być wykorzystany do oceny wydajności różnych projektów architektury, a wyniki analizy mogą być wykorzystane w procesie doskonalenia projektu systemu. Takie podejście pozwala wykryć błędy w projekcie architektury oraz uniknąć kosztów poprawy błędów wykrytych na późnych etapach wytwarzania oprogramowania. Istnieje wiele różnych i powszechnie znanych modeli wydajnościowych, np.: sieci kolejkowe, stochastyczne sieci Petriego czy procesy stochastyczne. W niniejszej pracy wykorzystano warstwowe sieci kolejkowe (Layered Queuing Network – LQN) [2].

LQN jest grafem, w którym można wyróżnić dwa typy wierzchołków. Pierwszy typ związany jest z oprogramowania. Oprogramowanie modelowane jest z wykorzystaniem „zadań". Każde zadanie posiada kolejkę wiadomości, w której oczekują przychodzące żądania oraz „wejścia", które reprezentują usługi udostępniane przez oprogramowanie. Czas obsługi żądania może być wyspecyfikowany z wykorzystaniem „faz", a bardziej złożone interakcje mogą być modelowane z wykorzystaniem „aktywności". Drugi typ węzłów reprezentuje zasoby obliczeniowe (procesory), które są wykorzystywane przez zadania podczas obsługi żądań. Zarówno węzły oprogramowania jak i sprzętu mogą działać w trybie jedno lub wielo-serwerowym. Możliwe jest dzięki temu modelowanie zadań wielowątkowych i maszyn wieloprocesorowych. Węzły powiązane są łukami, które reprezentują wysłane żądania (wysłanie wiadomości). Wyróżnia się trzy typy żądań: synchroniczne, asynchroniczne i przekazywanie [14]. Rysunek 3 przedstawia przykładowy graf LQN.



Rysunek 3. Przykładowy graf LQN

Na szczycie rysunku 5 znajduje się węzeł „klient" reprezentujący pewną klasę klientów. Każdy z klientów wysyła żądania do dwóch wejść: e2 i e3 zadania „usługa webowa" (prostokąt). Każde wejście ma określony czas obsługi żądania i wymagania w stosunku do innych zasobów. W tym przypadku, wejścia zadania „usługa webowa", wywołują wejście e4 zadania „baza danych". Każde zadanie reprezentujące oprogramowanie jest uruchomione na procesorze (kółko). Wejście e2 zostało dekomponowane na trzy aktywności tworzące dwie ścieżki alternatywne. Dla każdej aktywności określono czas obsługi żądania.

4. Mapowanie modelu oprogramowania na sieć LQN

Model oprogramowania SOA jest bazą do generacji modelu wydajnościowego. Graf LQN generowana jest na podstawie modelu usług i projektu usług.

Struktura sieci LQN generowana jest na bazie diagramów ujmujących statyczne aspekty systemu. Węzły sieci LQN reprezentujące oprogramowanie („zadania") generowane są dla: wykonywalnej wersji procesów, usług złożonych i bazowych udostępnionych konsumentom, komponentów odpowiedzialnych za realizację usług. Dodatkowe zadania generowane są dla elementów sieci komputerowej. Umożliwia to badanie czasu transmisji wiadomości przesyłanych z wykorzystaniem sieci. Węzły LQN reprezentujące zasoby obliczeniowe („procesory") generowane są na podstawie diagramów wdrożenia. Na ich podstawie można również wnioskować, jak komponenty oprogramowania rozmieszczone są na węzłach sieci. W przypadku usług bazowych dokumentowana jest ich szczegółowa realizacja. W celu zapewnienia śledzenia między warstwami usług i komponentów, zadania LQN generowane są dla obu elementów. Na szczególną uwagę zasługują usługi złożone i procesy. Ze względu na fakt, iż pełnią one rolę kontrolerów i klientów usług bazowych, węzły zadań LQN reprezentujące te usługi są bezpośrednio powiązane z procesorami.

Dla każdego zadania LQN, reprezentującego wykonywalną wersję procesu, generowane jest jedno wejście. Jest ono dekomponowane na aktywności, zgodnie z definicją procesu biznesowego. Wejścia zadań LQN generowane są na podstawie operacji, udokumentowanych w specyfikacji usług i komponentów. W zależności od złożoności realizowanej operacji, wejście może być dekomponowane na aktywności lub fazy. Na podstawie adnotacji z profilu MARTE generowane są czasy realizacji żądań.

Diagramy interakcji stanowią bazę do generacji łuków między węzłami sieci LQN (wywołania, żądania, przesyłanie wiadomości). Na ich podstawie tworzone są również węzły odpowiedzialne za generowanie strumieni zgłoszeń. Rysunek 4 zawiera algorytm transformacji.

1. Generacja struktury LQN
- **1.1. generuj zadania LQN dla: procesów, usług złożonych, usług bazowych, komponentów/klas odpowiedzialnych za realizację usług, elementów sieci komputerowej**
- **1.2. mapuj węzły z diagramu wdrożenia na procesory LQN**
- **1.3. generuj łuki między zadaniami i procesorami LQN**

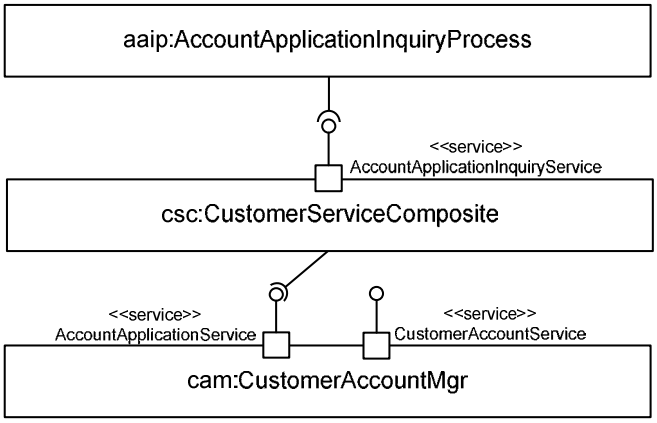
2. Generacja wejść, faz, aktywności
- **2.1. generuj wejścia, dla każdej operacji udostępnionej przez: proces, usługi, komponenty/klasy realizujące usługi oraz wiadomości przesyłanej przez elementy sieci komputerowej**
- **2.2. dla każdego wejścia {**
  - **2.2.1. generuj aktywności lub fazy**
  - **2.2.2. określ czasy wykonania faz i aktywności**
- **}**
- **2.3. dla każdego scenariusza {**
  - **2.3.1. generuj zadanie reprezentujące strumień zgłoszeń**
  - **2.3.2. generuj łuki (wywołania, żądania) dla wejść i aktywności**
- **}**

Rysunek 4. Algorytm transformacji

5. Przykład

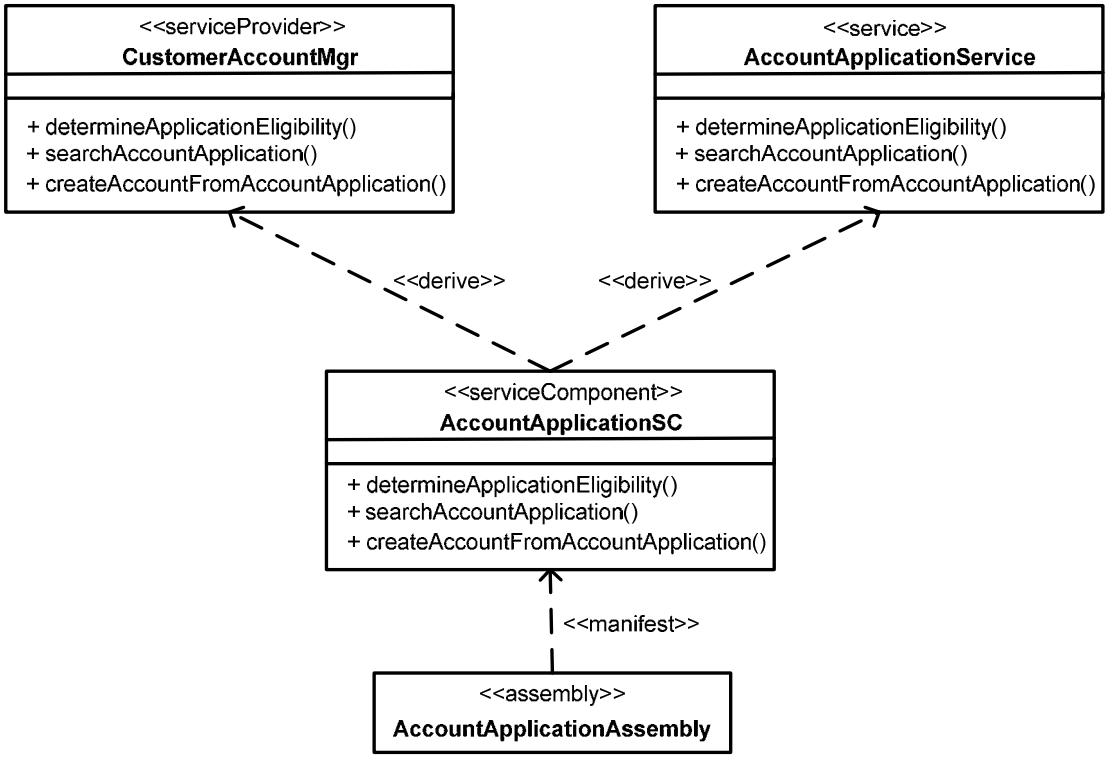
W bieżącym punkcie omówiono fragmentem przykładowego modelu systemu SOA, przedstawionego w [12]. System wspiera proces biznesowy realizowany w obszarze funkcjonalnym obsługi klienta. Architektura systemu udokumentowana została z wykorzystaniem notacji UML oraz profili: UPSS i MARTE. Ze względu na czytelność i prostotę przykładu nie zamieszczono pełnych adnotacji profilu MARTE oraz szczegółowego projektu usług.

W ramach systemu wyróżniono trzy komponenty. *AccountApplicationInquiryProcess* jest wykonywalną reprezentacją procesy biznesowego oraz konsumentem usług niższego poziomu. Do realizacji swoich celów wymaga usługi *AccountApplicationInquiryService*, której dostawcą tej jest *CustomerServiceComposition*. Do realizacji funkcjonalności tej usługi złożonej wymagana jest usługa bazowa *AccountApplicationService* dostarczona przez dostawcę *CustomerAccountMgr*. Rysunek 5 przedstawia omówione komponenty.



Rysunek 5. Diagram strukturalny systemu SOA

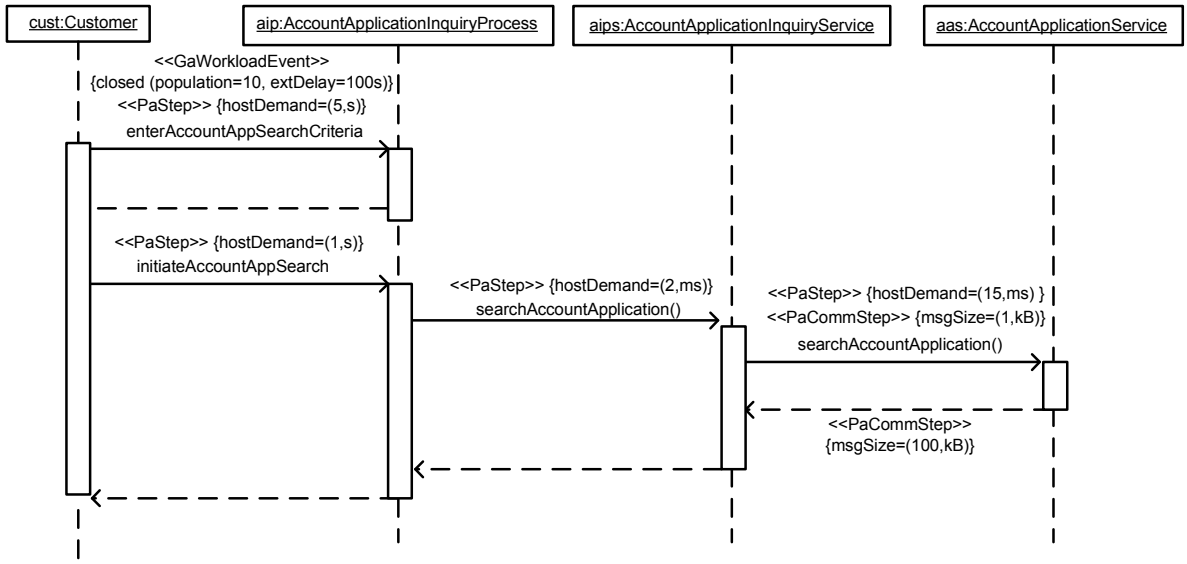
Z każdą usługą związana jest jej realizacja. Rysunek 6 przedstawia projekt usługi bazowej *AccountApplicationService*. Komponentem odpowiedzialnym za realizację funkcjonalności usługi oraz zapewnienie jej jakości jest *AccountApplicationSC*. Deskryptor instalacji *AccountApplicationAssembly* komponentu umożliwia śledzenie między diagramem opisującym komponent, a diagramem wdrożenia przedstawiającym rozmieszczenie komponentów na węzłach. Na potrzeby przykładu nie przedstawiono projektu komponentu, który może być uszczegółowiony z wykorzystaniem diagramów klas. Szczegółowy projekt komponentu również może być obiektem analizy wydajności.



Rysunek 6. Realizacja usługi bazowej *AccountApplicationService*

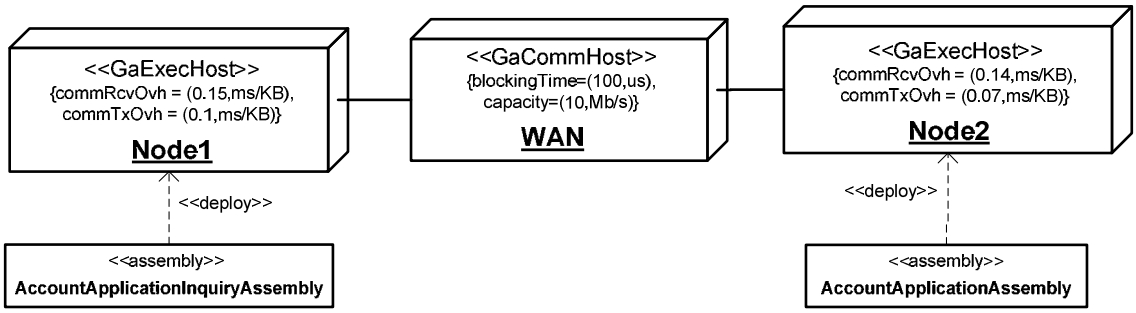
Zachowanie systemu zostało opisane z wykorzystaniem diagramu sekwencji. Rysunek 7 przedstawia interakcję usług realizujących fragment bazowego przepływu w procesie *AccountApplicationInquiryProcess*. Zdarzenia inicjujące proces generowane są zgodnie ze stru-

mieniem zamkniętym opisanym z wykorzystaniem stereotypu *GAWorkloadEvent*. Żądaniom dostępu do zasobów systemu nadany został stereotyp *PaStep*, natomiast wielkość komunikatów przesyłanych przez sieć została opisana za pomocą stereotypu *PaCommStep*. Zachowanie systemu może być modelowane na wyższym poziomie szczegółowości (np. interakcja klas odpowiedzialnych za implementację funkcjonalności dostarczanej przez komponent realizujący usługę bazową). Szczegółowy projekt interakcji klas odpowiedzialnych za realizację komponentu może być obiektem analizy wydajności.



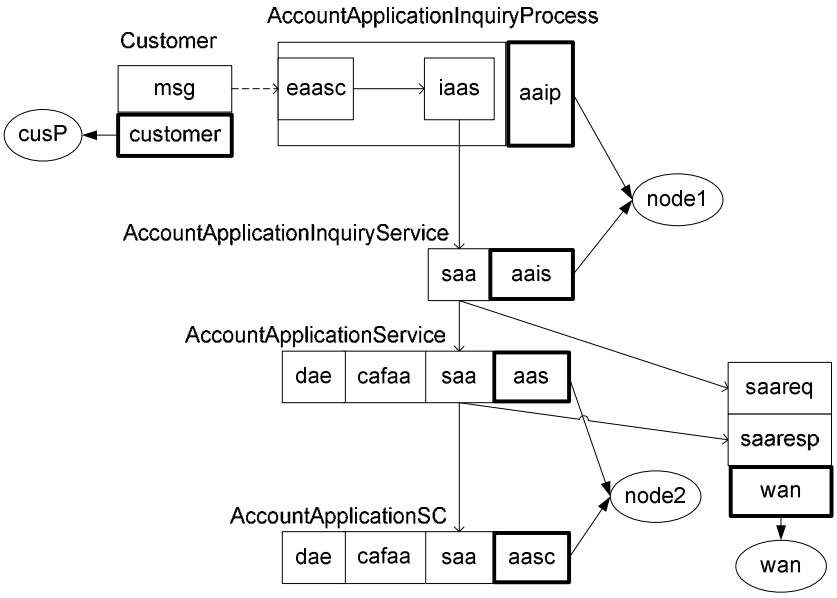
Rysunek 7. Fragment bazowego przepływ procesu zapytania o status aplikacji

Diagram wdrożenia odzwierciedla fizyczną strukturę całego systemu, z uwzględnieniem oprogramowania i sprzętu. Rysunek 8 przedstawia strukturę fizyczną przykładowego systemu oraz rozmieszczenie jego elementów na węzłach. Na diagramie widać zasoby aktywne wykorzystywane są do modelowania zasobów obliczeniowych (*GaExecHost*) oraz element sieci komputerowej (*GaCommHost*).



Rysunek 8. Diagram wdrożenia

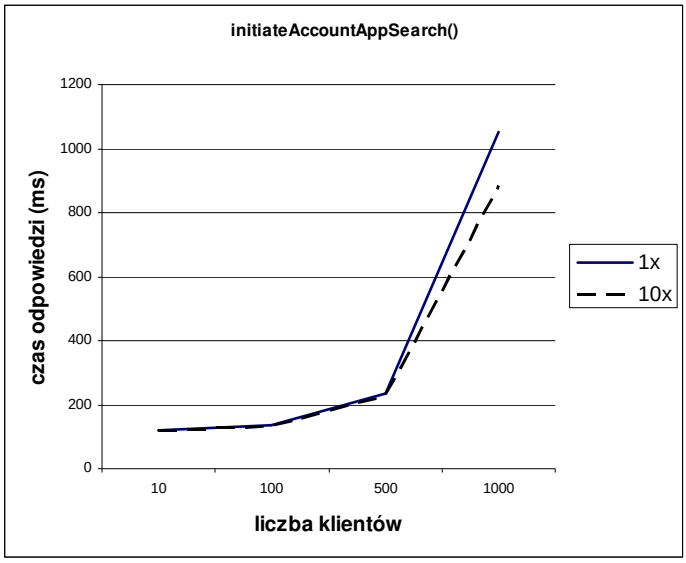
Model wydajnościowy generowany jest na podstawie diagramów opisanych powyżej. Diagramy transformowane są zgodnie z algorytmem zaprezentowanym na Rysunek 4. Rysunek 9 przedstawia sieć LQN dla omówionego przykładu. Przygotowany model może być rozwiązany z wykorzystaniem solwera LQN [16], a wyniki badania mogą być wykorzystane do doskonalenia projektu architektury systemu.



Rysunek 9. Przykładowa sieć LQN

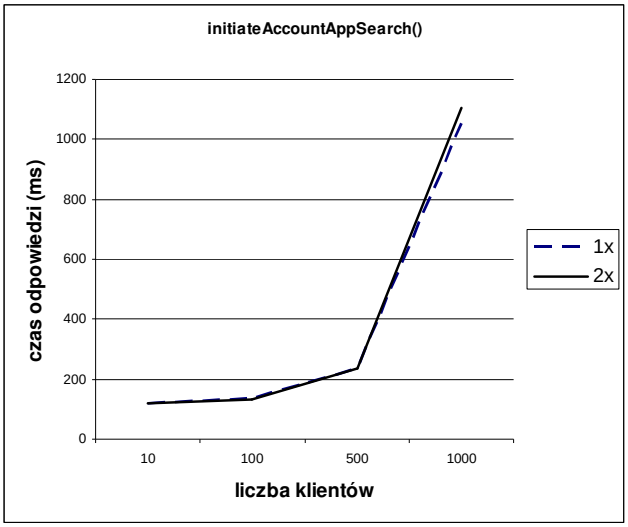
Wyniki przykładowych badań wygenerowanego grafu LQN zostały zaprezentowane poniżej. W celu przeprowadzenia analizy nie modyfikowano struktury i zachowania oprogramowania, a jedynie parametry przepustowości sieci i częstotliwości taktowania procesora.

Rysunek 10 przedstawia czas odpowiedzi operacji *initiateAccountSearch* w zależności od przepustowości sieci *WAN*. Na wykresie wyraźnie widać, iż dziesięciokrotne zwiększenie przepustowości sieci ma wpływ na czas odpowiedzi wraz z rosnącą liczbą klientów usługi. Może to stanowić przesłankę do zastosowania szybszego łącza.



Rysunek 10. Czas odpowiedzi w zależności od liczby klientów i przepustowości sieci WAN

Rysunek 11 przedstawia czas odpowiedzi operacji *initiateAccountSearch* w zależności od częstotliwości taktowania procesora w węźle *node1*. Na podstawie wykresu można wnioskować, iż dwukrotne zwiększenie częstotliwości taktowania nie wpływa znacząco na czas odpowiedzi badanej operacji. Na tej podstawie można podjąć decyzję o zastosowaniu słabszego procesora.



Rysunek 11. Czas odpowiedzi w zależności od liczby klientów i częstotliwości taktowania procesora

Przedstawiony przykład i wyniki badań wydajności potwierdzają, iż analiza wydajności na etapie projektowania może wspierać proces doskonalenia architektury systemu.

5. Podsumowanie

W pracy zaprezentowano zastosowanie narzędzi, metod i technik umożliwiająca ocenę wydajności architektur SOA. Przedstawiono algorytm, który na podstawie modelu usługi i projektu usług generuje model wydajnościowy oparty na LQN. Utworzony model może być wykorzystany do badania różnych projektów architektury, a wyniki badań mogą być zastosowane w procesie doskonalenia projektu systemu.

Jednym z podstawowych problemów, związanych z badaniami wydajności na wczesnych etapach wytwarzania systemu, jest duża liczba różnorodnych modeli z różnych poziomów abstrakcji. Poprawna parametryzacja modeli wymaga czasu i doświadczenia. Systemy realizowane zgodnie z koncepcją SOA są z reguły systemami bardzo dużymi, a więc analiza ich wydajności może być zadaniem złożonym. Kolejnym problemem są dodatkowe adnotacje nanoszone na diagramy. Twórcy oprogramowania, którzy korzystają z modeli podczas budowy systemów, nie są szkoleni w zakresie formalizmów wymaganych do przeprowadzenia badań. Adnotacje nanoszone na diagramy mogą mieć zatem wpływ na czytelność modeli wykorzystywanych w procesie wytwarzania systemu w architekturze SOA.

Bibliografia

[1] D. C. Petriu, H. Shen, *Applying the UML Performance Profile: Graph Grammar-based Derivation of LQN Models from UML Specifications*, TOOLS '02 Proceedings of the 12th International Conference on Computer Performance Evaluation, Modelling Techniques and Tools, Springer-Verlag London, UK, 2002, pp. 159-177

[2] J. Babau, M. Blay-Fornarino, J. Champeau, *Model Driven Engineering for distributed Real-Time Systems*, ISTE Ltd and John Wiley & Sons Inc., 2010

[3] P. C. Brebner, *Performance Modeling for Service Oriented Architecture*, ICSE Companion '08 Companion of the 30th international conference on Software engineering, ACM New York, NY, USA, 2008

[4] C.U. Smith. L. G. Williams, *Performance Engineering of Software Systems*, IEEE Transactions on Software Engineering, Volume 19 Issue 7, July 1993, pp.496
[5] M. Alhaj, D. C. Petriu, *Approach for generating performance models from UML models of SOA systems*, CASCON '10 Proceedings of the 2010 Conference of the Center for Advanced Studies on Collaborative Research IBM Corp. Riverton, NJ, USA, 2010, pp. 268-282
[6] Object Management Group, *UML profile for Schedulability, Performance and Time*, http://www.omg.org/technology/documents/profile_catalog.htm, [dostęp: 04.2012]
[7] Object Management Group, *UML Profile for Modeling and Analysis of Real-time and Embedded Systems (MARTE)*, http://www.omg.org/technology/documents/profile_ catalog.htm, [dostęp: 04.2012]
[8] A. D'Ambrogio, P. Boccirelli, *Model-driven Approach to Describe and Predict the Performance of Composite Services*, WOSP '07 Proceedings of the 6th international workshop on Software and performance, ACM, New York, USA, 2007, pp. 78 - 89
[9] A. Maduko, R. Jafri, J. A. Miller, *Modeling and Simulation of Quality of Service for Composition Web Services*, 7th World Multiconference on Systemics, Cybernetics and Informatics, 2003, pp. 420-425
[10] D. Rud, A. Schmietendorf, R. Dumke, *Performance Modeling of WS-BPEL-Based Web Service Compsition*, SCW '06 Proceedings of the IEEE Services Computing Workshops IEEE Computer Society Washington, DC, USA, 2006, pp. 140 - 147
[11] J. Grundy, J. Hosking, L. Li, N.Liu, *Performance Engineering of Service Composition*, SOSE '06 Proceedings of the 2006 international workshop on Service-oriented software engineering, ACM New York, NY, USA, 2006, pp. 26 – 32
[12] U. Wahli, L. Ackerman, A. Di Bari, *Building SOA Solutions Using the Rational SDP*, Redbooks, 2007
[13] Object Management Group, *Unified Modeling Language*, http://www.uml.org/, [dostęp: 04.2012]
[14] Object Management Group, *Business Process Model and Notation*, http://www.bpmn.org/, [dostęp: 04.2012]
[15] IBM, *UML 2.0 Profile for Software Services*, http://www.ibm.com/developerworks/ rational/library/05/419_soa/, [dostęp: 04.2012]
[16] Layered Queuing Network Solver, http://www.sce.carleton.ca/rads/lqns, [dostęp: 04.2012]
[17] N. M. Josuttis, *SOA in Practice: The Art of Distributed System Design (Theory in Practice),* O'Reilly Media, 2007
[18] Petriu D.C., Shousha C., Jalnapurkar A., *Architecture Based Performance Analysis Applied to a Telecommunication Systems*, IEEE Transactions on Software Engineering, Vol. 26 Issue 11, November 2000, IEEE Press Piscataway, NJ, USA, pp. 1049 – 1065

Specyfikowanie wymagań jako bariera wdrażania systemów klasy B2B

Leszek Grocholski

Uniwersytet Wrocławski, Instytut Informatyki

Abstract:

Evaluation of grant applications presents a unique opportunity for access to real projects documentation of real systems in real companies. The author has evaluated 45 projects for funding B2B IT systems. Analysis of errors in these proposals and given by Polish Agency for Enterprise Development negative data on the criteria evaluated in over 600 projects allowed to say that the applicant must have a basic knowledge of software engineering. Requirement specification is one of the main barriers to obtaining financing. The study identified the areas of requirements specification which concern the lack of basic knowledge. The results of this paper base on big number of real IT project documentation. It is unique because IT projects are the secrets of business. The obtained results can help companies applying for funding in the proper preparation of project documentation.

Keywords:

B2B, requirement specification, innovative economy

1. Wprowadzenie

Jako główna bariera rozwoju przedsiębiorstwa zawsze na pierwszy miejscu wymieniany jest brak środków finansowych. Rząd zauważa tą barierę i przeznacza w Programie Operacyjnym Innowacyjna Gospodarka (w skrócie POIG) znaczne dofinansowanie projektów informatycznych o podstawowym znaczeniu dla gospodarki. Jako jedną z 2 klas systemów wartych dofinansowania uznano systemy klasy B2B. Dotyczy ich działanie 8.2 POIG o nazwie „Wspieranie wdrażania elektronicznego biznesu typu B2B".

W ramach działania 8.2 Polska Agencja Rozwoju Przedsiębiorczości (w skrócie PARP) może udzielić wsparcia na [2] „wdrażania elektronicznego biznesu typu B2B z przeznaczeniem na realizację projektów o charakterze technicznym, informatycznym i organizacyjnym, które prowadzą do realizacji procesów biznesowych w formie elektronicznej". Projekt może dotyczyć integracji istniejących systemów informatycznych przedsiębiorstw lub wdrażanie nowych systemów, które umożliwią automatyzacje wymiany informacji pomiędzy systemami informatycznymi współpracujących przedsiębiorstw.

Kwota wsparcia projektu w działaniu 8.2 POIG nie może być niższa niż 20 000 złotych i jednocześnie nie może przekraczać 2 000 000 złotych. Intensywność wsparcia jest zróżnicowana w zależności od m.in. statusu przedsiębiorcy, lokalizacji projektu i kategorii wydatków kwalifikowalnych.

Liczba projektów zakwalifikowana do dofinansowania była mała i nie przekraczała 33% złożonych wniosków. W tabeli 1 przedstawiono dane na ten temat.

Tabela 1. Liczba pozytywnie ocenionych wniosków dot. dz. 8.2 POIG

	nabór	Liczba złożonych wniosków	Liczba pozytywnie ocenionych wniosków	% pozytywnie ocenionych wniosków
1.	I/2010	1002	245	24%
2.	I /2011	618	202	33%
3.	II/2011	763	216	28 %

W pracy zidentyfikowano i zanalizowano jedną z głównych barier otrzymania dofinansowania - brak podstawowej wiedzy z zakresu specyfikowania wymagań systemu informatycznego. Drugą główną barierą jest brak umiejętności sporządzenia dokumentacji projektowej zgodnie z wymaganiami formalnymi. Niestety PARP nie podaje jaka liczba wniosków nie spełnia wymagań formalnych.

1.1. Pojęcie systemu B2B

Cel działania 8.2 POIG podaje Szczegółowy opis priorytetów PO IG, 2007-2013 Narodowe Strategiczne Ramy Odniesienia na lata 2007-2013 [3].

„Celem działania 8.2 jest stymulowanie tworzenia wspólnych przedsięwzięć biznesowych prowadzonych w formie elektronicznej (B2B).”

Definicje relacji i systemu klasy B2B podaje Rozporządzenie Ministra Rozwoju Regionalnego z dnia 13 sierpnia 2008r. w sprawie udzielania przez PARP pomocy finansowej na wspieranie tworzenia i rozwoju gospodarki elektronicznej w ramach Programu Operacyjnego Innowacyjna Gospodarka, 2007—2013 [2].

„B2B obejmuje relacje usługowe oraz klasę systemów teleinformatycznych przeznaczonych do automatycznej komunikacji handlowej (wymiany danych) oraz koordynacji działań pomiędzy przedsiębiorstwami, stanowiące niezbędne ogniwo procesów biznesowych tych przedsiębiorstw, dotyczące różnego rodzaju współpracy pomiędzy takimi przedsiębiorstwami, w tym również w modelu wirtualnego przedsiębiorstwa, z zastosowaniem usług elektronicznych.”

Zgodnie z zapisami § 9 ust.2 pkt 4 powyższego rozporządzenia dofinansowanie w ramach działania 8.2 może uzyskać przedsiębiorca, który „planuje rozpoczęcie lub rozwój współpracy w oparciu o rozwiązanie elektroniczne, w szczególności przez dostosowanie własnych systemów informatycznych do systemów informatycznych przedsiębiorców, z którymi kooperuje, w celu umożliwienia automatyzacji wymiany informacji między systemami informatycznymi współpracujących przedsiębiorców”. Dlatego przedsiębiorcy współpracujący też zasadniczo powinni posiadać własne systemy informatyczne lub wdrożyć je równolegle z realizacją projektu przez wnioskodawcę. W wyniku realizacji projektu beneficjent posiadać będzie system informatyczny docelowo zintegrowany z systemami informatycznymi partnerów.

1.2. Metoda identyfikacji braku wiedzy

Poziom wiedzy z zakresu inżynierii oprogramowania w przypadku przedsiębiorstw badany jest z reguły przy pomocy ankiet. Wyniki pochodzą głównie ze Stanów Zjednoczonych. Autorowi nie są znane wyniki badań dotyczących specyfikacji wymagań systemów informatycznych przeprowadzonych na podstawie dokumentacji projektowej pochodzącej z dużej liczby przedsiębiorstw, w dodatku o ujednoliconej strukturze. Sporządzanie przez dużą liczbę przedsiębiorstw dokumentacji projektowej o tej samej strukturze możliwe jest

w zasadzie tylko w przypadku przetargów i konkursów. W tych przypadkach treść dokumentów jest z oczywistych powodów tajna.

Ocena wniosków o dofinansowanie złożonych w ramach działania 8.2 Programu Operacyjnego Innowacyjna Gospodarka stanowiła unikalną okazje analizy rzeczywistych projektów dotyczących zastosowania rzeczywistych systemów informatycznych w rzeczywistych przedsiębiorstwach. Autor ocenił 45 wniosków dotyczących dofinansowania systemów klasy B2B. Analiza błędów popełnianych w tych wnioskach oraz podane przez PARP dane [4], [5] na temat negatywnie ocenionych kryteriów ponad 600 wniosków złożonych w naborach I i II 2011r. umożliwiły stwierdzenie, że wnioskodawca musi posiadać podstawową wiedze dot. specyfikowania wymagań.

1.3. Dokumentacja projektowa

Dokumentacja projektu ubiegającego się o dofinansowanie składa się z:

- wniosku,
- biznes planu,
- załączników, m in.:
  – kopii dokumentów rejestracyjnych spółki,
  – kopi dokumentów potwierdzających współprace firm.

Informacje opisujące zawartość dokumentacji projektu ubiegającego się o dofinansowanie i warunki uzyskania dofinansowania zawarte są w następujących dokumentach:

- Instrukcja wypełniana wniosku o dofinansowanie realizacji projektu w ramach Programu Operacyjnego Innowacyjna Gospodarka - Działanie 8.2 Wspieranie wdrażania elektronicznego biznesu typu B2B [1]. Objętość: 28 stron.
- Przewodnik po kryteriach wyboru finansowanych operacji w ramach Programu Operacyjnego Innowacyjna Gospodarka, 2007-2013[1]. Objętość: 16 stron.
- Wytyczne Ministra Rozwoju Regionalnego w zakresie kwalifikowania wydatków w Ramach Programu Operacyjnego Innowacyjna Gospodarka[1]. Objętość: 79 stron.
- Rozporządzenie Ministra Rozwoju Regionalnego z dnia 13 sierpnia 2008 r. w sprawie udzielania przez Polską Agencję Rozwoju Przedsiębiorczości pomocy finansowej na wspieranie tworzenia i rozwoju gospodarki elektronicznej w ramach Programu Operacyjnego Innowacyjna Gospodarka, 2007-2013, wraz z późniejszymi zmianami [2]. Objętość: 46 stron.

Mimo ze podstawowe dokumenty: Rozporządzenie Ministra z 13 sierpnia 2008r., Przewodnik, Instrukcja wypełniania wniosku napisane są jasno i stosunkowo prostym językiem to jednak bardzo dużo wniosków zawiera błędy formalne i merytoryczne, powodujące niezakwalifikowanie wniosku do dofinansowania.

Zakres wymaganej dokumentacji obejmuje przede wszystkim opis wymagań dotyczących systemu B2B i uzasadnienie efektywności ekonomicznej projektu. Dokumenty projektowe zawierają wymagania dotyczące systemu B2B to:

- Wniosek o dofinansowanie, który składa się on z 22 punktów z których większość zawiera wiele podpunktów. Instrukcja wypełniania wniosku liczy 28 stron.
- Biznes plan składający się z 19 punktów z których z których większość zawiera podpunkty. Dla każdego punktu i podpunktu biznes plan dokładnie określa ich zawartość.

Szczegółowa lista i opis wymagań dotyczących systemu B2B, które w działaniu 8.2 POIG musi przedstawić wnioskodawca, zajęłaby kilkadziesiąt stron. Do opisu wymagań nie jest wymagana żadna formalna notacja. Agregacja wymagań dotyczących systemu B2B i potraktowanie tych wymagań jako specyficznego modelu byłyby bardzo obszerne i mogą

stanowić temat odrębnych badań. Ponieważ zakres opisu każdego wymagania jest szczegółowo określony a wiele opisów wymagań ma postać tabel to opis ten można nazwać specyfikacją wymagań. Zakres wymaganej specyfikacji wymagań jest w dużym obszarze zgodny z normą ANSI/IEEE Std 830-1993„Recommended Practice for Software Requirements Specifications". Ocena czy wyspecyfikowane wymagania spełniają wymagane kryteria akceptacji przeprowadzana jest na podstawie, udostępnionego również wnioskodawcy, podstawowego dokumentu o nazwie: Przewodnik po kryteriach wyboru finansowanych operacji w ramach Programu Operacyjnego Innowacyjna Gospodarka, 2007-2013. Dokument ten w dalszej części nazywany jest w skrócie „Przewodnikiem po kryteriach wyboru". Praca dotyczy dokumentu w wersji obowiązującej do II naboru w 2011r. [7].

2. Identyfikacja braku wiedzy z zakresu specyfikowania wymagań

Autor ocenił w latach 2010 – 2011 45 wniosków o dofinansowanie dotyczących działania 8.2 POIG. Dla I naboru z roku 2011r. w które złożono ponad 600 wniosków PARP podał [5] dane dotyczące procentowego udziału wniosków, dla których dane obligatoryjne kryterium merytoryczne uznano za niespełnione.

Nazwa działania 8.2 POIG to „Wspieranie wdrażania elektronicznego biznesu typu B2B". Poniżej uzasadniono, że brak wiedzy nt specyfikowania wymagań systemu B2B można uważać jak główna barierę „wdrażania elektronicznego biznesu typu B2B".

W pracy na podstawie własnych badań nt projektów dotyczących dofinansowania w ramach dz. 8.2 POIG zidentyfikowano obszary specyfikowania wymagań, których dotyczy brak podstawowej wiedzy. Dane podane przez PARP umożliwiły określenie jak często występują zidentyfikowane braki.

Dla każdego z 9 obligatoryjnych, merytorycznych kryteriów oceny wniosku o dofinansowanie projektu, określono informacje, które zgodnie z Przewodnikiem po kryteriach wyboru podlegają weryfikacji. Na ich podstawie oraz na podstawie wymagań określonych dla składników wniosku i biznes planu określono zakres wymaganej wiedzy. Następnie przedstawiono braki, które powodują negatywną ocenę danego kryterium oraz syntetycznie określono obszary, których dotyczy brak wiedzy.

Dokumentacji konkursowa określa wymagania, które musi spełnić dokumentacji projektowa. Dla dokumentacji projektowej dokumentacji konkursowa określa:

- ogólne wymagania dot. systemu B2B,
- wymagania dotyczące projektu wytworzenia i wdrożenia systemy B2B.

Negatywnie oceniony wniosek nie spełnia wymagań określonych w dokumentacji konkursowej. Niespełnienie wymagań określonych w dokumentacji konkursowej wynika z:

- braku wiedzy (nieznajomości) dokumentacji konkursowej,
- braku wiedzy umożliwiającej zastosowanie się do wymagań wyartykułowanych w dokumentacji konkursowej.

Dlatego wiedza, która została zweryfikowana w każdym kryterium dotyczy:

- znajomości wymagań wyartykułowanych w dokumentacji konkursowej,
- znajomości specyfikowania wymagań.

2.1. Kryterium nr 1: Przedmiot projektu jest zgodny z celami PO IG oraz działania 8.2 PO IG

Zgodnie z Przewodnikiem po kryteriach wyboru badamy, czy:

- Przedmiot projektu jest zgodny z celem i zakresem merytorycznym Działania.

- Planowany do realizacji system przeznaczony jest do automatycznej komunikacji handlowej (wymiany danych) oraz koordynacji działań pomiędzy przedsiębiorstwami.
- Projekt polega na przeprowadzeniu inwestycji.

Wymagany zakres wiedzy:

- określanie celów przedsięwzięcia z punktu widzenia wnioskodawcy i partnerów,
- określanie zakresu oraz kontekstu przedsięwzięcia,
- ogólne określenie wymagań systemu B2B,
- opracowanie koncepcji systemu,
- znajomość obowiązującej definicji systemu klasy B2B,
- ogólny opis systemu informatycznego,
- określanie danych przesyłanych między systemami informatycznymi,
- znajomość obowiązującego prawa.

Typowe błędy:

- Projekt dotyczy systemu B2C a nie B2B.
- Między wnioskodawcą a firmą wymienioną jako partner nie zachodziła współpraca potwierdzona udokumentowanymi zdarzeniami gospodarczymi.
- Dofinansowanie ma być wykorzystane na systemu, którego właścicielem będzie wnioskodawca i który będzie udostępniany partnerom bez integracji z ich systemami.
- Wniosek nie zawiera informacji jakie systemy informatyczne partnerów będą integrowane z systemem wnioskodawcy i jakie dane będą automatycznie przesyłane.

Negatywna ocena tego kryterium była konsekwencją braku wiedzy dotyczącej opisu koncepcji systemu informatycznego – określenia celu i zakresu systemu oraz jego integracji z innymi systemami.

2.2. Kryterium nr 2: Projekt jest zgodny ze zdefiniowanymi potrzebami wnioskodawcy oraz współpracujących przedsiębiorstw

Zgodnie z Przewodnikiem po kryteriach wyboru: „Badamy, czy wnioskodawca określił potrzeby swoje i współpracujących przedsiębiorstw, związane z automatyzacją wymiany informacji w ramach realizowanych lub planowanych do realizacji procesów biznesowych oraz uzasadnił konieczność realizacji projektu będącego przedmiotem wniosku".

Wymagany zakres wiedzy:

- identyfikacja i analiza potrzeb własnych wnioskodawcy i partnerów biznesowych w zakresie automatyzacji wymiany informacji w ramach procesów biznesowych objętych projektem,
- uzasadnienie konieczności realizacji projektu,
- identyfikacja procesów biznesowych,
- opis danych związanych z procesem biznesowym.

Typowe błędy:

- Brak opisu potrzeb wnioskodawcy lub partnerów związanych z automatyzacją wymiany informacji w ramach realizowanych lub planowanych do realizacji procesów biznesowych.
- Brak uzasadnienia konieczności realizacji projektu będącego przedmiotem wniosku.
- Projekt zaspokoi potrzeby tylko wnioskodawcy a nie współpracujących przedsiębiorstw.
- Projekt nie jest zgodny ze zdefiniowanymi potrzebami wnioskodawcy i współpracujących przedsiębiorstw.
- Stosowanie dla jednego procesu biznesowego różnych nazw.

- Brak informacji jakie informacje biznesowe będą przedmiotem automatycznej komunikacji wewnątrz systemu B2B.

Zauważono brak wiedzy dotyczącej identyfikacji i analizy potrzeb własnych wnioskodawcy i partnerów biznesowych w zakresie automatyzacji wymiany informacji w ramach procesów biznesowych objętych projektem.

2.3. Kryterium nr 3: Projekt zwiększa efektywność procesów stosowanych przez wnioskodawcę we współpracy z partnerami biznesowymi

Zgodnie z Przewodnikiem po kryteriach wyboru: „Analizujemy projekt pod kątem zwiększenia efektywności procesów biznesowych realizowanych pomiędzy współpracującymi przedsiębiorstwami. Badamy, jakie procesy biznesowe współpracujących przedsiębiorstw i Wnioskodawcy zostaną wsparte poprzez automatyzację komunikacji handlowej oraz jakie korzyści będą z tego wynikały dla wnioskodawcy i jego partnerów oraz na czym polega wzrost efektywności tych procesów ?"

Wymagany zakres wiedzy:

- modelowanie procesów biznesowych,
- miary efektywności procesów biznesowych,
- opis dialogu człowieka z systemem informatycznym,
- różnice pomiędzy procesami biznesowymi zachodzącymi z dostawcami i odbiorcami.

Typowe błędy:

- Brak precyzyjnego opisu przebiegu procesów biznesowych przed i po integracji z systemami partnerów biznesowych.
- Brak uzasadnienia jak wdrożenie systemu B2B wpłynie na zwiększenie efektywności procesów stosowanych przez wnioskodawcę we współpracy z partnerem biznesowym.
- Brak precyzyjnego podania miar efektywności procesów biznesowych.
- Stosowanie tego samego opisu procesów dot. współpracy z dostawcą i odbiorcą.
- Utożsamianie procesu biznesowego z modułem (funkcją) systemu informatycznego.
- Brak opisu jakie informacje pozyskiwać będzie wnioskodawca z systemów partnerów a jakie partnerzy z systemu wnioskodawcy.
- Twierdzenie, że tego samego rodzaju informacje będą przesyłane w komunikacji z dostawcą oraz odbiorcą.

Stwierdzono brak wiedzy dotyczącej identyfikacji, opisu, analizy i miar procesów biznesowych objętych projektem.

2.4. Kryterium nr 4: Planowane wydatki są kwalifikowalne w ramach działania, uzasadnione, racjonalne i adekwatne do zakresu i celów projektu

Zgodnie z Przewodnikiem po kryteriach wyboru, badamy czy:

- Przedstawione we wniosku wydatki kwalifikowane zostały określone prawidłowo, tj. zgodnie z definicjami i katalogiem wydatków kwalifikowalnych dla Działania.
- Przedstawione w projekcie wydatki są dostatecznie uzasadnione i racjonalne z punktu widzenia realizacji danego projektu oraz adekwatne do jego zakresu i celu, np. czy są niezbędne do prawidłowej realizacji projektu.
- Wydatki zostały zaprezentowane we wniosku w sposób niedostatecznie konkretny i uniemożliwiający pełną ocenę wszystkich aspektów kwalifikowalności.
- Poszczególne wydatki nie zostały w sposób znaczący przeszacowane.

Wymagany zakres wiedzy:

- modele wytwarzania oprogramowania,
- metody wyceny pracochłonności wytwarzania oprogramowania,
- metody wyceny usług: eksperckich, wdrożeniowych, doradczych, szkoleniowych,
- dekompozycja oprogramowania na komponenty składowe,
- identyfikacja wydatków dotyczących wytworzenia i wdrożenia systemu informatycznego,
- identyfikacja i opis oprogramowania systemowego i bazodanowego,
- identyfikacja i opis sprzętu,
- identyfikacja i opis usług doradczych i wdrożeniowych,
- aktualne ceny rynkowe sprzętu, oprogramowania i usług,
- klasyfikacja wydatków w działaniu 8.2 POIG.

Typowe błędy:

- Brak lub zbytnia ogólnikowość opisu wytwarzania i wdrożenia oprogramowania.
- Niestosowanie modeli wytwarzania i wdrażania oprogramowania.
- Brak podania metody oszacowania wysokości wydatku.
- Brak podania metody wyceny pracochłonności i kosztu wytworzenie oprogramo-wania oraz usług.
- Wnioskowanie o dofinansowanie wydatków niekwalifikowanych w działaniu 8.2: wynagrodzeń pracowników, kosztów zarządzania projektem, promocji projektu.
- Zaplanowanie wykorzystywanie zakupionych dóbr do innych celów niż działanie systemu B2B.
- Ponoszenie wydatków na rzecz partnerów biznesowych,
- Łączenie w jednej pozycji wielu wydatków.
- Brak precyzyjnego opisu wydatku np. nie podanie dla szkolenia: zakresu, liczby uczestników, czasu trwania szkolenia, stawki godzinowej (dziennej).
- Brak wystarczającego uzasadnienia wydatku.
- Planowanie wydatków, które nie odpowiadają zakresowi i celowi projektu.
- Podanie zbyt wysokiej ceny.

Analiza spełnienia tego kryterium umożliwiła stwierdzenie braku wiedzy dotyczącej modeli wytwarzania i wdrażania systemów informatycznych oraz szacowania ich kosztów.

2.5. Kryterium nr 5: W projekcie stosowane są innowacyjne rozwiązania informatyczne

Zgodnie z Przewodnikiem po kryteriach wyboru:

„Analizujemy projekt pod katem innowacyjności przedstawionych rozwiązań informatycznych, dotyczących automatyzacji procesów biznesowych pomiędzy współpracującymi przedsiębiorstwami.

Badamy jedynie deklarowaną innowacyjność na poziomie przedsiębiorstwa. Wnioskodawca powinien uzasadnić, że planowane rozwiązania informatyczne doprowadzą do zwiększenia efektywności realizowanych procesów biznesowych dzięki zastosowaniu mediów elektronicznych (pokazać wartość dodaną projektu: w jakim zakresie przedsiębiorstwo będzie lepiej funkcjonowało po zakończeniu projektu).”

Wymagany zakres wiedzy:

- znajomości technologii systemów informatycznych,
- rodzaje i opis innowacji,

- opis zmian umożliwionych przez system informatyczny,
- uzasadnienie, że zwiększenie efektywności realizowanych procesów biznesowych nastąpi w wyniku wdrożenia systemu informatycznego.

Typowe błędy:

- Nie zamieszczenie wiarygodnego opisu zmian, które umożliwi wdrożenie systemu B2B.
- Brak uzasadnienia, że wdrożenie systemu umożliwi innowacyjność na poziomie przedsiębiorstwa.
- Brak wykazania wartości dodaną projektu: w jakim zakresie przedsiębiorstwo będzie lepiej funkcjonowało po zakończeniu projektu
- Brak podania lub zbyt ogólnikowy opis technologii systemu B2B.
- Zbyt ogólnikowy opisu systemu B2B.

Negatywna ocena tego kryterium była konsekwencją braku wiedzy dotyczącej możliwych do uzyskania efektów wdrożenia systemu informatycznego klasy B2B.

2.6. Kryterium nr 6: Wskaźniki produktu i rezultatu są: obiektywnie weryfikowalne, odzwierciedlają założone cele projektu, adekwatne dla danego rodzaju projektu, realne do osiągnięcia

Zgodnie z Przewodnikiem po kryteriach wyboru:

„Badamy prawidłowość informacji dotyczących wskaźników produktu i rezultatu, mając na uwadze specyfikę działań przewidzianych w ramach projektu oraz spójność wskaźników z całym projektem. Badaniu będzie podlegało, czy wskaźniki oceny odpowiednio odwierciedlają deklarowane efekty projektu oraz czy są możliwe do zmierzenia (na podstawie danych, gromadzonych w tym celu przez Wnioskodawcę/Beneficjenta oraz danych ogólnodostępnych z jednoznacznie zidentyfikowanego, wiarygodnego źródła).

Sprawdzamy, czy określono wskaźniki produktu i rezultatu zgodnie z Instrukcją wypełniania wniosku.

Sprawdzamy, czy wnioskodawca podał wartość dla wszystkich pięciu obligatoryjnych wskaźników produktu. Ponadto sprawdzamy, czy Wnioskodawca określił wartość dwu obligatoryjnych wskaźników rezultatu oraz przedstawił prawidłowo korespondujące cele na poziomie rezultatu."

Wymagany zakres wiedzy:

- miary efektywności procesów biznesowych,
- identyfikacja wskaźników odzwierciedlających cele projektu,
- identyfikacja wskaźników adekwatnych dla danego rodzaju projektu,
- uzasadnienie realności i możliwości osiągnięcia przez wnioskodawcę wartości wskaźników,
- weryfikacja wymagań,
- pomiar wyników wdrożenia systemu informatycznego,
- metody weryfikacji osiągnięcia przez wskaźnik określonej wartości,
- pomiar przychodów i kosztów będących konsekwencją wdrożeniem systemu informatycznego.

Typowe błędy:

- Różne wartości wskaźników podane we wniosku i biznes planie.
- Podane we wniosku i biznes planie nazwy i/lub wartości wskaźnika są niespójne.
- Brak wiedzy na temat metod weryfikacji wdrożenia systemu B2B.
- Brak wiedzy na temat metod weryfikacji posiadania przez system B2B określonych cech.

- Brak wskazania obiektywnych źródeł danych dla weryfikacji wartości (bazowych i docelowych) wszystkich wskaźników.
- Nieprawidłowe podanie wartość dwu obligatoryjnych wskaźników rezultatu.
- Brak wiarygodnego uzasadnienia wartości wskaźnika określającego prognozowaną zmianę przychodów lub kosztów.
- Brak podania metody weryfikacji wartości wskaźnika albo jej bardzo ogólnikowy opis.
- Zaproponowanie wskaźnika, który nie odzwierciedla celów projektu (np. zatrudnienie nowych pracowników).
- Podanie wskaźnika nieadekwatnego dla danego rodzaju projektu, np. dzienna liczba odsłon portalu.
- Dla specyficznych projektów brak określenia dodatkowych wskaźników rezultatu oraz przedstawienia opisu powiązanych z nimi celów na poziomie rezultatu.
- Brak informacji umożliwiający ocenę realności osiągnięcia podanej wartości wskaźnika.

Ocena spełnienia tego kryterium umożliwiła stwierdzenie braku wiedzy dotyczącej wskaźników ilustrujących wdrożenie systemu informatycznego i osiągnięcie efektów wdrożenia, w tym efektów ekonomicznych oraz ich pomiar.

2.7. Kryterium nr 7: Projekt ma co najmniej neutralny wpływ na polityki horyzontalne UE wymienione w art. 16 i 17 rozporządzenia Rady (WE) nr 1083/2006

Zgodnie z Przewodnikiem po kryteriach wyboru: „Na etapie oceny merytorycznej obligatoryjnej ocenie podlegać będzie, czy projekt ma co najmniej neutralny wpływ na polityki horyzontalne UE. Ocena dokonywana będzie głównie z punktu widzenia wykluczenia negatywnego wpływu. Ocenie podlega zgodność poszczególnych opisów i deklaracji dotyczących planowanego projektu z politykami horyzontalnymi UE.”…

Wymagany zakres wiedzy:

- znajomość polityk horyzontalnych UE,
- uzasadnienie co najmniej neutralnego wpływu projektu na polityki horyzontalne UE.

Typowe błędy:

- Zadeklarowano ale nie uzasadniono, że projekt ma co najmniej neutralny wpływ na polityki horyzontalne UE.
- Brak scharakteryzowania w projekcie polityki równości szans kobiet i mężczyzn oraz nie dyskryminacji i zrównoważonego rozwoju.
- Brak uzasadnienia, że projekt ma neutralny lub pozytywny wpływ na środowisko.
- W deklaracjach i uzasadnieniach pominięto którąś z polityk horyzontalnych UE.

Stwierdzono brak wiedzy nt sposobów uzasadnienia, że projekt ma co najmniej neutralny wpływ na polityki horyzontalne UE.

2.8. Kryterium nr 8: Środki będą wykorzystane w sposób efektywny (relacja nakład/ rezultat)

Zgodnie z Przewodnikiem po kryteriach wyboru:

„Badamy, czy planowany projekt jest możliwy do realizacji z technicznego, finansowego oraz organizacyjnego punktu widzenia oraz czy plan działań przedstawiony we wniosku o dofinansowanie opisany jest szczegółowo i określa poszczególne etapy projektu.

Sprawdzamy również, czy Wnioskodawca posiada odpowiednie zasoby niezbędne do realizacji projektu, o których współfinansowanie nie występuje we wniosku. Sprawdzamy,

czy wnioskodawca w sposób czytelny przedstawił wiarygodne źródła finansowania projektu na poszczególnych etapach jego realizacji."…

Wymagany zakres wiedzy:

- szacowanie wysokości wydatków,
- uzasadnienie konieczności poniesienia wydatku,
- ocena efektywności inwestycji,
- wskaźniki efektywności inwestycji,
- uzasadnienie czy, kiedy i jak planowane nakłady na inwestycję zostaną zwrócone,
- metody obliczania wartości wskaźników efektywności,
- efektywność wykorzystania funduszy publicznych,

Typowe błędy:

- Brak możliwości stwierdzenia, że poszczególne wydatki zostaną wykorzystane zgodnie z zasadami gospodarki finansowej opartej na zasadach bilansowych, w szczególności najkorzystniejszej relacji nakładów do rezultatów.
- Zbyt ogólnikowy opis wydatków.
- Brak uzasadnienia wydatków.
- Zawyżone wydatki.
- Przekraczający 10% wszystkich wydatków udział wydatków uznanych za niekwalifikowalne.
- Brak wiarygodnego uzasadnienia wartości wskaźnika określającego prognozowaną zmianę przychodów lub kosztów.
- Niska wartość współczynnika efektywności ekonomicznej.
- Brak podania metody obliczenia okresu zwrotu projektu.
- Brak wskazania jakim sposobem planowane nakłady na inwestycję zostaną zwrócone.
- Marnotrawienie funduszy publicznych, np:
  - sztuczny podział projektu na wiele mniejszych projektów,
  - integracji systemu informatycznego firmy A wytworzonego przez firmę informatyczną B z systemem informatycznym tejże firmy B.

Stwierdzono brak wiedzy nt szacowania wydatków projektu informatycznego oraz metod oceny jego efektywności.

2.9. Kryterium nr 9: Projekt jest wykonalny technicznie i finansowo, a jego przebieg rzeczowo-finansowy jest czytelny, szczegółowy i realny do wykonania

Zgodnie z Przewodnikiem po kryteriach wyboru: „Badamy, czy planowany projekt jest możliwy do realizacji z technicznego, finansowego oraz organizacyjnego punktu widzenia oraz czy plan działań przedstawiony we wniosku o dofinansowanie opisany jest szczegółowo i określa poszczególne etapy projektu. Sprawdzamy również, czy Wnioskodawca posiada odpowiednie zasoby niezbędne do realizacji projektu, o których współfinansowanie nie występuje we wniosku. Sprawdzamy, czy wnioskodawca w sposób czytelny przedstawił wiarygodne źródła finansowania projektu na poszczególnych etapach jego realizacji." …

Wymagany zakres wiedzy:

- uzasadnienie realności technicznej realizacji projektu,
- uzasadnienie realności organizacyjnej realizacji projektu,
- wybór i uzasadnienie przyjętej technologii,
- uzasadnienie realności finansowej realizacji projektu,

- czytelne przedstawienie wiarygodnych źródeł finansowania projektu na poszczególnych etapach jego realizacji.
- sporządzanie szczegółowego harmonogramu rzeczowo - finansowego,
- uzasadnienie posiadania odpowiednich zasobów,

Typowe błędy:

- Wnioskodawca nie posiada odpowiednich zasobów(technicznych, osobowych, lokalowych itp.) niezbędnych do realizacji projektu, o których współfinansowanie nie występuje we wniosku.
- Brak informacji uzasadniających wykonalność techniczną projektu.
- Brak uzasadnienia możliwości realizacji projektu od strony organizacyjnej.
- Brak przedstawienia w czytelny sposób wiarygodnych źródeł finansowania projektu na poszczególnych etapach jego realizacji.
- Brak dla określonych etapów projektu środków finansowych na ich realizację.
- Brak przedstawienia logicznie ułożonego i szczegółowego planu działań prowadzących do osiągnięcia założonych celów;
- Niespójność przebiegu rzeczowo-finansowego i planu działań.
- Nieczytelny, zbyt mało szczegółowy przebieg rzeczowo-finansowy.
- Łączne przedstawianie w jednej pozycji wielu wydatków.

Negatywna ocena tego kryterium była konsekwencją braku wiedzy nt planowania realizacji projektu od strony technicznej, organizacyjnej i finansowej.

3. Analiza obszarów braku wiedzy z zakresu specyfikowania wymagań

W punkcie 3.1 zidentyfikowano najczęściej występujące obszary braku wiedzy nt specyfikowania wymagań systemu klasy B2B. Wyniki dotyczą projektów dofinansowania w ramach działania 8.2 POIG "Wspieranie wdrażania elektronicznego biznesu typu B2B".

W punkcie 3.2, dla tych samych kryteriów jak w działaniu 8.2, krótko przedstawiono najczęściej występujące obszary braku wiedzy nt specyfikowania wymagań dla systemów umożliwiających świadczenie e-usług. Wyniki dotyczą projektów dofinansowania w ramach działania 8.1 POIG "Wspieranie działalności gospodarczej w dziedzinie gospodarki elektronicznej".

3.1. Najczęściej występujące obszary braku wiedzy - działanie 8.2 POIG

Podczas szkoleń ekspertów PARP [5] podał jaki procent wniosków dot. działania 8.2 POIG - "Wspieranie wdrażania elektronicznego biznesu typu B2B" nie spełnia danego obligatoryjnego kryterium merytorycznego. Dane dotyczą I naboru w 2011r. W tabeli 2 przedstawiono te dane.

Dane te wraz z przeprowadzoną w pracy analizą wymaganej wiedzy umożliwiły identyfikacje obszarów braku wiedzy wnioskodawców nt specyfikowania wymagań. Biorąc pod uwagę częstość występowania braki wiedzy dotyczą następujących obszarów:

- wskaźniki ilustrujące wdrożenie systemu informatycznego i osiągnięcie efektów wdrożenia, w tym efektów ekonomicznych oraz ich pomiar,
- modele wytwarzania i wdrażania systemów informatycznych,
- opis i szacowanie wysokości wydatków projektu informatycznego,
- opis koncepcji systemu informatycznego – określenie celu i zakresu systemu oraz jego integracji z innymi systemami,
- planowanie realizacji projektu od strony technicznej, organizacyjnej i finansowej.

Tabela 2. Dz. 8.2 POIG - Udział wniosków niespełniających dane kryterium obligatoryjne

lp	nazwa merytorycznego kryterium obligatoryjnego	%
1.	Przedmiot projektu jest zgodny z celami PO IG oraz działania 8.2 POIG	20
2.	Projekt jest zgodny ze zdefiniowanymi potrzebami wnioskodawcy oraz współpracujących przedsiębiorstw	7
3.	Projekt zwiększa efektywność procesów stosowanych przez wnioskodawcę we współpracy z partnerami biznesowymi	8
4.	Planowane wydatki są kwalifikowalne w ramach działania, uzasadnione, racjonalne i adekwatne do zakresu i celów projektu	50
5.	W projekcie stosowane są innowacyjne rozwiązania informatyczne	2
6.	Wskaźniki produktu i rezultatu są: obiektywnie weryfikowalne, odzwierciedlają założone cele projektu, adekwatne dla danego rodzaju projektu, realne do osiągnięcia	64
7.	Projekt ma co najmniej neutralny wpływ na polityki horyzontalne UE wymienione w art. 16 i 17 rozporządzenia Rady (WE) nr 1083/2006	0,3
8.	Środki będą wykorzystane w sposób efektywny (relacja nakład/ rezultat)	43
9.	Projekt jest wykonalny technicznie i finansowo, a jego przebieg rzeczowo-finansowy jest czytelny, szczegółowy i realny do wykonania	18

W działaniu 8.2 POIG wnioskodawcy to najczęściej istniejące na rynku kilka lub kilkanaście lat firmy, zatrudniające nawet kilkadziesiąt pracowników. Uzyskane wyniki nie świadczą o należytym zaangażowaniu wnioskodawcy w przygotowanie dokumentacji projektowej. Projekt ma strategiczne znaczenia dla wnioskodawcy. Wnioskodawca dużą cześć projektu będzie finansował ze środków własnych. Dlatego powinien być zainteresowany precyzyjnym definiowaniem i pomiarem wskaźników wdrożenia systemu oraz szczegółowym opisem i uzasadnieniem planowanych wydatków czy czytelnym opracowaniem koncepcji systemu. Jednak w dokumentacji projektowej wnioskodawca często nie zamieszcza wymaganych informacji.

Wnioskodawca posiada własne systemy informatyczne i często zatrudnia specjalistów informatyków. Standardowym uzasadnieniem oszacowania kosztów oprogramowania, wdrożenia i sprzętu są oferty dostawców. Wnioski o dofinansowanie często sporządzane są przez firmy konsultingowe. Uzyskane wyniki świadczą zatem również o poziomie wiedzy wszystkich stron uczestniczących w sporządzaniu dokumentacji projektowej:

- pracowników, często specjalistów informatyków,
- dostawców,
- firm konsultingowych.

3.2. Najczęściej występujące obszary braku wiedzy – działanie 8.1 POIG

Autor oceniał również projekty dotyczące dofinansowanie systemów umożliwiających świadczenie e-usług – działanie 8.1 POIG "Wspieranie działalności gospodarczej w dziedzinie gospodarki elektronicznej".

W latach 2010 - 2011 liczba złożonych wniosków dot. działania 8.1 była większa niż dla działania 8.2. i wynosiła: I nabór /2010r. – 1850, I/2011r. - 1400, II/2011r. – 746.

Tabela 3. Dz. 8.1 i 8.2 - udział wniosków niespełniających danego kryterium obligatoryjnego

Nr wg 8.2	**nazwa merytorycznego kryterium obligatoryjnego**	**dz 8.1 I/2011**	**dz 8.1 II/2011**	**dz 8.2 I/2011**
1.	Przedmiot projektu jest zgodny z celami PO IG oraz działania 8.1/8.2 PO IG	18%	16%	20%
4.	Planowane wydatki są kwalifikowalne w ramach działania, uzasadnione, racjonalne i adekwatne do zakresu i celów projektu	51%	38%	50%
6.	Wskaźniki produktu i rezultatu są: obiektywnie weryfikowalne, odzwierciedlają założone cele projektu, adekwatne dla danego rodzaju projektu, realne do osiągnięcia	65%	48%	64%
7.	Projekt ma co najmniej neutralny wpływ na polityki horyzontalne UE wymienione w art. 16 i 17 rozporządzenia Rady (WE) nr 1083/2006	0,2%	0,5%	0,3%
8.	Środki będą wykorzystane w sposób efektywny (relacja nakład/ rezultat)	48%	33%	43%
9.	Projekt jest wykonalny technicznie i finansowo, a jego przebieg rzeczowo-finansowy jest czytelny, szczegółowy i realny do wykonania	37%	32%	18%

Procentowy udział pozytywnie ocenionych wniosków był mniejszy niż dla działania 8.2. Autor ocenił działaniu 8.1 166 wniosków. Podczas szkoleń PARP również dla projektów dot. działania 8.1 podał jaki procent wniosków nie spełnia danego kryterium merytorycznego. Dane te dotyczą naborów I/2011r. [5] i II/2011r. [6]. Wnioski dotyczące działania 8.1 oceniane były pod względem spełnienia trzynastu merytorycznych kryteriów obligatoryjnych, a dotyczące działania 8.2 - dziewięciu. Część kryteriów oceny merytorycznej dla działania 8.1 była taka sama jak dla działania 8.2. Umożliwia to częściowe porównanie wyników. W tabeli 3 podano za PARP jaki procent wniosków nie spełnia danego wspólnego kryterium. Dane te pokazują, że dla działania 8.1, w przypadków tych samych kryteriów jak w działaniu 8.2 braki dot. obszarów wiedzy nt specyfikowania wymagań systemu informatycznego występowały podobnie często.

W działaniu 8.1 POIG wnioskodawcy to najczęściej firmy jedno- lub dwuosobowe, istniejące krócej niż 1 rok. Udziałowcy firm ubiegających się o dofinansowanie w ramach dz. 8.1 reprezentują wszystkie profesje. Często są to bardzo młodzi ludzie, nawet studenci. Ich doświadczanie biznesowe i informatyczne są bardzo różne. Oczywiście w działaniu 8.1 dokumentacja projektowa też często powstaje we współpracy wnioskodawcy z informatykami, dostawcami rozwiązań IT oraz firmami konsultingowymi. Zidentyfikowane obszary braku wiedzy dotyczą zatem również osób i firm współpracujących z wnioskodawcami.

4. Podsumowanie

Możliwość dofinansowania, w ramach działania 8.2 POIG - „Wspieranie wdrażania elektronicznego biznesu typu B2B" cieszy się dużym zainteresowaniem. W każdym z lat 2010 i 2011 złożono ponad jeden tysiąc wniosków. W pracy uzasadniono, że jedną z głównych z barier uzyskania dofinansowania jest brak wiedzy nt specyfikowania wymagań sys-

temu B2B. Dlatego brak wiedzy nt specyfikowania wymagań systemu B2B można uważać jak główna barierę „wdrażania elektronicznego biznesu typu B2B".

Unikalność uzyskanych wyników polega tym, że dokumentacja projektów informatycznych stanowią tajemnice firm i uzyskanie od dużej liczby firm dokumentacji jest niemożliwe. Ocena 45 wniosków o dofinansowanie projektów, analiza dokumentacji ich dotyczących oraz dane udostępnione przez PARP pozwoliły na dokonanie diagnozy stanu podstawowej wiedzy dot. specyfikowania wymagań w firmach ubiegających się o dofinansowanie wdrożenie systemów klasy B2B. Uzyskane wyniki świadczą również o poziomie wiedzy stron uczestniczących w sporządzaniu dokumentacji projektowej: pracowników wnioskodawcy, dostawców IT, firm konsultingowych. Podobne wyniki uzyskano również dla projektów dot. dz 8.1 POIG - "Wspieranie działalności gospodarczej w dziedzinie gospodarki elektronicznej".

Osiągnięte wyniki mają charakter specyficznego kompendium wiedzy eksperta oceniającego wnioski. Wiedza ta jest zgodna ze statystkami publikowanymi przez PARP. Przedstawione wyniki mogą pomóc firmom ubiegającym się o dofinansowanie prawidłowo przygotować wniosek o dofinansowanie a następnie go rozliczyć.

Bibliografia

[1] Strona WWW PARP dotycząca działania 8.2 POIG poświęcona naborowi wniosków: *http://poig.parp.gov.pl/index/index/748*

[2] Rozporządzenie Ministra Rozwoju Regionalnego z dnia 13 sierpnia 2008 r. w sprawie udzielania przez Polską Agencję Rozwoju Przedsiębiorczości pomocy finansowej na wspieranie tworzenia i rozwój gospodarki elektronicznej w ramach Programu Operacyjnego Innowacyjna Gospodarka, 2007-2013: *http://www.parp.gov.pl/index/more/8311*

[3] Szczegółowy opis priorytetów PO IG, 2007-2013 Narodowe Strategiczne Ramy Odniesienia na lata 2007-2013: *http://www.poig.gov.pl/Dokumenty/Documents/uszcz_POIG_v6_101008.pdf*

[4] Błędy popełniane we wnioskach aplikacyjnych, dotyczących dz. 8.2 POIG: *http://poig.parp.gov.pl/index/index/1273*

[5] Dane dotyczące negatywnie ocenionych kryteriów dla wniosków złożonych w ramach Dz. 8.1 i 8.2 POIG, Michał Wiśniewski, materiały szkoleniowe, praca nieopublikowana

[6] Dane dotyczące negatywnie ocenionych wniosków dot. dz. 8.1 POIG: Michał Wiśniewski – materiały szkoleniowe z dnia 13 IV 2012r: *http://poig.parp.gov.pl/files/74/108/277/550/13344.pdf*

[7] Przewodnik po kryteriach wyboru finansowanych operacji w ramach Programu Operacyjnego Innowacyjna Gospodarka, 2007-2013: *http://poig.parp.gov.pl/files/74/108/212/548/12829.pdf*

Wykorzystanie szyny ESB do efektywnej synchronizacji danych pomiarowych dużej objętości

Przemysław Wyszkowski

Akademia Górniczo-Hutnicza w Krakowie, Katedra Informatyki

Streszczenie:

Eksperyment TOTEM (Total Cross Section, Elastic Scattering and Diffraction Dissociation at the LHC) w Europejskiej Organizacji Badań Jądrowych (CERN) bada całkowity przekrój czynny, rozpraszanie elastyczne i dysocjację dyfrakcyjną protonów ulegających kolizjom w LHC (Wielki Zderzacz Hadronów). Aby badanie tych zjawisk było możliwe, konieczne jest przetwarzanie ogromnej ilości danych pochodzących z różnych źródeł: detektorów TOTEM'u, detektorów eksperymentu CMS (Compact Muon Solenoid), aparatury pomiarowej zainstalowanej w tunelu Wielkiego Zderzacza Hadronów oraz wielu innych, pomniejszych systemów zewnętrznych. W celu umożliwienia wydajnej i wygodnej pracy naukowców niezbędnym jest zapewnienie centralnego punktu, w którym wszystkie potrzebne dane (tak w stanie surowym jak i przetworzonym) będą przechowywane. Praca ma na celu prezentację wykorzystania koncepcji szyny integracyjnej Enterprise Service Bus (ESB) w budowie infrastruktury programowej do efektywnego przesyłu danych pomiarowych dużej objętości. Zaprezentowane zostały technologie i mechanizmy realizujące koncepcję szyny integracyjnej, model systemu przesyłu danych w oparciu o wykorzystanie kolejek komunikatów, jego implementacja na potrzeby eksperymentu TOTEM oraz badania wydajności i skalowalności rozwiązania.

Słowa kluczowe:

przesył danych, integracja systemów, Service-Oriented Architecture (SOA), OSGi, Enterprise Service Bus (ESB), Message-Oriented Middleware (MOM), Enterprise Integration Patterns (EIP), CERN, TOTEM

1. Wprowadzenie

Nieustający rozwój systemów informatycznych pociąga za sobą ciągłe zmiany technologii, w których są one realizowane. Mimo technicznych różnic dzielących systemy (język implementacji, architektura, protokoły komunikacji czy producent) istnieje potrzeba współpracy istniejących rozwiązań z nowo powstałymi aplikacjami. Problem integracji systemów informatycznych jest dużym wyzwaniem dla instytucji pragnących maksymalizować efektywność swojego działania. Koszty związane z integracją przewyższają często wartość unifikowanego oprogramowania, dlatego temat ten jest niezwykle ważny z perspektywy przemysłu informatycznego, będąc jednocześnie podejmowanym w badaniach naukowych.

Przesyłanie informacji na odległość to podstawowe zadanie stawiane systemom komputerowym. Od czasu narodzin pierwszych sieci komputerowych minęło już kilkadziesiąt lat, a jednak wciąż powstają nowe, doskonalsze rozwiązania, które zwiększają efektywność przetwarzania. W dobie Internetu globalna wymiana informacji jest kluczowym elementem w rozwoju praktycznie każdej gałęzi przemysłu, edukacji a także placówek naukowych.

W niniejszym artykule dyskutowany jest temat integracji heterogenicznych systemów w celu efektywnego dostarczania wysokoobjętościowych danych na potrzeby wspomagania

eksperymentów fizycznych. W pracach wykorzystano koncepcję *ESB* (ang. *Enterprise Service Bus*), poddając próbie potencjał oferowany przez implementujące je rozwiązania.

2. Masowe przetwarzanie danych w eksperymentach fizyki wysokich energii

Eksperymenty *Fizyki Wysokich Energii* (ang. *HEP – High Energy Physics*) są największym producentem danych w historii ludzkości[1]. *CERN, Europejska Organizacja Badań Jądrowych* w Genewie będąc w posiadaniu obecnie największego na świecie akceleratora cząstek elementarnych, tzw. *Wielkiego Zderzacza Hadronów*, (ang. *LHC – Large Hadron Collider*) dostarcza największy strumień danych. Zapanowanie nad ogromem informacji nie byłoby możliwe bez użycia technologii i wiedzy informatycznej.

Dla informatyków pracujących nad usprawnieniem działania eksperymentów fizycznych kluczowym jest zrozumienie w jaki sposób generowane są wartości pomiarowe, jak są reprezentowane i przechowywane, a także w jaki sposób następuje ich przetwarzanie i przesyłanie.

2.1. Cykl życia danych pomiarowych

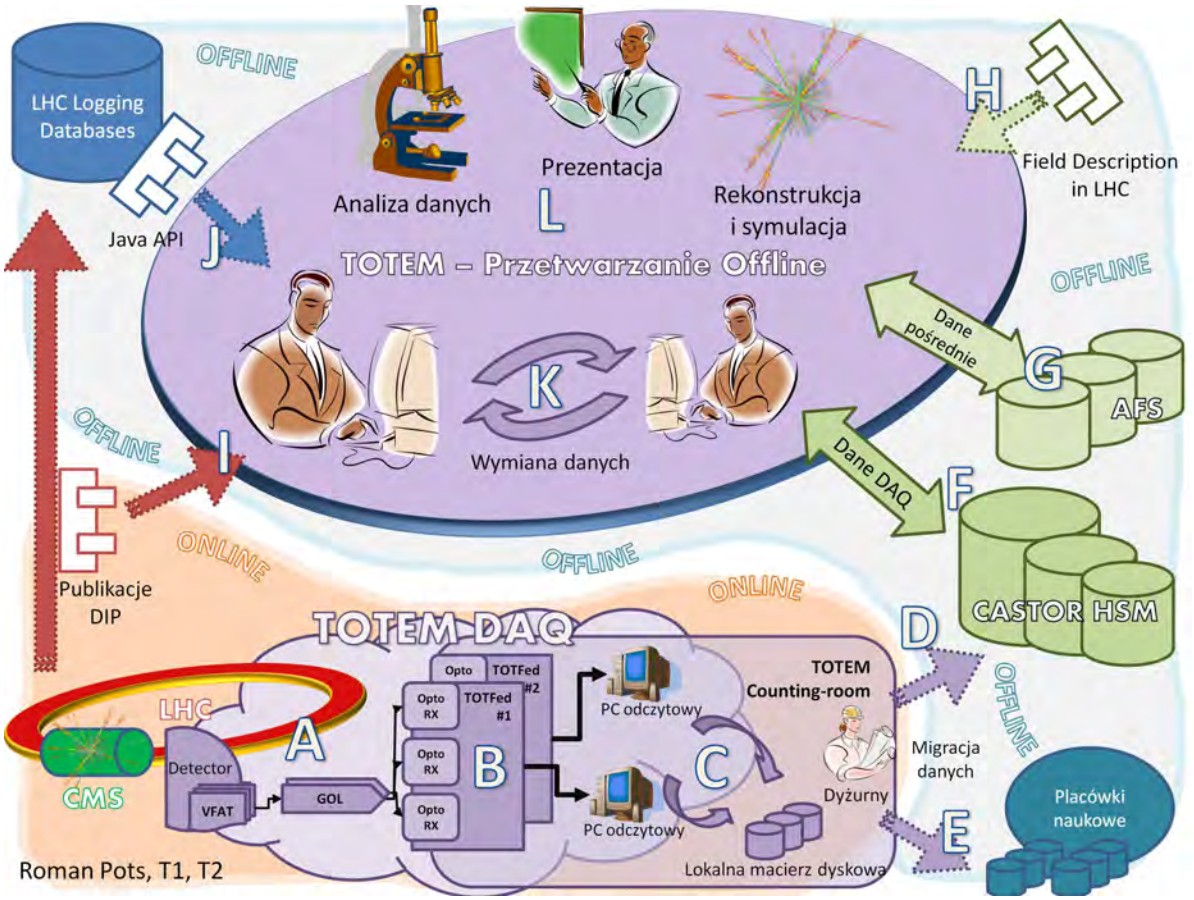
Od zderzenia cząstek wewnątrz detektora do rekonstrukcji parametrów zdarzenia w postaci znaczącego pomiaru fizycznego wiedzie skomplikowana droga złożona z dwóch głównych faz: przetwarzania *Online*, po którym następuje przetwarzanie *Offline*.

Przetwarzanie *Online* związane jest bezpośrednio z aparaturą rejestrującą zdarzenia mające miejsce podczas zderzeń cząstek. W wypadku akceleratorów cząstek elementarnych są to skomplikowane układy elektroniczne rejestrujące wartości w zależności od wartości pola magnetycznego lub elektrycznego jak również innych wielkości fizycznych. W tej fazie, by sprostać twardym wymaganiom czasowym z uwagi na niezwykle dynamiczną naturę eksperymentu, wykorzystywane jest czysto sprzętowe przetwarzanie przy pomocy układów *ASIC* (ang. *Application Specific Integrated Circuit*) oraz *FPGA* (ang. *Field Programmable Gate Array*)[2].

Surowe dane zarejestrowane przez detektory w fazie *Online* nie reprezentują jeszcze istotnych wartości fizycznych. Przykładowa dana binarna może mówić o tym czy komórka detektora została pobudzona polem elektrycznym o wartości powyżej ustalonego progu. Aby dowiedzieć się, czy ta zmiana wartości może reprezentować przemieszczającą się w pobliżu cząstkę obdarzoną ładunkiem elektrycznym należy skorelować ze sobą wartości otrzymywane przez grupy czujników, zaangażować algorytmy wykrywania wzorców, filtrowania i transformat. Wspomniane czynności wchodzą w skład przetwarzania Offline, którego końcowym wynikiem są zrekonstruowane wartości fizyczne takie jak trajektorie cząsteczek czy ich masy lub energie. Dzięki rekonstrukcji naukowcy mogą prześledzić zjawiska zachodzące wewnątrz detektora i wnioskować na ich podstawie o słuszności stawianych tez.

2.2. Eksperyment TOTEM – obecny model obliczeniowy

Eksperyment *TOTEM* (ang. *TOTal Elastic and diffractive cross section Measurement*) jest jednym z mniejszych wśród eksperymentów działających przy tunelu *LHC*. Głównym celem realizowanym przez *TOTEM* jest pomiar całkowitego przekroju czynnego oraz dysocjacji dyfrakcyjnej protonów ulegających kolizjom w *LHC*. Po szerszy opis programu fizycznego realizowanego przez eksperyment należy odwołać się do [3].



Rysunek 1. Obecny model przetwarzania w eksperymencie TOTEM w CERN

Rysunek 1 ilustruje obecny model przetwarzania danych przez naukowców w kolaboracji *TOTEM*. Dane, na których operują naukowcy, pochodzą z różnych źródeł, cechuje je heterogeniczna natura formatu, przechowywania i dostępu. Na rysunku widać podstawowy podział na przetwarzanie w fazach *Offline* i *Online* oraz mnogość źródeł danych, z którymi styczność mają uczestnicy eksperymentu. Aby otrzymać interesujące rezultaty, naukowiec musi wpierw uzyskać dostęp i zgromadzić wymagane przez siebie dane w konkretnym miejscu przestrzeni dyskowej. Po przeprowadzeniu obliczeń dzieli się wynikami z pozostałymi zainteresowanymi na zasadzie wymiany lokalnych zasobów. Kolejne litery alfabetu symbolizują przesył i transformacje danych. Dokładniejszy opis procesów przetwarzania danych (A-L) w eksperymencie *TOTEM* zawarty jest w [4].

Sposób pracy naukowców eksperymentu *TOTEM* nie podąża w pełni za uznanymi standardami pracy zespołowej i nie korzysta w pełni z udogodnień dostarczanych przez współczesną technologię informacyjną. Manualna wymiana oraz brak scentralizowanego repozytorium danych jest źródłem pomyłek oraz obniżonej efektywności pracy.

3. Implementacja system TOTEM DBPop

Aby usprawnić pracę naukowców eksperymentu *TOTEM*, podjęto prace nad zintegrowaniem wszystkich źródeł danych przetwarzanych w fazie *Offline* we wspólnej bazie danych – *TOTEM Offline Database*. Relacyjna baza zrealizowana w technologii Oracle zawiera schemat dwudziestu tabel i jest utrzymywana przez departament *IT* w *CERN*[1].

[1] *http://information-technology.web.cern.ch/*

Wszelkie dane potrzebne do prowadzenia obliczeń (tak w stanie surowym jak i przetworzonym) powinny być przechowywane w owej bazie. Rozwiązanie takie zapewnia szereg korzyści, m.in.: jednolity dostęp do wszystkich potrzebnych danych, brak konieczności wiedzy użytkowników o sposobie ich pozyskiwania jak również zintegrowany sposób pracy z danymi pomiarowymi.

3.1. Wymagania

Projekt migracji z modelu przetwarzania zaprezentowanego w rozdziale 2.2 do scentralizowanego przetwarzania z wykorzystaniem bazy danych jest obecnie w trakcie realizacji. Zadaniem autora było skonstruowanie systemu *TOTEM DBPop* (ang. *Database Population System*), którego celem będzie zautomatyzowane dostarczanie informacji pomiarowych dla bazy danych eksperymentu. Głównym wymaganiem stawianym przed aplikacją jest integracja wszelkich źródeł danych oraz ekspozycja jednorodnego interfejsu dla ich pozyskiwania. System powinien spełniać surowe wymagania dotyczące wydajności przesyłanych danych oraz w sposób zmaksymalizowany utylizować zasoby obliczeniowe optymalnie rozkładając obciążenie.

Na podstawie przeprowadzonej analizy wymagań i analizy domenowej wynika, że postawione zadanie jest w rzeczywistości scenariuszem integracyjnym, dlatego zasadnym jest skorzystanie z koncepcji opracowanych pod tym właśnie kątem.

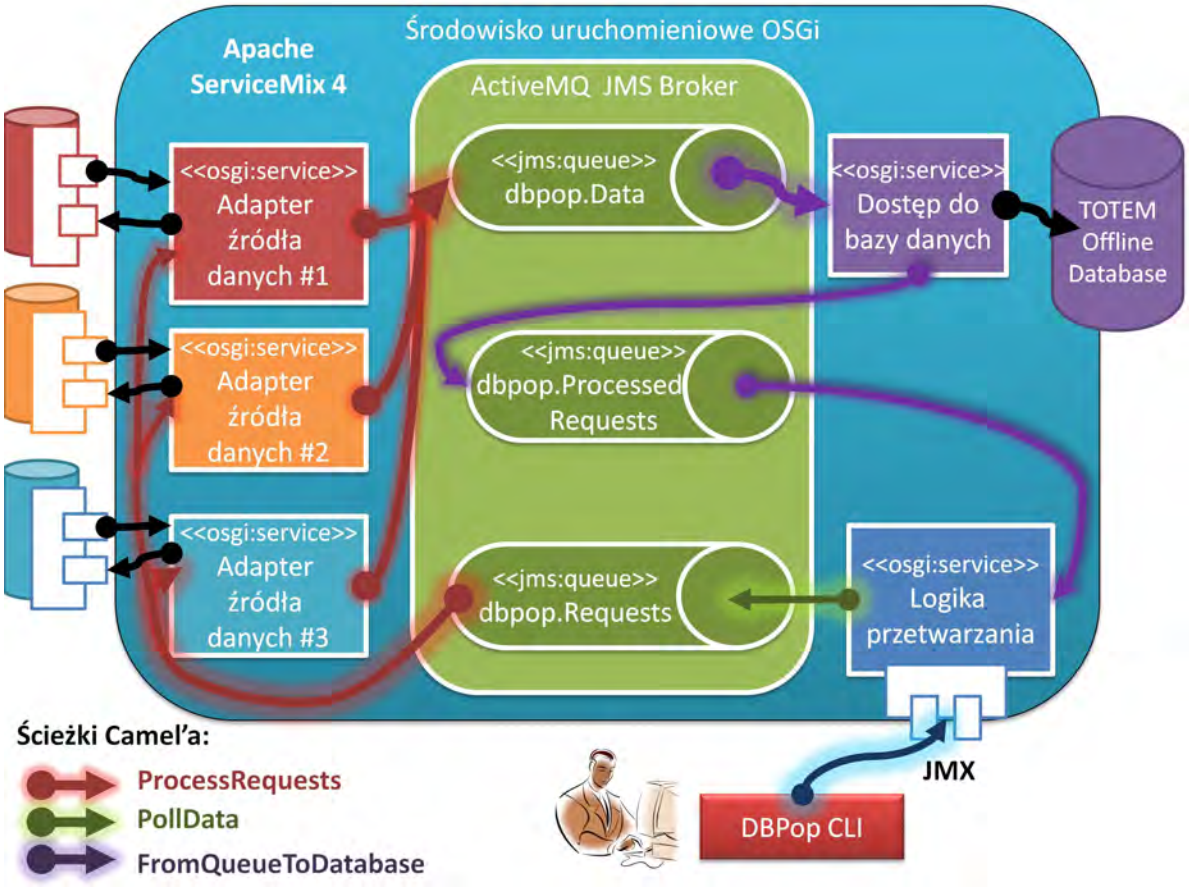
3.2. Architektura

W implementacji rozwiązania postanowiono skorzystać z wiodących obecnie trendów w projektowaniu i konstruowaniu oprogramowania – podejścia zgodnego z paradygmatem Architektury zorientowanej na usługi (ang. *SOA – Service-Oriented Architecture*) w oparciu o szynę *Enterprise Service Bus*. System składa się z luźno powiązanych usług w postaci komponentów osadzonych w środowisku uruchomieniowym, komunikujących się za pośrednictwem infrastruktury do przesyłania komunikatów *MOM* (ang. *Message-Oriented Middleware*) i połączonych wzajemnie przy pomocy mechanizmów *EIP* (ang. *Enterprise Integration Patterns*).

Pamiętać należy, że *ESB* nie jest konkretną technologią ani nawet specyfikacją. Istnieje zestaw wytycznych [5], które powinny być spełnione przez produkt realizujący te pryncypia, w szczególności: wirtualizacja usług (w kontekście stosowanego protokołu komunikacyjnego, eksponowanego interfejsu oraz lokalizacji) czy zapewnienie realizacji tzw. *Cross-cutting concerns* (m.in. logowanie, bezpieczeństwo, monitorowanie i rozliczenia).

Rysunek 2 przedstawia architekturę systemu *DBPop* w oparciu o mechanizmy dostarczane przez otwartą implementację koncepcji *ESB – Apache ServiceMix*[2] w wersji czwartej. ServiceMix jest w rzeczywistości wachlarzem projektów Apache, które posłużyły do realizacji zakładanych przez *ESB* funkcjonalności. *Karaf*[3] zapewnia środowisko uruchomieniowe dla usług, *ActiveMQ*[4] dostarcza środków niezawodnej komunikacji w standardzie *JMS* (ang. *Java Message Service*): kolejek oraz subskrypcji na tematy. *CXF*[5] pozwala na integrację usług przy pomocy otwartych standardów a *Camel*[6] umożliwia deklaratywną konfigurację trasowania między usługami (ang. routing).

[2] *http://servicemix.apache.org/*
[3] *http://karaf.apache.org/*
[4] *http://activemq.apache.org/*
[5] *http://cxf.apache.org/*
[6] *http://camel.apache.org*



Rysunek 2. Architektura systemu TOTEM DBPop w oparciu o otwartą implementację ESB, Apache ServiceMix

Językiem implementacji systemu jest *Java* z uwagi na swoją dojrzałość, niezależność od platformy, bogactwo narzędzi jak i doświadczenie autora w użyciu tej właśnie technologii. *Spring Framework*[7] i zestaw szablonów przez niego dostarczanych jest wykorzystywany ze względu na oferowane mechanizmy odwrócenia sterowania (ang. *IoC – Inversion of Control*) jak również dla uproszczenia struktury projektu. Do interakcji z bazą danych z poziomu kodu aplikacji wykorzystano ujednoliconą specyfikację *JPA* (ang. *Java Persistence API*) wraz z *Hibernate*[8], jako dostawcy funkcjonalności *ORM* (ang. *Object-Relational Mapping*).

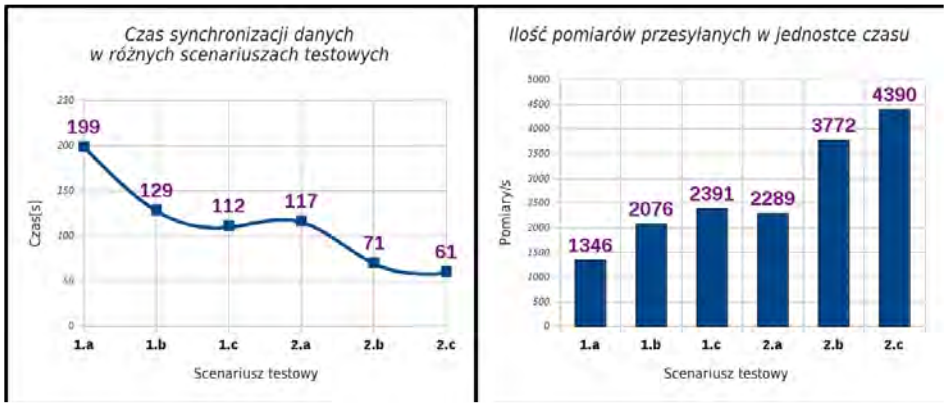
4. Testy

Aby zewaluować działanie systemu opracowano grupę scenariuszy testowych, których wykonanie przeprowadzono w zestawionym środowisku wewnątrz sieci *CERN*. Zdefiniowano 5 scenariuszy zakładających różny sposób konfiguracji i rozmieszczenia elementów systemu: bezpośrednie wywołania metod – odpowiadający monolitycznej aplikacji (1.a), broker rozmieszony poza maszyną wirtualną szyny *ESB* (1.b), broker rozmieszczony w tej samej maszynie wirtualnej co szyna (1.c), rozproszenie (2.a) oraz klasteryzacja (2.b). Scenariusze miały na celu ocenę różnych aspektów działania systemu, z których najważniejsze to efektywność oraz skalowalność. W każdym scenariuszu przesyłano ponad 260 tysięcy pomiarów różnego formatu i objętości.

[7] *http://www.springframework.org/*
[8] *http://www.hibernate.org/*

4.1. Efektywność

Wyniki testów przedstawione na Rysunku 3 ukazują wyższość asynchronicznego przetwarzania nad zsynchronizowanym sposobem pracy komponentów. Zysk wynika bezpośrednio z charakteru wykonywanych operacji, z których znakomita większość jest operacjami wejścia-wyjścia, oraz wykorzystania wolnych zasobów obliczeniowych poprzez mechanizmy współbieżności. W miarę zwielokrotnienia ilości jednostek obliczeniowych w łatwy sposób zapewnia się równoległość wykonania. Wartym uwagi jest fakt, że dzięki stosowaniu się do pryncypiów *SOA* [6] (m.in. bezstanowość i luźne powiązania usług) w celu rozproszenia aplikacji nie trzeba zmieniać implementacji rozwiązania - zmienia się jedynie konfiguracja usług.

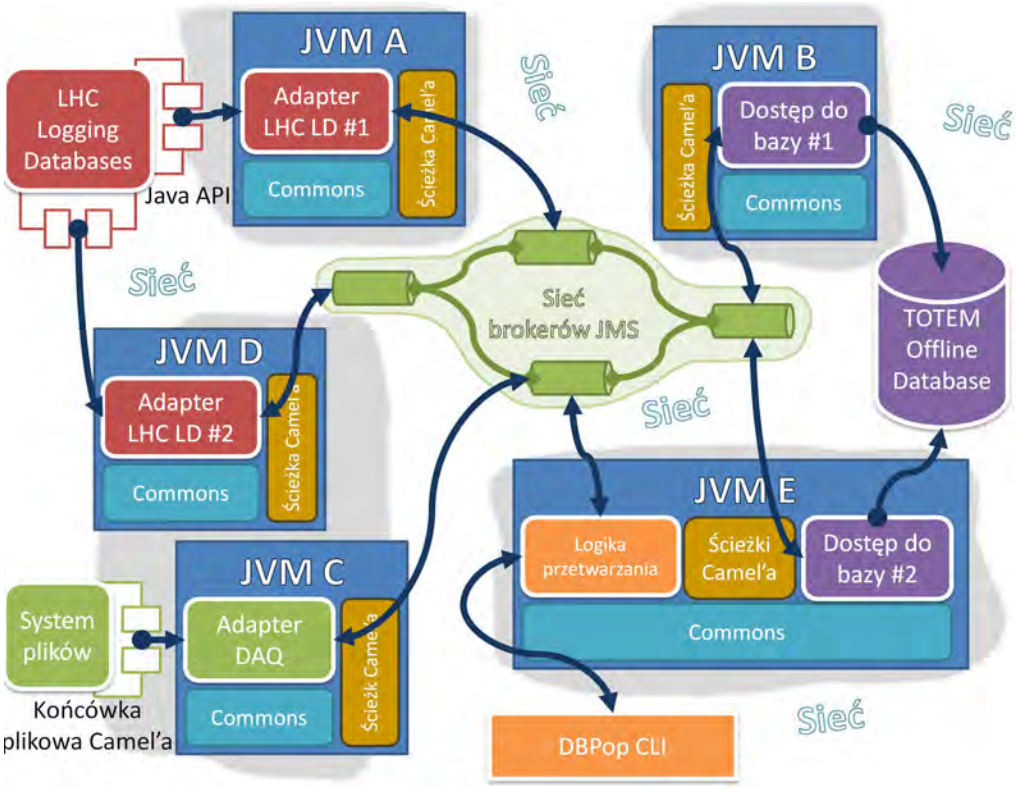


Rysunek 3. Wyniki testów przedstawiają czas przesyłu danych testowych oraz ilość transferowanych pomiarów w jednostce czasu dla różnych przypadków testowych

4.2. Skalowalność

W miarę rozwoju badań naukowców, popyt na dane będzie wzrastał. Nie można zakładać obciążenia na stałym poziomie, dlatego system powinien być gotowy na zwiększone wymagania. Moc obliczeniowa pojedynczego węzła w pewnym momencie będzie niewystarczająca, by sprostać zwiększonemu strumieniowi danych.

Bezstanowość i luźne powiązania elementów przesyłowych systemu *DBPop* (adaptery, kolejki) jest cechą, która pozwala na bardzo elastyczną kompozycję systemu. Przykładowe rozmieszczenie komponentów zostało ukazane na Rysunku 4. Szare obszary w głębi reprezentują maszyny fizyczne, na których rozmieszone zostały *maszyny wirtualne Javy* (ang. *JVM – Java Virtual Machine*) goszczące pracujące komponenty. Zrównoleglenie wykonania osiąga się przez zwielokrotnienie komponentów oraz zmodyfikowaną konfigurację trasowania. Kolejki z żądaniami dostarczania danych pełnią funkcję równoważenia obciążenia, dzięki czemu wszystkie elementy pracują z maksymalną prędkością. Strojenie systemu polega na optymalnym doborze ilości komponentów danego typu i odpowiednim ich rozmieszczeniu.



Rysunek 4. Scenariusz testowy 2.b ukazujący rozproszenie i klasteryzację komponentów DBPop

4.3. Ocena pozostałych cech rozwiązania

Oprócz najbardziej naturalnych wskaźników – efektywności i skalowalności należy zwrócić uwagę na inne właściwości zaimplementowanego systemu, takie jak: odporność na błędy czy łatwość rozbudowy. Podczas synchronizacji danych trwających nieraz godzinami może dojść do sytuacji wyjątkowych, takich jak: błędy transmisji, niedostępność systemu zewnętrznego czy awaria jednego z węzłów systemu *DBPop*. Dzięki mechanizmom alternatywnego połączenia (ang. failover) i ponownego dostarczenia (ang. redelivery) system jest w stanie samodzielnie powrócić do poprawnego działania. Rozbudowa systemu polega na skorzystaniu z wielu domyślnie dostępnych końcówek (ang. endpoint) udostępnianych przez system *Camel* lub zbudowanie własnego, bezstanowego i autonomicznego adaptera źródła danych i jego integracja z istniejącymi komponentami poprzez mechanizmy trasowania.

5. Podsumowanie i wnioski

Autor za najważniejsze osiągnięcie uważa opracowanie koncepcji infrastruktury programowej do efektywnej synchronizacji danych. Koncepcja ta powstała przez analizę środków jakich dostarcza podejście *Enterprise Service Bus* do konstrukcji oprogramowania. Wykorzystanie infrastruktury *Message-Oriented Middleware* okazuje się bardzo efektywnym i elastycznym środkiem transportu danych. Autor pokazuje wyższość asynchronicznego, nieblokującego przetwarzania nad tradycyjnymi blokującymi i synchronicznymi wywołaniami. Praca jest praktycznie wykorzystywana jako system dostarczania danych dla bazy eksperymentu *TOTEM* w *CERN*.

Innymi ważnymi osiągnięciami są: wykazanie prostoty modelu programowania współbieżnego w podejściu komponentowym oraz udowodnienie, że najnowsze rozwiązania ar-

chitektoniczne mogą służyć nie tylko w przemyśle informatycznym ale również w obszarze badań naukowych, jakimi są eksperymenty *Fizyki Wysokich Energii*.

System jest nadal rozwijany. Oprócz oczywistej rozbudowy w postaci obsługi kolejnych źródeł danych istnieją obszary potencjalnych dalszych usprawnień. W najbliższej przyszłości zamiarem autora jest integracja systemu ekstrakcji danych z bazy ze stworzonym systemem zaopatrywania bazy w dane oraz zastąpienie wszelkich bezpośrednich odwołań do bazy przez zunifikowaną warstwę dostępową.

Bibliografia

[1] L.R. Evans. *The Large Hadron Collider: a marvel of technology*. Lozanna, Szwajcaria, Fundamental sciences, EPFL Press, 2009

[2] Antchev G. et al, *The TOTEM front end driver, its components and applications in the TOTEM experiment.* Topical Workshop on Electronics for Particle Physics, Praga, Czechy, 2007, s.211-214

[3] The TOTEM Collaboration at al: *The TOTEM Experiment at the CERN Large Hadron Collider*, Journal of Instrumentation 3 s. 8007, 2008

[4] Wyszkowski P.: *ESB application for effective synchronization of large volume measurements data*. LAP Lambert Academic Publishing, Saarbrücken, Niemcy, 2012

[5] IBM Software Group, Flurry G.: *Exploring the Enterprise Service Bus, Part 1: Discover how an ESB can help you meet the requirements for your SOA solution*. 2007. url: https://www.ibm.com/developerworks/webservices/library/ar-esbpat1/

[6] SOA Systems Inc: *SOA Principles*. 2007-2009. *http://www.soaprinciples.com/*

Wdrażanie kursu „Zespołowe przedsięwzięcie inżynierskie" – doświadczenia, ocena, ulepszanie

Bogumiła Hnatkowska, Iwona Dubielewicz, Lech Tuzinkiewicz

Politechnika Wrocławska, Wydział Informatyki i Zarządzania

Abstract:

A new course, Engineering Team Project (ETP) was offered for the first time to student two years ago at Computer Science and Management Faculty, Wroclaw University of Technology. Any teaching process being a service offered to students should be evaluated and improved timely. The main aim of the paper is to present the results of the evaluation of the implemented two editions of the ETP course with respect to the defined expectation for it. The paper provides a brief overview of existing definition and organization of this course. Afterwards the course assessments carried out by the students and teachers are presented. Some corrective actions (e.g. the organization of other student courses as well as the ETP course itself) resulting from the ETP course evaluation has been undertaken.

Keywords:

student team project, course evaluation, education process improvement,

1. Wprowadzenie

Wydział Informatyki i Zarządzania Politechniki Wrocławskiej w roku akademickim 2010/11 zrealizował po raz pierwszy obowiązkowy kurs pt. „Zespołowe przedsięwzięcie inżynierskie (ZPI)" dla studentów VII/VIII semestru studiów inżynierskich stacjonarnych/niestacjonarnych. Jego pozytywne ukończenie stanowi warunek konieczny otrzymania przez studenta tytułu inżyniera.

Kurs ZPI to usługa, oferowana przez zespól pracowników, dla studentów Wydziału IZ. Na cykl życia tej usługi składają się (do tej pory) dwie fazy:

- faza inicjowania i opracowania (ang. inception and development phases), związane z przygotowaniem kursu do realizacji, oraz
- faza świadczenia usługi (ang. operation phase), polegająca na realizacji kursu ZPI w kolejnych edycjach (co rok).

Głównym celem artykułu jest przedstawienie doświadczeń płynących z realizacji kursu ZPI, wyników ocen kursu z dwóch lat oraz podjętych i planowanych akcji naprawczych. Pośrednim celem artykułu jest odpowiedź na pytanie, jak przygotować studentów do pracy zespołowej.

Celem dodatkowym artykułu jest wskazanie najczęściej występujących błędów w realizacji studenckiego przedsięwzięcia informatycznego. Błędy te zostały zauważone przez oceniających i stąd pojawił się pomysł (oraz próba) określenia tzw. modelu referencyjnego typowego projektu, który ma stanowić punkt odniesienia zarówno dla prowadzących przedsięwzięcia jak i studentów, w stwierdzeniu, czy zakres planowanych/wykonanych prac nie jest zbyt wąski lub zbyt szeroki.

Zebrane doświadczenia powinny być pomocne podczas organizacji podobnych kursów i mogą służyć do celów porównawczych.

W pierwszej części artykułu przedstawiono krótko fazę inicjowania i opracowania usługi „Kurs ZPI" (opisano je bardziej szczegółowo w [6, 7]). W drugiej części omówiono realizację dwóch edycji kursu. Część trzecia, zasadnicza, omawia metodę ewaluacji edycji kursu. Prezentowane są oceny i uwagi zebrane od studentów i pracowników, jak również omówione są typowe błędy w planowaniu, popełniane przez studentów. W podsumowaniu podano wnioski dotyczące przyszłych realizacji kursu ZPI.

2. Faza inicjowania i opracowania

2.1. Inicjowanie przedsięwzięcia

Celem kursu ZPI jest zespołowe wykonanie zdefiniowanego produktu w warunkach jak najbardziej zbliżonych do rzeczywistych. Kurs ZPI trwa 10 tygodni. Jest realizowany od dwóch lat dla dużej i zmiennej liczby studentów, przez coraz większą liczbę prowadzących. Praca zespołu jest wspomagana narzędziem Microsoft TFS 2010 [11] zainstalowanym w laboratoriach Instytutu, do którego studenci posiadają autoryzowany dostęp zarówno w trakcie zajęć laboratoryjnych, jak również poza Uczelnią. Praca zespołu studenckiego odbywa się w ściśle określonych terminach w laboratoriach Instytutu, ale większą część zadań studenci wykonują bez nadzoru prowadzących, zgodnie z przyjętym planem i zasadami współpracy (komunikacji), które są ustalone indywidualnie przez poszczególne zespoły.

Kurs jest realizowany zgodnie z następującymi założeniami. Zespoły studentów liczą 4 osoby, w wyjątkowych sytuacjach 3 lub 5. Studenci sami organizują się w zespoły, wybierając do realizacji proponowane tematy. Pomysłodawcami tematów projektów• są przede wszystkim pracownicy Instytutu Informatyki Politechniki Wrocławskiej, ale dopuszcza się również tematy proponowane przez studentów, którzy z wybranym przez siebie prowadzącym uzgodnią możliwość ich realizacji (w kolejnych edycjach zauważa się wzrost liczby tych tematów)•. Kategorie produktów wytwarzanych w ramach kursu ZPI są różne; w większości (ok. 90%) projektów wytwarzanymi produktami są aplikacje programowe zrealizowane w wybranym środowisku i języku programowania, lecz produktami mogą być również urządzenia czy instalacja oprogramowania sieciowego (serwery pracujące w sieci i ich konfiguracja). Każdy członek zespołu jest zobowiązany do przepracowania pewnej, ustalonej liczby godzin tygodniowo nad projektem.

Ze względu na dużą skalę przedsięwzięcia do zorganizowania kursu oraz przygotowania prowadzących, powołano specjalną komisję. Komisja dokonała wyboru środowiska informatycznego wspierającego pracę zespołową i przygotowała materiały szkoleniowe zarówno dla prowadzących zajęcia jak i studentów. Ponadto zorganizowano seminarium, na którym pracownicy dydaktyczni zostali przeszkoleni w zakresie prowadzenia kursu z wykorzystaniem środowiska TFS.

2.2. Definicja przedsięwzięcia

Efektem końcowym przedsięwzięcia ZPI powinien być produkt (zwykle programowy) wykonywany zgodnie z modelem iteracyjno-przyrostowym cyklu życia produktu [9]. Definicja procesu wytwarzania produktu nie jest określona, natomiast został zaproponowany proces (metodyka) zarządzania przedsięwzięciem – metodyka Scrum [10, 12]. Dodatkowo zaproponowano zbiór dobrych praktyk [6, 7], które wg opinii Komisji powinny wspomóc zespoły w terminowym osiągnięciu celu (tzn. wytworzeniu produktu).

Zaproponowano, by w planowaniu globalnym (okres semestru) przedsięwzięcia były realizowane od 1 do 3 wydań produktu. W ramach każdego wydania definiowane są stałookresowe iteracje (od 2 do 9).

Na początku każdej iteracji studenci powinni zaplanować (wybrać) wymagania na produkt (fragment zamówienia produktowego, ang. Product Backlog), które zamierzają zrealizować w czasie iteracji. Ponadto, w trakcie planowania, określane są zadania (będące realizacją wymagań), wybierane przez członków zespołu do wykonania.

Wymaga się, by, zgodnie z metodyką Scrum, po zakończeniu iteracji odbywała się ocena uzyskanego produktu i podejmowane są decyzje dotyczące zakresu dalszych prac.

Pozostawienie swobody wyboru co do liczby wydań oraz iteracji wynika głównie z różnych kategorii wykonywanych produktów; wielkości te są ustalane przez prowadzącego zespół, przy współudziale studentów.

Założono, że proces realizacji produktu plasuje się na 3-cim poziomie dojrzałości wg [3] to znaczy, iż proces jest zdefiniowany i udokumentowany. Możliwe jest też jego ulepszanie na podstawie zebranych doświadczeń.

3. Faza świadczenia usługi „kurs ZPI"

Kurs ZPI w formie 10-tygodniowej był realizowany dwukrotnie:

- 1-sza edycja (2010/2011) przez ponad 140 studentów podzielonych na 35 zespołów i 15 prowadzących;
- 2-ga edycja (2011/2012) przez 175 studentów, 44 zespoły i 17 prowadzących.

W pierwszej edycji kursu każdy członek zespołu był zobowiązany do przepracowania 40 godzin tygodniowo, a w drugiej – 20-25 godzin, na rzecz ZPI.

Wytwarzanie produktu było prowadzone podczas spotkań zespołu studentów w Instytucie (1-3 razy w tygodniu) i w innych lokalizacjach. Zajęcia (spotkania) w Instytucie, które odbywały się po skończeniu iteracji, miały jednolity format: prezentacja produktu prowadzącemu, ocena wykonanego produktu i planowanie kolejnej iteracji. Inne spotkania miały charakter czysto roboczy, a ich organizacja i przeprowadzenie jest pozostawione zespołowi.

Wykorzystywanie narzędzia MS TFS pozwoliło na rejestrację i specyfikację wymagań na projekt (dla aplikacji programowych umożliwia się więc stosowanie praktyk Product Backlog i Sprint Backlog [10, 12]), a także monitorowanie pracy zespołu i śledzenie postępu projektu (przez stosowanie diagramu wypalania Scrum [10, 12]).

Dodatkowo narzędzie to jest wykorzystywane (przez część zespołów) jako repozytorium artefaktów projektu.

4. Ewaluacja realizowanej usługi „kurs ZPI"

Po pierwszej edycji kursu dokonano jego oceny zarówno ilościowej, jak i jakościowej [8] oraz podjęto działania, mające na celu ulepszenie jego realizacji (polepszając tym samym proces dydaktyczny). Po drugiej edycji kursu ponownie przeprowadzono ocenę kursu (ilościową i jakościową) z perspektywy studentów• i pracowników.

Oceny kursu dokonano na podstawie ankiety, która zawierała następującą listę pytań:

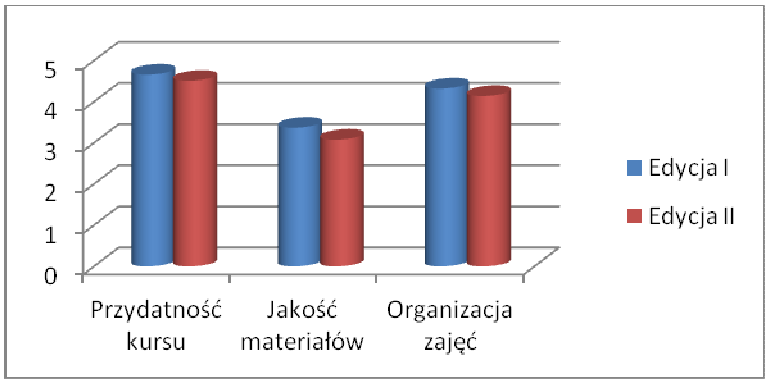
- Oceń poniższe elementy w skali 1-5 (1 – ocena najniższa, 5 – ocena najwyższa):
  - Przydatność kursu
  - Jakość materiałów pomocniczych
  - Organizacja zajęć
- Które elementy były najbardziej wartościowe
- Które elementy należy poprawić/zmienić w następnych edycjach

Ankieta była anonimowa, wypełniana głównie drogą elektroniczną. Oceny pracowników nie różniły się znacząco od ocen studenckich, dlatego w artykule zostaną pominięte.

4.1. Ewaluacja ilościowa

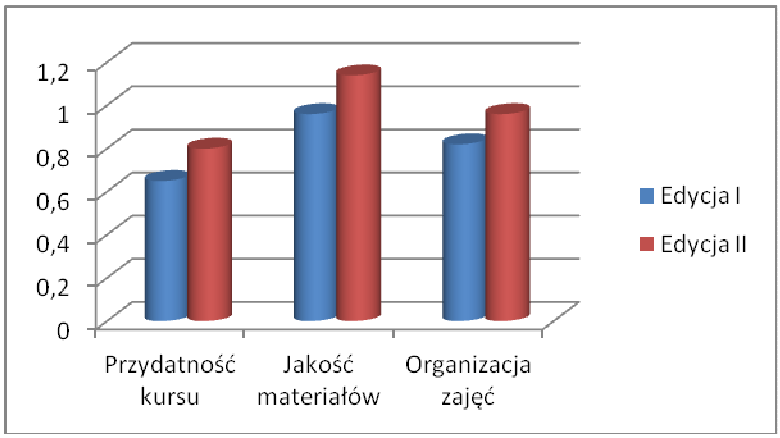
Na rys. 1. zaprezentowano średnie oceny wystawione przez studentów za przydatność kursu, jakość materiałów pomocniczych i organizację zajęć w dwóch kolejnych latach, przy czym liczba respondentów w 2010 wynosiła 93/145 (64%), a w 2011 - 85/175 (49%).

Oceny w poszczególnych kategoriach uzyskane w kolejnych latach są do siebie zbliżone, choć wszędzie zaznaczyła się tendencja spadkowa. Najlepiej oceniono przydatność kursu i organizację zajęć, najgorzej – jakość materiałów pomocniczych. Jest to konsekwencją niezbyt rozwiniętej akcji informacyjnej wśród studentów na temat kursu i dostępnych zasobów. Część studentów drugiej edycji nawet nie wiedziała, że istnieją takie materiały i gdzie ich szukać.



Rysunek 1. Średnie ocen kursu wystawionych przez studentów

Oceny wystawiane przez studentów po drugiej edycji były bardziej zróżnicowane, co ilustruje rys. 2., pokazujący ich odchylenie standardowe. Może to świadczyć m.in. o różnym poziomie wiedzy i zaangażowania prowadzących.



Rysunek 2. Odchylenie standardowe ocen kursu wystawionych przez studentów

4.2. Ewaluacja jakościowa

W ramach pytań jakościowych pytano studentów o elementy, które – z ich perspektywy – były najbardziej wartościowe (warte powtórzenia) i o elementy, które należy ulepszyć, zmienić. Odpowiedzi na powyższe pytania otwarte zostały skrupulatnie przeanalizowane.

Do elementów najbardziej wartościowych zaliczono (w kolejności najczęściej występujących odpowiedzi):

- Edycja 2010:
  - Możliwość pracy w zespole
  - Stosowana metodyka (lekka)
  - Nowoczesne narzędzia
  - Możliwość zrealizowania projektu od początku do końca
  - Regularne spotkania i planowanie
  - Interesujące tematy
- Edycja 2011:
  - Możliwość pracy w zespole
  - Stosowana metodyka (lekka)
  - Regularne spotkania i planowanie
  - Nowoczesne narzędzia
  - Duża samodzielność zespołów

Jak widać, studenci konsekwentnie doceniali możliwość zebrania doświadczeń w ramach pracy zespołowej, praktyczne zastosowanie metodyki wykorzystywanej w firmach IT, zapoznanie się z nowymi (również komercyjnymi) narzędziami do planowania i realizacji przedsięwzięć (np. systemy wersjonowania kodu). Regularne spotkania i planowanie w małej skali pozwoliło im ukończyć projekt w bardzo krótkim terminie.

Wśród elementów wymagających poprawy wymieniono:

- Edycja 2010:
  - Za mało czasu na realizację projektu
  - Zbyt późno realizowany kurs z zarządzania projektem i brak szkolenia z TFS
  - Zbyt skomplikowane narzędzie do planowania (TFS)
  - Brak jednolitego systemu oceniania projektów
- Edycja 2011:
  - Zbyt skomplikowane narzędzie do planowania (TFS)
  - Za mało czasu na realizację projektu
  - Zbyt intensywny, za krótki semestr
  - Niejednolite wymagania dokumentacyjne w różnych zespołach
  - Zbyt mało swobody studentów w zakresie definiowania tematów do realizacji

4.3. Podjęte i planowane akcje naprawcze

Po pierwszej edycji kursu w 2010 r. wnikliwie przeanalizowano głosy studentów, na temat realizacji kursu. W celu poprawy kolejnej edycji: przeniesiono kurs „Zarządzanie projektem" na wcześniejszy semestr (w trakcie pierwszej edycji był realizowany równolegle do ZPI), zmniejszono obciążenia studenta z 40h/tydzień do 20h/tydzień (w tym samym semestrze studenci mają również inne kursy); przygotowano i zrealizowano w kolejnej edycji konkurs (z nagrodami finansowymi) na najlepszy projekt; nominacja do konkursu pozwalała prowadzącemu wystawić oceny celujące.

W tym samym roku dokonano również analizy rejestrowanych danych w systemie TFS, dotyczących m.in. liczby wymagań na produkt (liczba wpisów w Product Backlog), liczby zadań przypadających na iterację i czasu poświęcanego przez studenta na realizację projek-

tu. Zaobserwowano znaczne rozbieżności pomiędzy zespołami, np. liczba PBI: minimum 4 (zespół prawdopodobnie zarzucił używanie narzędzia), maksimum 89, średnia 23; liczba zadań zaplanowanych w trakcie przedsięwzięcia: minimum 32, maksimum 313, średnia 128. Obserwacje potwierdziły głosy studentów dotyczące ich nadmiernego obciążenia (średnio zespoły wypracowały 77% planowanego obciążenia) i wykazały, iż brakuje punktu odniesienia, pozwalającego prowadzić projekty o porównywalnej złożoności. Stąd pomysł, aby po kolejnym roku doświadczeń zbudować model referencyjny przedsięwzięcia informatycznego.

Podobnie postąpiono po edycji kursu w 2011 r. W większości akcje naprawcze dotyczyły ujednolicenia wymagań dotyczących dokumentacji projektu i sposobu jego rozliczenia (prezentacja końcowa). Prowadzący ZPI (w tym również osoby dopiero do tej roli wdrażane) uzgodnili (poprzez głosowanie) zakres dokumentacji dla projektów programowych:

- Dokument Wizji.
- Historie użytkowników wzbogacone o scenariusze tekstowe i/lub prototypy interfejsu użytkownika.
- Opis architektury (co najmniej dwa widoki: logiczny i fizyczny) i zastosowanych wzorców.
- Dokumentacja dla użytkownika (manual) lub interaktywny system pomocy.
- Dokumentacja dla administratora.
- Prezentacja końcowa.

W uzasadnionych przypadkach możliwe jest rozszerzenie o model danych i projekt bazy danych.

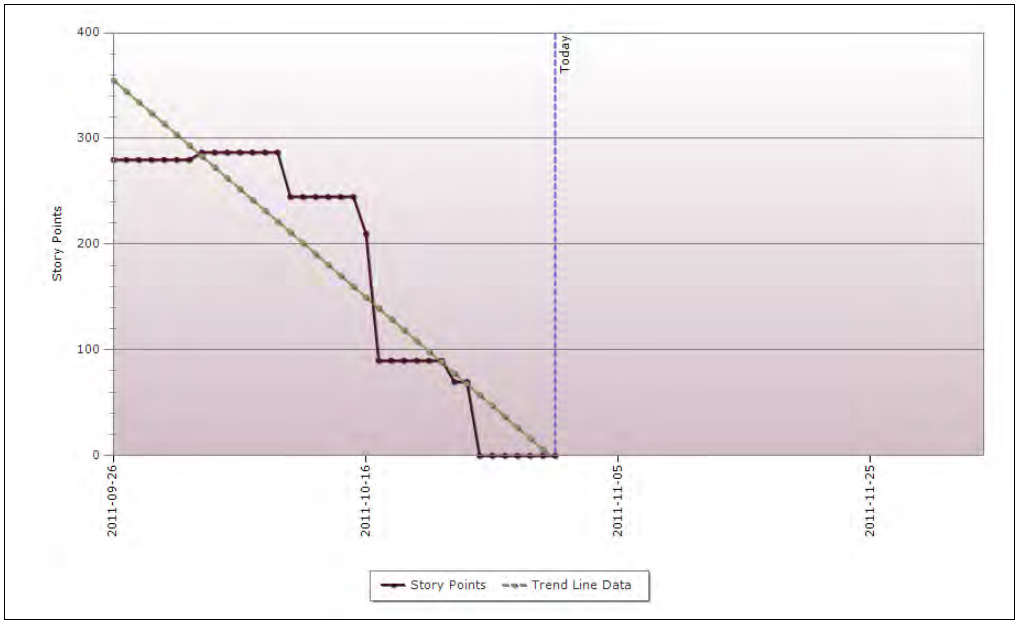
Dodatkowo zaplanowano kurs TFS w ramach zajęć z Zarządzania projektem (w tym roku realizowany przez Volvo), opracowanie krótkiej instrukcji dla typowych zadań w TFS (chwilowo prace wstrzymano ze względu na planowaną zmianę wersji programu), oraz dodatkowe szkolenie dla chętnych pracowników z zakresu metodyki i narzędzi.

Ustalono również wytyczne dla tzw. model referencyjnego typowego przedsięwzięcia informatycznego (projekt softwareowy). Wyrażono go w terminach podstawowych, mierzalnych wartości, pozostawiając pewną swobodę realizacyjną:

- Liczba elementów PBI (Product Backlog Item): 20 ± 5,
- Łączna liczba zadań wykonanych przez 1 osobę: 40-80 (maksymalny czas trwania zadania wynosi 8h, preferowane są zadania o długości 4h),
- Łączna liczba godzin pracy 1 osoby: 200-250.

5. Typowe błędy studentów

Do monitorowania postępów projektu względem planów w metodyce Scrum służą diagramy wypalania. Diagramy te są budowane zarówno dla poszczególnych iteracji (sprintów), jak i dla wydań. Przykładowy, poprawny (oczekiwany) diagram wypalania dla jednej iteracji jest przedstawiony na rys. 3. Oś X reprezentuje kolejne dni trwania sprintu, natomiast oś Y – nakład pracy związany z realizacją zadań.



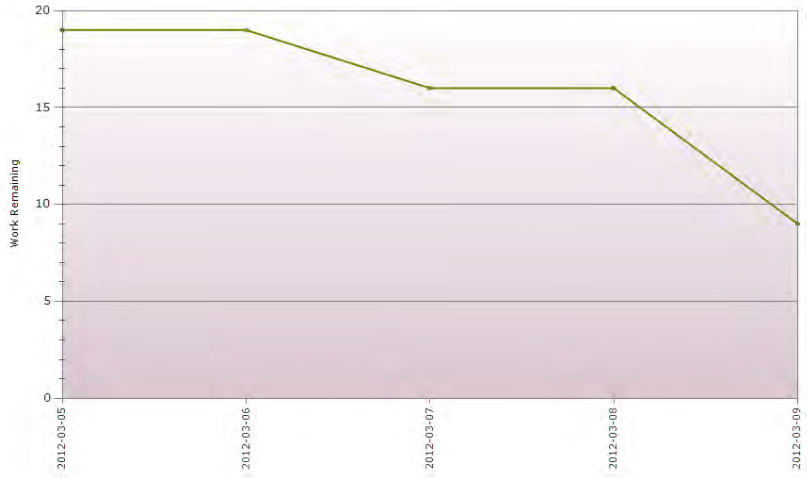
Rysunek 3. Przykładowy diagram wypalania dla iteracji

Przerywana linia reprezentuje planowaną ilość pracy do wykonania (linia trendu), natomiast linia ciągła – rzeczywiście wykonaną ilość pracy. W startowym punkcie diagramu punkt na osi Y określa nakład pracy związany z wykonaniem zaplanowanych zadań wymaganych do realizacji zamówienia sprintu oszacowany w tzw. punktach funkcyjnych (*story points*). Zakłada się, że na koniec iteracji wszystkie zadania zostaną wykonane tzn. wymagany nakład pracy jest równy zero – punkt końcowy linii trendu pokrywa się z linią realizacji zadań. Linia realizacji zadań poniżej linii trendu wskazuje na wyprzedzenie, a powyżej, na opóźnienie w stosunku do planu realizacji zadań sprintu.

W praktyce diagramy wypalania mogą znacznie różnić się od przedstawionego diagramu na rys. 3. Doświadczenie zespołu wykonawczego (umiejętność planowania i oceny możliwości wykonawczych), znajomość i dyscyplina w przestrzeganiu zasad postępowania zgodnie z metodyką Scrum mają ogromny wpływ na realizację planu sprintu. Dla zilustrowania takich przypadków zostaną przedstawione rzeczywiste przykłady realizacji sprintu przez zespoły, które po raz pierwszy realizują przedsięwzięcie informatyczne zgodnie z metodyką Scrum.

Diagram wypalania przedstawiony na rysunku 4. jest ilustracją sytuacji, którą można zinterpretować w następujący sposób:

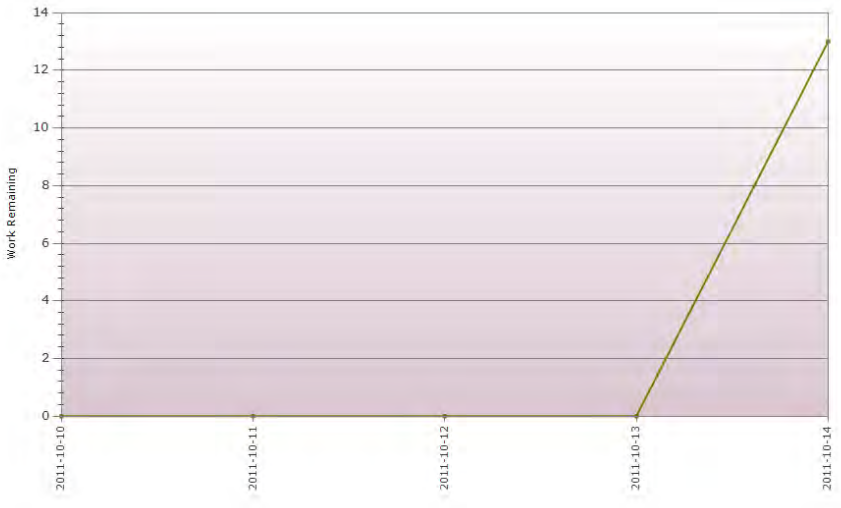
1. Pracochłonność zadań została niepoprawnie oszacowana przez zespół (niedoszacowanie).
2. Rozpoczęte zadania nie były zakończone w planowanym terminie (członkowie zespołu mogli być zaangażowani w realizację innych przedsięwzięć lub wystąpiły przeszkody uniemożliwiające realizację planowanych zadań – wymagana dodatkowa analiza przyczyn opóźnienia) lub nie odnotowano w narzędziu zakończenia realizacji zadań, mimo, iż faktycznie je wykonano.



Rysunek 4. Diagram wypalania ilustrujący brak osiągnięcia celu sprintu

Diagram wypalania przedstawiony na rysunku 5. wskazuje, że zespół ewidentnie nie przestrzegał zasad wynikających z metodyki Scrum. W szczególności może to dotyczyć następujących przypadków:

1. Zespół wykonawców nie zaplanował zadań na początku sprintu (planowanie sprintu w trakcie jego realizacji).
2. Zaplanowane zadania nie zostały rozpoczęte zgodnie zaplanowanym terminem.
3. Zaplanowane zadania były przypisywane do sprintu w trakcie jego realizacji (przyrost nakładu pracy).

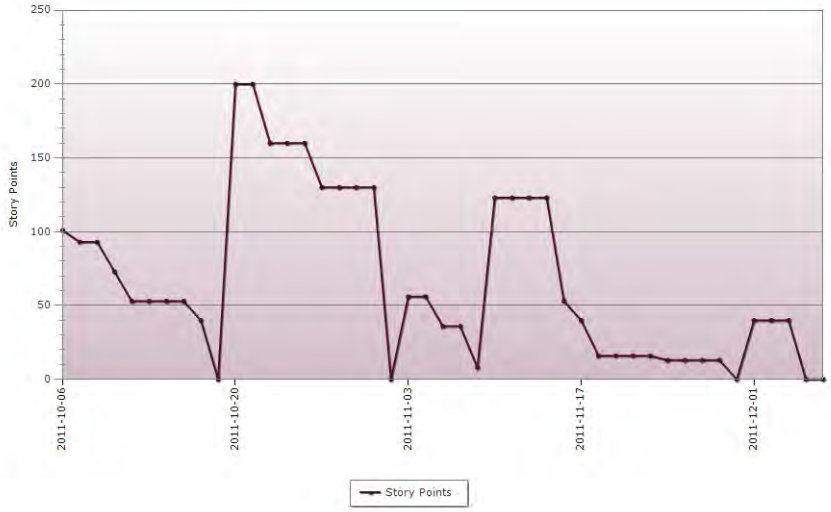


Rysunek 5. Diagram wypalania ilustrujący postępowanie zespołu niezgodne z metodyką Scrum

Ostatni diagram wypalania (rys. 6.) jest ilustracją realizacji zadań z okresu kilku sprintów (dotyczacy wydania produktu). Warto zwrócić uwagę na to, że planowane czasochłonności zadań, przeznaczonych do realizacji w kolejnych sprintach, nie są jednakowe, co może wskazywać na:

1. Niewłaściwą ocenę wydajności zespólu wykonawców
2. Sytuacje losowe w zespole np. choroba

3. Ingerencję osób trzecich w pracę zespołu (niedopuszczelne w metodyce Scrum)
4. Brak umiejętności planowania sprintów



Rysunek 5. Diagram wypalania dla wydania produktu

6. Wnioski końcowe

Zespołowa realizacja przedsięwzięcia informatycznego ma miejsce na wielu uniwersytetach, np. [2, 4, 5, 13]. Istnieją jednak różnice pomiędzy ich realizacją, a realizacją na naszej uczelni:

- Efektem kursu ZPI jest działające wydanie systemu/usługi, często z wersją instalacyjną i podręcznikiem użytkownika, podczas gdy na innych uczelniach często powstaje tylko prototyp systemu,
- Kurs ZPI jest kursem praktycznym, na którym studenci mogą wykorzystać wiedzę z innych kursów, dodatkowo zdobywając doświadczenia w pracy zespołowej; na innych uczelniach często nacisk jest kładziony na cele edukacyjne (tzn. pokazuje się jak wykonywać kolejne etapy wytwarzania oprogramowania oraz jak je dokumentować).

Realizacja dwóch edycji kursu ZPI uświadomiła nam problemy, z którymi prawdopodobnie będziemy spotykać się w kolejnych edycjach kursu. Przede wszystkim należy rozwiązać problem skalowalności zaproponowanej organizacji kursu. Rosnąca z roku na rok liczba studentów (140, 170, 250) skutkuje większą liczbą zespołów, co z kolei wymaga przygotowania coraz większych zasobów tzn. pracowników prowadzących kurs, tematów do realizacji, laboratoriów. Efektem tej sytuacji jest trudność w:

- Wytypowaniu odpowiednio przygotowanych pracowników (będących np. na bieżąco z nowymi technologiami wytwórczymi jak i metodykami zarządczymi),
- Sformułowaniu jednorodnych kryteriów oceny:
  - złożoności tematów,
  - jakości wytworzonych produktów (w celu porównania),
  - jakości realizowanego procesu.

Wydaje się, że dobrym rozwiązaniem byłoby pozyskanie większej liczby tematów z firm zewnętrznych wraz z opiekunami.

Innym problemem, trudniejszym do rozwiązania, jest stworzenie zespołu z 4-osobowej grupy studenckiej w bardzo krótkim czasie (2-3 tygodnie). Procesy formowania i normowania się zespołu są pozostawione studentom, co przy braku studenta-lidera w zespole może

trwać długo. Co więcej, przeprowadzone przez nas badania ankietowe dotyczące predyspozycji studentów do pełnienia ról w zespole wg klasyfikacji Belbina [1] wykazały brak odpowiedniej liczby studentów o osobowości koordynator (ang. coordinator). W kolejnych dwóch rocznikach tylko ok. 4% populacji studentów wykazywało takie predyspozycje, podczas gdy powinno ich być ok. 25%.

W efekcie analizy uzyskanych ocen kursu zaproponowano ulepszania procesu jego realizacji m.in. poprzez:

1. Szkolenie studentów (w celu poznania środowiska pracy zespołowej) w semestrze poprzedzającym kurs ZPI.
2. Realizację kursu „Elementy Zarządzania Projektami informatycznymi" w semestrze poprzedzającym kurs ZPI.
3. Zaproponowanie tzw. modelu referencyjnego projektu zespołowego, który ma pomóc prowadzącym w określeniu realizowalnego zakresu dla przedsięwzięcia.
4. Wprowadzenie standardów dokumentacyjnych dla przedsięwzięcia.

Bibliografia

[1] Belbin M., *Belbin Team Role Theory* [online], http://www.belbin.com/ [dostęp: 2012]
[2] Bielikova M., Navrat P. *Experiences with Designing a Team Project Module for Teaching Teamwork to Students*, Journal of Computing and Information Technology, Vol. 13, No 1, 2005, http://cit.srce.hr/index.php/CIT/article/view/1558
[3] Collofello J. S., Ng Ch. H. *Assessing the Process Maturity Utilized in Software Engineering Team Project Courses*. Frontiers in Education Conference, Vol. 1, 12A9/5 – 12A9/9, Puerto Rico, 1999
[4] Delaney D., Mitchell G. *Tutoring project-based learning: a case study of a third year software engineering module at NUI* [online] http://www.aishe.org/readings/2005-2/contents.html [dostęp 2012]
[5] Dubielewicz I. , Hnatkowska B. *Improving software development process implemented in team project course.* In Proc. of the 8th International Conference on Computational Science, Part II, Berlin, 2008. Springer-Verlag, s. 687-696
[6] Dubielewicz, I., Hnatkowska, B. *Praktyki w inżynierii oprogramowania – perspektywa pracy zespołowej*. Inżynieria oprogramowania w procesach integracji systemów informatycznych, Górski J., Orłowski C. (eds), Pomorskie Wydawnictwo Naukowo-Techniczne PWNT, Gdańsk, 2010, s. 121-128.
[7] Dubielewicz I,. Hnatkowska, B. *Best practices in students team projects*. Information systems architecture and technology: IT models in management process, Wilimowska Z. (eds.), Oficyna Wydawnicza Politechniki Wrocławskiej,Wrocław, 2010, s. 487-497
[8] Hnatkowska B., *Software Engineering Team Project – Lessons Learned*, Proceedings of IFIP TC2 CEE-SET 2011 conference (w przygotowaniu).
[9] ISO/IEC 12207:2008, Information technology – Software life cycle processes.
[10] Levin R. *Understanding Scrum – Best Practices Guide* [online] http://www.brighthub.com/office/project-management/articles/68791.aspx [dostęp: 2012]
[11] Microsoft Corporation. *Team Development with Visual Studio Team Foundation Server*, Microsoft Press, 2007, s. 153-156
[12] Schwaber K. *Agile Software Development with Scrum*, Micosoft Press, 2004
[13] Zaigham M., *A Framework for Software Engineering Education: A Group Projects Approach*, International Journal of Education and Information Technologies, Issue 3, Vo. 1, 2007

Stanowiska językowe do budowy serwisów informacyjnych

Włodzimierz Dąbrowski[1], Agata Kosior[2], Andrzej Stasiak[2]

[1]*Politechnika Warszawska, Instytut Sterowania i Elektroniki Przemysłowej*
[2]*Wojskowa Akademia Techniczna, Wydział Cybernetyki*

Abstract:

This article describes a method of software development based on the concept of language workbenches. This method was used and verified in the process of building an information service about student projects. The proposed method defines three internal domain specific languages and determined how to use them in the process of the construction site of the modeling requirements for a system to automatically generate the service source code as a result of the transformation of the models.

Keywords:

Language Workbenches, Domain Specific Language, Domain Specific Modeling, UML Profiles

1. Wprowadzenie

Artykuł dotyczy zastosowania stanowisk językowych do budowy serwisów informacyjnych bazując na autorskiej metodzie KMS. Koncepcja stanowisk językowych[1] (ang. Language Workbenches) została zaproponowana przez Flowers'a [12] i wykorzystuje język UML jako platformę integracji piktograficznych języków dziedzinowych - DSL (ang. Domain Specific Language) [8][11][12][14] jako środka do budowy serwisów informacyjnych.

Ważnym aspektem wytwarzania systemów informatycznych z wykorzystaniem stanowisk językowych jest możliwość łączenia wielu dialektów, jako wewnętrznych języków dziedzinowych (Internal DSLs) i tworzenia w ten sposób kompletnych stanowisk językowych opisywanych przez zewnętrzne języki dziedzinowe (External DSLs) dla specjalizowanych dziedzin biznesowych. W [17] zbudowano kompletne stanowisko, na które złożyły się trzy dialekty (w KMS określane jako wewnętrzne DSLs): $SJ = \sum_{i=1}^{3} DSL_i$, gdzie kolejne DSL_i oznaczają odpowiednio: język opisu serwisu, projektu, komunikacji.

Mimo, że obecnie powszechnie stosowanym językiem modelowania ogólnego przeznaczenia jest UML [1][6][25], będący standardem w inżynierii oprogramowania, to fakt wykorzystywania go przez różne grupy zawodowe oraz stały wzrost złożoności narzędzi informatycznych wspierających różne dziedziny biznesowe coraz częściej uzasadnia potrzebę wykorzystywania języków dziedzinowych – DSL.

[1] Stanowisko językowe w naszym rozumieniu to przestrzeń, w której stosowany będzie dany język wraz z narzędziami językowymi, aby za jego pomocą skutecznie opisać ustalony dla tego stanowiska wycinek rzeczywistości. Warto tu zauważyć, że podobnie jak w fabryce zwykle do budowy systemu wymaganych jest wiele stanowisk pracy, podobnie i u nas, do celowego opisu systemu wymaganych jest wiele stanowisk językowych.

Kolejną inspirację dla opracowania własnej metody budowy serwisów informacyjnych KMS (obok stanowisk językowych) stanowił proces wytwarzania oprogramowania bazujący na modelowaniu dziedzin DSM (ang. Domain Specific Modeling) [14][15], zgodnie z którym ekspert dziedzinowy musi tylko określić pojęcia i specyficzne konstrukcje oraz zasady obowiązujące w dziedzinie, a środowisko DSM udostępni mu narzędzia do budowy: metamodeli [7][23], profili [9][10], edytorów piktograficznych [14], modeli docelowej dziedziny i ich transformacji [1][2][13][21].

Modelowanie dziedziny odwołuje się także do koncepcji generacji wykonywalnego kodu źródłowego (znanych z MDA- ang. Model Driven Architecture [22] i MDE- ang. Model Driven Engineering [26]), który budowany jest automatycznie – w wyniku transformacji opracowanych modeli zapisanych w języku dziedzinowym. Wolny od „ręcznego" tworzenia i utrzymywania kodu źródłowego proces jego wytwarzania może znacznie poprawić wydajność pracy programistów. Dodatkowo niezawodność automatycznego generowania w porównaniu do ręcznego kodowania pozwoli również na zmniejszenie liczby defektów w programach, a tym samym poprawę jakości budowanego oprogramowania. Języki dziedzinowe są wyspecjalizowane, ukierunkowane na konkretne zadania i problemy, przez co mogą być składniowo proste i posiadają ograniczoną do wąskiej domeny semantykę (ograniczony i zamknięty zbiór pojęć).

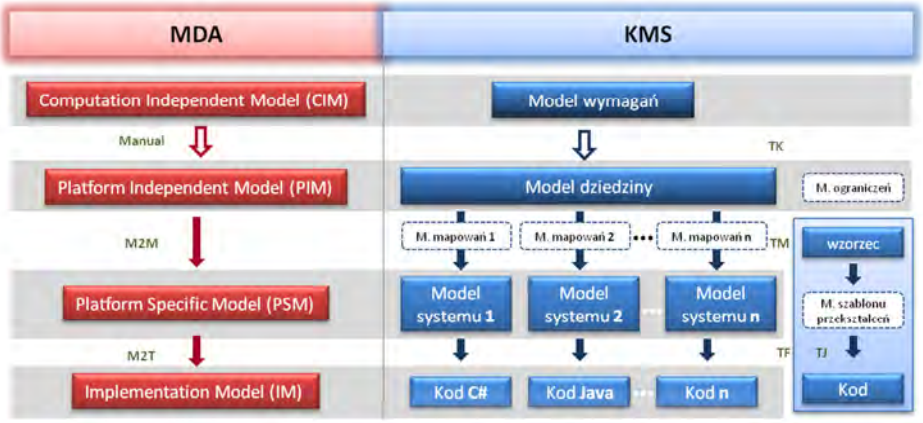
Przeciwieństwem języków dziedzinowych są języki programowania ogólnego, takie jak C# czy Java. DSL projektuje się, by ułatwić opisywanie zagadnień pewnej dobrze określonej dziedziny oraz skrócić proces wytwórczy oprogramowania systemu informatycznego. Języki dziedzinowe pośredniczą pomiędzy użytkownikiem (zamawiającym), a stroną wdrażającą (wykonawcą) i przyczyniają się do zwiększania użyteczności wytwarzanego oprogramowania dzięki temu, że pozwalają na precyzyjniejsze specyfikowanie wymagań (semantyka DSL jest wprost zaczerpnięta ze słownika pojęć dziedziny, którą on opisuje). Zaletą wytwarzania oprogramowania opartego o koncepcję budowy stanowisk językowych jest możliwość automatycznego przekształcania zbioru prostych opisów w językach dziedzinowych w elementy o wiele bardziej skomplikowane, co pozwala na minimalizację kosztów i czasu potrzebnego na dostosowanie rozwiązań do nowych wymagań funkcjonalnych oraz technologicznych.

W kolejnych punktach przedstawimy: autorską metodę budowy serwisów informacyjnych, następnie opis procesu budowy języków dziedzinowych i zakończymy określeniem zakresu transformacji opracowanych modeli.

2. Metoda budowy serwisów informacyjnych KMS

Metoda KMS bazuje na założeniach metodyk MDA (ang. Model Driven Architecture) [22] oraz MDE (ang. Model Driven Engineering) [26] i określa zestaw wymaganych działań, sposób ich realizacji oraz zasady dotyczące budowy modeli, transformacji i testowania. Związki między KMS a MDA przedstawiono na (Rys. 1). W zakresie budowy narzędzi wspomagających wytwarzanie języków dziedzinowych, metoda ta odwołuje się do podejścia DSM [14][15] i stanowisk językowych [12].

W ramach proponowanej metody główne działania koncentrują się na budowie niezależnych metamodeli, odpowiadających im profili języka UML i transformacji - definiując stanowisko językowe – a właściwie wewnętrzne języki dziedzinowe (DSLs), które go tworzą. Abstrakcyjna składnia opisana jest przez metamodel – schemat w MOF (Meta-Object Facility) [23]. Edytor abstrakcyjnej reprezentacji stanowi profil języka UML [25], a generator opisany jest zbiorem transformacji [21].



Rysunek 1. Mapowania platform MDA na dyscypliny KMS

W metodzie zostały określone następujące transformacje (Rys. 1): TK (transformacja konceptualna) prowadząca do przekształcenia modelu wymagań w model dziedziny, TM (transformacja modeli), która przekształca uzupełniony o model ograniczeń i model mapowań, model dziedziny w model systemu, TF (transformacja finalna), zawierająca model szablonu przekształceń modelu systemu w jego kod, podobnie jak TJ (transformacja JET).

Metoda KMS określa zestaw wymaganych działań przewidzianych w procesie wytwórczym jako jej dyscyplin (model wymagań, model dziedziny (zawierający model ograniczeń), model systemu poprzedzony podjęciem decyzji architektonicznych wyrażonych za pomocą modelu mapowań, kod) oraz scenariusze ich realizacji. Scenariusze przedstawiono na diagramach aktywności, w których dyscypliny określono jako tory, a kolejność wykonywania działań opisana jest przez akcje i związki między nimi (Rys. 2).

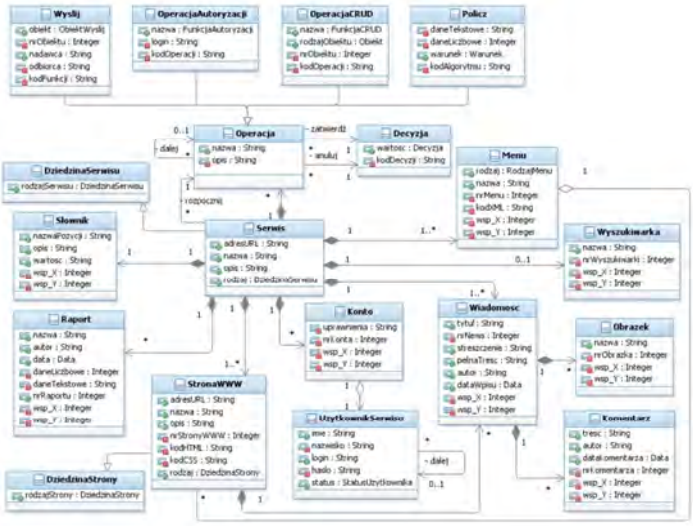
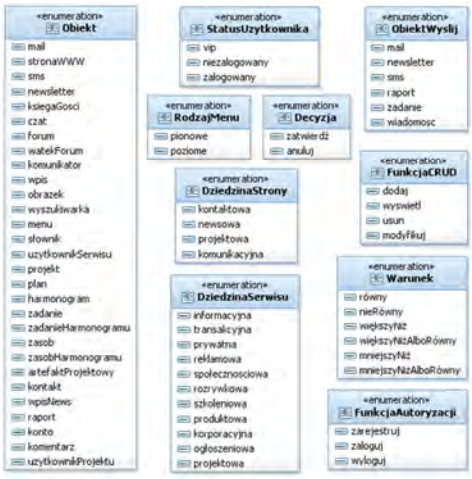
W proponowanej metodzie, z punktu widzenia zarządzania projektem zakłada się, że przejście do implementacji następuje po zakończeniu modelowania, w którym w sposób przyrostowy budujemy kolejne wydania „przetestowanych modeli" zapisanych w językach dziedzinowych (DSLs). Implementacja natomiast jest realizowana dopiero po dostarczeniu końcowej wersji modelu systemu i w znacznym stopniu stanowi wynik automatycznych transformacji modeli w kod systemu i opisy wymaganych konfiguracji. W metodzie przyjęto, że wszystkie opracowywane zgodnie z MDA modele zostaną połączone transformacjami: manualnymi (Manual) bądź automatycznymi (Model-to-Model [M2M], Model-to-Text [M2T]).

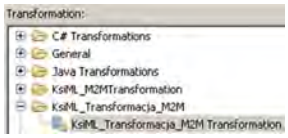
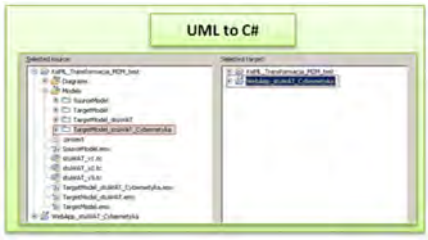
Koncepcję proponowanego podejścia transformacyjnego można opisać zależnością:

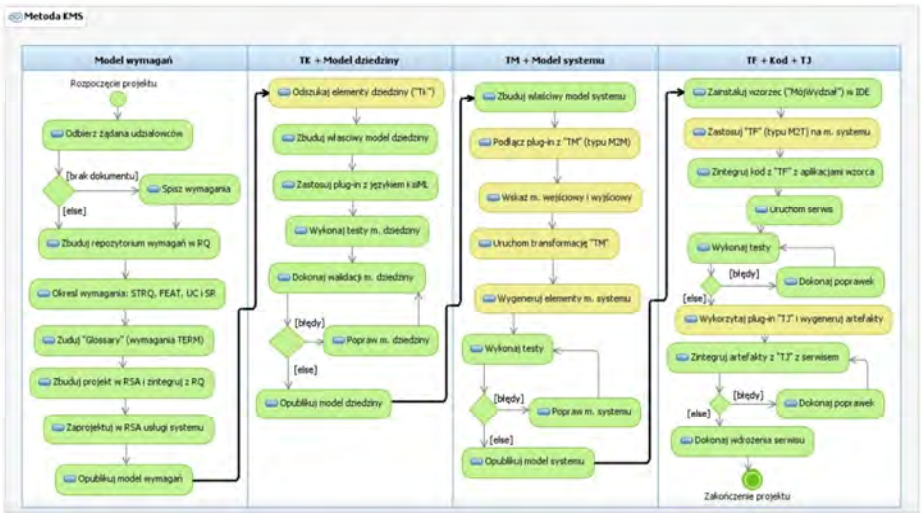
$$\mathbf{CIM}[RM] \xrightarrow{\text{Manual}} \mathbf{PIM}[DM] \xrightarrow{\text{M2M}} \mathbf{PSM}[\{SM\}] \xrightarrow{\text{M2T}} \mathbf{IM}[\{Code\}],$$

gdzie: RM – Model wymagań (ang. Requirement Model), DM – Model dziedziny (ang. Domain Model), SM – Modele systemu (ang. set of System Models), CODE – Kod (ang. set of Implementation Models). W metodzie KMS proponuje się przyjęcie etapów procesu wytwórczego zgodnych z RUP (Rys. 3), tj.: etap rozpoczęcia (ang. Inception phase), etap rozwinięcia (ang. Elaboration phase), etap konstrukcji (ang. Construction phase), etap przekazania (ang. Transition phase).

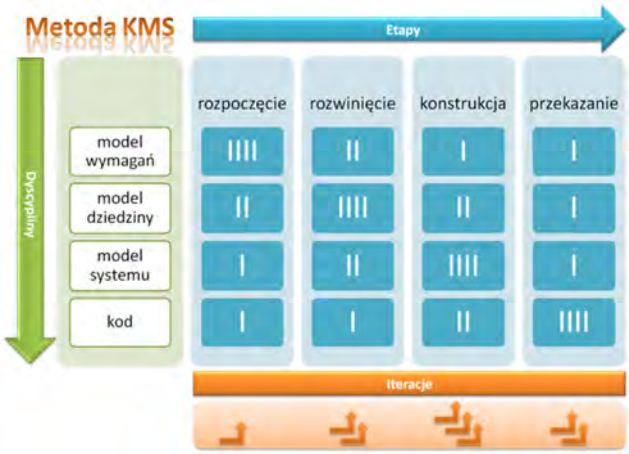
Tabela 1. Zestawienie artefaktów dyscyplin i narzędzi wykorzystywanych w metodzie KMS

<table>
<tr><th>Dyscyplina</th><th>Artefakty</th><th>Narzędzia</th></tr>
<tr><td>Model wymagań</td><td>repozytorium wymagań + model usług
 </td><td>notacja UML oraz standardowe mechanizmy dostępne w narzędziach CASE (np. RSA)
 </td></tr>
<tr><td rowspan="2">Model dziedziny</td><td>metamodel
  </td><td rowspan="2">język dziedzinowy KsiML repezentowany trzema profilami
 </td></tr>
<tr><td>model ograniczeń
  </td></tr>
</table>

Model systemu	model mapowań (M2M) 	mechanizm transformacji M2M 
Kod	model mapowań (M2T) 	mechanizm transformacji M2T



Rysunek 2. Algorytm działania metody KMS



Rysunek 3. Dyscypliny, etapy i iteracje metody KMS

3. Stanowiska językowe dla serwisu informacyjnego o projektach studenckich

Jednym z pierwszych skutecznych środowisk wspierających procesy modelowania języków dziedzinowych (wg. koncepcji „fabryk oprogramowania") było SDK [5][8] dla środowiska Microsoft Visual Studio (Visual Studio Visualization and Modeling SDK (VMSDK)). Jednak obecnie rozwój tej koncepcji wytwarzania oprogramowania zasadniczo koncentruje się na platformie Eclipse Modeling Framework [4], jako Generic Eclipse Modeling System (GEMS) [27].

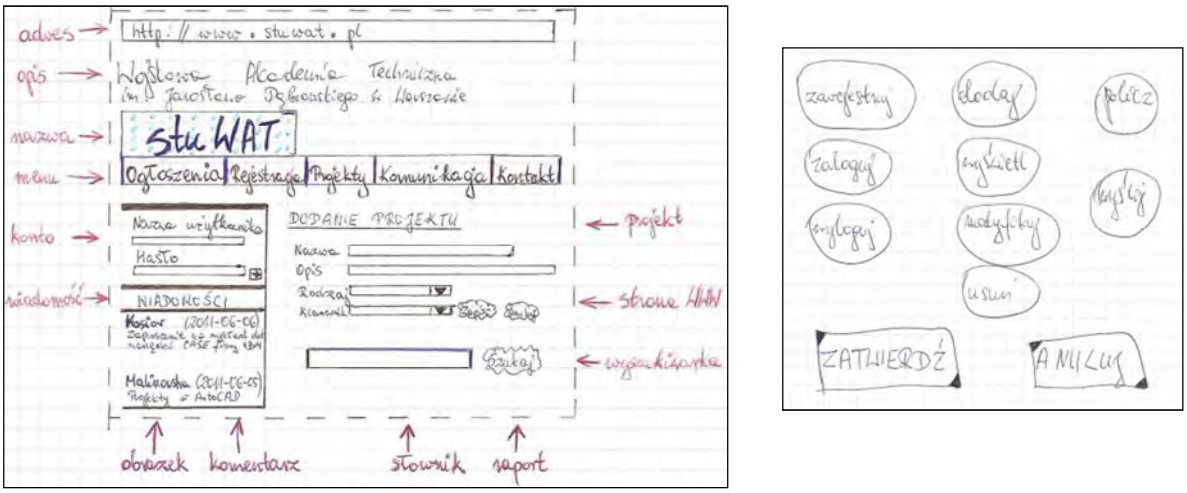
W swojej pracy nad budową serwisów informacyjnych, których z założenia miało powstać kilka, postanowiliśmy odwołać się do koncepcji języków dziedzinowych (DSL) głównie ze względu na możliwość weryfikacji utworzonych modeli dla docelowej dziedziny przez szybkie konstruowanie rozwiązań (ang. „correct by construction").

3.1. Szkic dziedziny

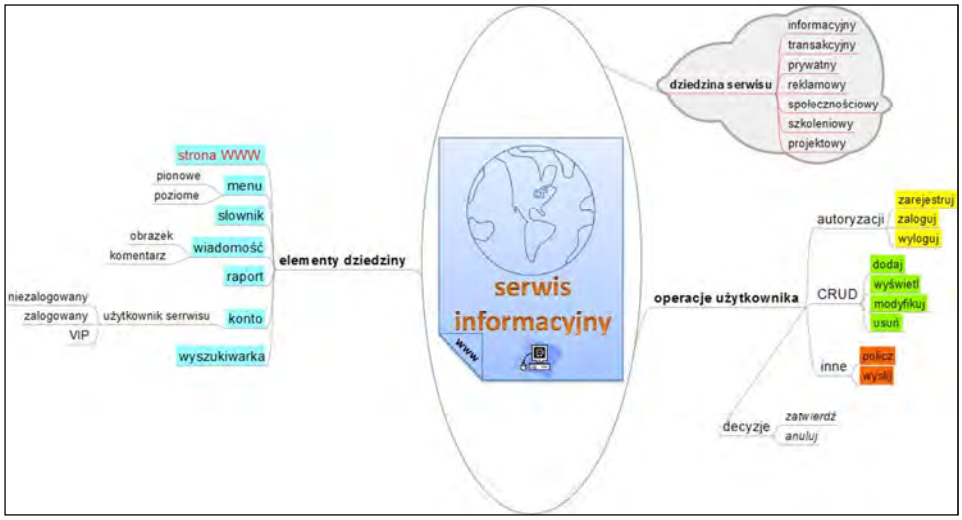
Zaprojektowanie nowego języka DSL wymaga przeprowadzenia szczegółowej analizy dziedziny [5][12][15], w wyniku której powinien powstać wstępny szkic modelu dziedziny [14][18].

Szkicowanie pozwala na określenie „głównych ram" dziedziny, ponieważ ma na celu zobrazowanie wszystkich elementów języka, zawierających pewne wspólne cechy i właściwości, które muszą zostać włączone do składni nowego języka DSL. Zwykle szkic dziedziny można wykonać odręcznie (z pomocą kartki papieru i długopisu), ale również przy pomocy elektronicznych aplikacji (chociażby tych, które umożliwiają projektowanie tzw. „map myśli" (ang. Mind Mapping)).

Przykłady takich szkiców zostały zaprezentowane na (Rys. 4) i (Rys. 5).



Rysunek 4. Szkic modelu dziedziny



Rysunek 5. Szkic modelu dziedziny (elektroniczna mapa myśli)

3.2. Metamodel

Na podstawie analizy utworzonych szkiców (Rys. 4) i (Rys. 5), stanowiących nieformalny opis dziedziny, budowany jest metamodel nazywany modelem koncepcyjnym (Tabela 1. Model dziedziny). Opisuje on podmioty (byty) jako obiekty (koncepcje) - ich atrybuty, relacje, oraz reguły ich łączenia i ograniczenia. Jego reprezentacją graficzną będzie diagram klas języka UML, który zobrazuje słownictwo i podstawowe pojęcia z dziedziny problemu. Metamodel jest wzorem, weryfikatorem oraz walidatorem budowanych modeli, które przedstawiają konkretny problem projektowy z danej dziedziny. Poprawnie przemyślana struktura dziedziny określa obraz pojęciowy badanego problemu i może być wykorzystana do weryfikacji i walidacji znajomości dziedziny. Zgodnie z definicją OMG [23], budowa oprogramowania opiera się na wspólnym metamodelu opisującym sposób zapisu modelu konkretnego systemu.

3.2.1. Reguły semantyczne

Budowę metamodelu opiera się na szkicu dziedziny [14] oraz na repozytorium wymagań [6], a w szczególności słowniku projektu będącym naturalnym źródłem informacji

o dziedzinie problemu i pozwalającym na precyzyjniejsze modelowanie elementów języka dziedzinowego.

Projektowanie języków dziedzinowych związane jest najczęściej z konkretnym portfelem projektów, lub projektem informatycznym, który ma ściśle określony zakres funkcjonalny. Z uwagi na potrzebę reużycia elementów języka dziedzinowego w innych projektach należy pamiętać, że kompletny słownik języka, to taki który umożliwia rozwiązywanie wszystkich problemów projektowych w określonej dziedzinie. Wynika z tego, że należałoby określić możliwie największy, tzn. pełny zbiór terminów tworzących opis dziedziny, co jednak powodowałoby znaczny wzrost złożoności języka i trudności w jego poprawnym stosowaniu. W praktyce, więc poszukujemy minimalnego zbioru terminów, który pozwala opisać nam maksymalną (lub przynajmniej określoną w wymaganiach na modelowany system) liczbę problemów projektowych w dziedzinie projektu.

Język dziedzinowy ma dostarczać wielowymiarowy słownik do budowy rozwiązań w danej dziedzinie, dlatego podczas jego projektowania, należy pamiętać o właściwym jego zakresie, kompletności i szczegółowości, łatwości obsługi i przyjazności narzędzi do tworzenia modeli. Z analizy tych cech wynika, że opracowywanie kompletnego języka dziedzinowego będzie można skutecznie przeprowadzić odwołując się do procesu iteracyjnego, w którym stopniowo rozbudowujemy jego zakres zapewniając wymaganą szczegółowość i kompletność, nie zapominając jednak o przyjazności i łatwości jego obsługi. W pierwszym cyklu zaleca się określenie krytycznego słownictwa, a w kolejnych iteracjach jego rozwinięcie i doprecyzowanie (wykorzystując do tego celu opinie ekspertów dziedzinowych, deweloperów oraz wyniki testów modeli). Używając informacji zwrotnych od użytkowników języka należy ciągłe udoskonalać DSL, sprawdzając w szczególności: przyjęte zasady i założenia, użyteczność języka, jego kompletność i zakres.

3.2.2. Reguły syntaktyczne

Wyodrębnione klasy określiły słownictwo dziedziny, zaś atrybuty, relacje i ograniczenia (stanowiące reguły budowy i łączenia elementów dziedziny), doprowadziły do jego uszczegółowienia [13]. Zaprojektowane reguły tworzą wspólnie tzw. model ograniczeń, który można uszczegółowić za pomocą modeli języka UML z ograniczeniami doprecyzowanymi w języku OCL [16][24]. OCL posiada zestaw wbudowanych operatorów, predykatów, ma możliwość definiowania własnych funkcji, warunków i niezmienników. Może być używany przy niemal wszystkich elementach występujących w UML. Język OCL można scharakteryzować następującymi własnościami:

- wyraża dowolną regułę logiczną: warunki wstępne, końcowe, niezmienniki, wyniki metod itd.,
- nie może modyfikować modelu, jedynie go sprawdzać,
- można go związać z dowolnym elementem modelu (klasą, operacją, atrybutem, asocjacją itd.).

Zbudowany metamodel (Tabela 1 - Metamodel), oprócz szablonowych typów (tzw. „UMLPrimitiveTypes") [13], został wzbogacony autorskimi wyliczeniami (klasami enumeration).

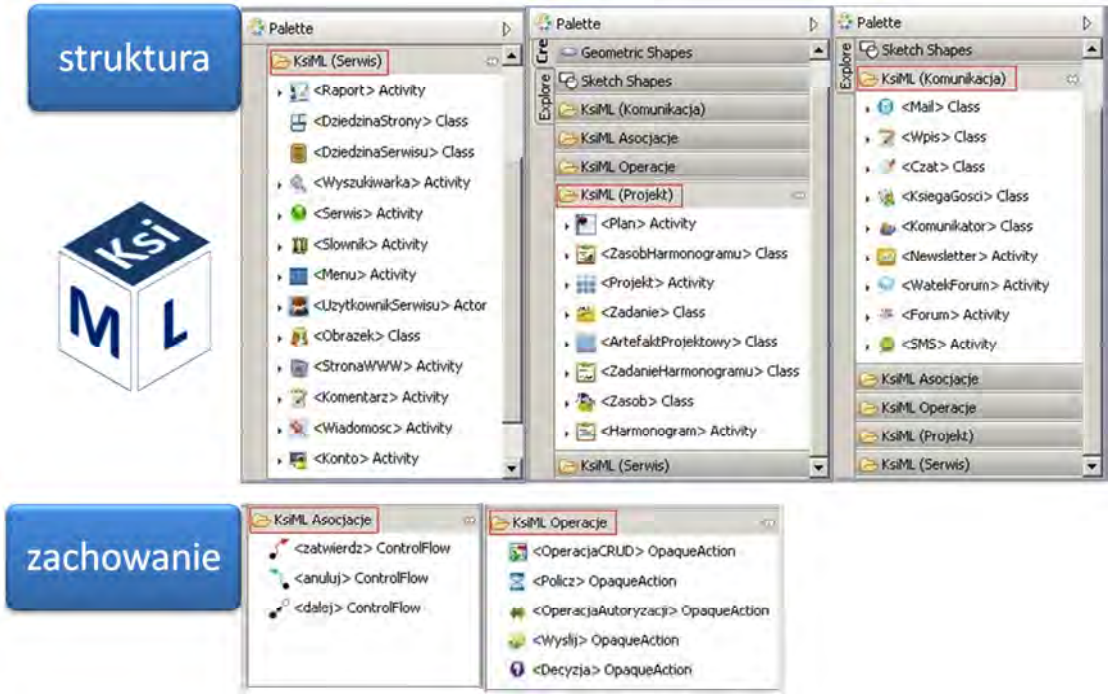
3.3. Profil UML

Metamodel jest formalną definicją języka dziedzinowego [12][14]. Określa jego słownik oraz gramatykę, czyli pokazuje kontekst pojęciowy badanego problemu, przy jednoczesnym zapewnieniu, że DSL jest semantycznie kompletny, jednoznaczny i skoncentrowany wyłącznie na specyficznej (konkretnej) dziedzinie.

Wykorzystywanie nowych języków dziedzinowych w środowiskach CASE wymaga wcześniejszego wdrożenia odpowiednich mechanizmów za pomocą profili UML.

Warto podkreślić, że język UML posiada trzy rodzaje mechanizmów rozszerzeń (profile, stereotypy, metki) [3][6][20], które pozwalają na zwiększenie zakresu zastosowania tego języka poprzez redefiniowanie pojęć i doprecyzowanie ich znaczenia do nowych obszarów zastosowań. Piktograficzną reprezentacją profilu jest jego paleta, która określa nie tylko pojęcia (ich graficzne metafory) opisywanych nią wewnętrznych języków dziedzinowych, ale również kontekst ich użycia – typy diagramów na których wystąpienia pojęć będą poprawne.

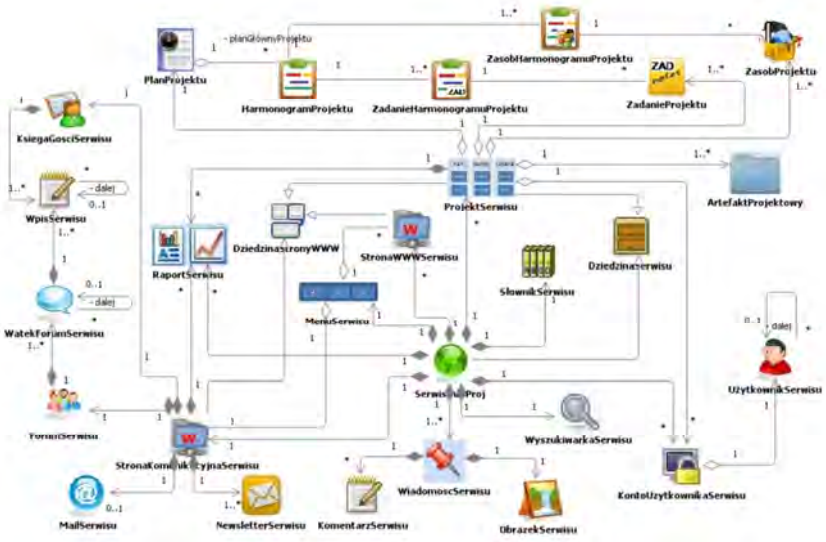
Palety dla profili KsiML_Serwis, KsiML_Projekt, KsiML_Komunikacja przedstawiono na (Rys. 6).



Rysunek 6. Palety dla profili „KsiML_Serwis", „KsiML_Projekt", „KsiML_Komunikacja"

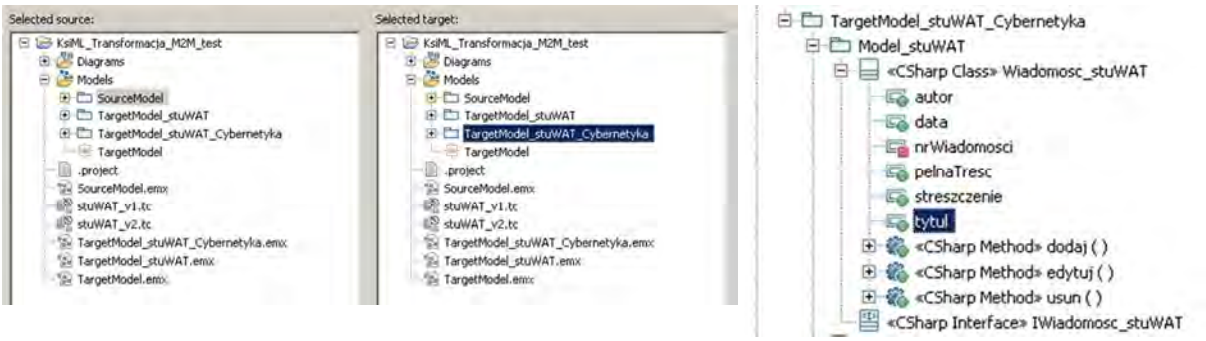
3.3. Model systemu i jego transformacje

Model systemu stanowi zbiór wyrażeń zapisanych w językach dziedzinowych należących do danego stanowiska językowego. Wyrażenia te zapisane są w postaci diagramów. Przykładowy model opisu struktury serwisu informacyjnego w języku KsiML przedstawiono na (Rys. 7).



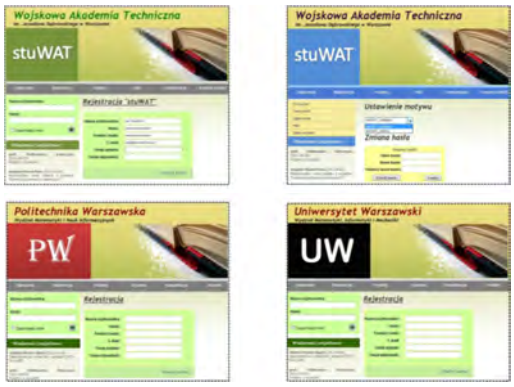
Rysunek 7. Przykładowy model serwisu informacyjnego w języku KsiML

W zastosowanej metodzie model systemu zapisany w profilu KsiML – jako źródłowy poddawany jest transformacji typu model2model (M2M) [19], w wyniku działania której uzyskujemy przekształcony z KsiML model UML, opisany przez profil „CSharp_Transformation" jako model wynikowy (Rys. 8), który następnie możemy poprzez firmową transformację uml2cs przekształcić w kod C#.



Rysunek 8. Transformacja – modele: wejściowy, wyjściowy, wynikowy

Opis procesu budowy automatycznych transformacji zawarto w [17][19], prezentując zarówno scenariusze ich tworzenia jak i uzyskiwane wyniki dla wszystkich ich typów: M2M- transformacja typu model-model (Model- to- Model), M2T- transformacja typu model-tekst (Model- to-Text), T2M- transformacja typu tekst-model (Text - to- Model), JET- transformacja typu Java Emitter Template.



Rysunek 9. Wyniki działania transformacji JET. „Seryjna" budowa serwisów

3. Podsumowanie

Wynikiem przeprowadzonych prac, opisanych w artykule, jest opracowanie i skuteczna weryfikacja zaproponowanej metody wytwarzania serwisów informacyjnych bazującej na koncepcji stanowisk językowych (Language Workbenches). Przewidywanym wynikiem zastosowania proponowanej metody będzie znaczne zwiększenie efektywności procesów wytwórczych oprogramowania, przy założeniu wielokrotnego użycia opracowanych języków dziedzinowych w wielu projektach. Warto tu również dodać, że koncepcja wytwarzania serwisów informacyjnych polegająca na budowie modeli oraz ich transformacji sprzyja powstawaniu generycznych rozwiązań (Rys. 9).

Bibliografia

[1] Ackerman L., Portier B., Gerken C.: *Using model-driven development and pattern-based engineering to design SOA: Part 4. Model-to-model transformations and connecting models to EMFT JET transformations*. IBM developerWorks, 2006. http://www.ibm.com/developerworks/edu/dw-r-umlprofiles4.html

[2] Aniszczyk C., Marz N.: *Create more – better – code in Eclipse with JET*. IBM developerWorks, 2006. *http://www.ibm.com/developerworks/opensource/library/os-ecl-jet/*

[3] Booch G., Rumbaugh J., Jacobson I: *Unified Modeling Language User Guide, 2nd Ed.* Addison-Wesley, 2005.

[4] Budinsky F., Steinberg D., Merks E., Ellersick R., Grose T.J.: *Eclipse Modeling Framework*. Addison-Wesley, Reading, MA (2003)

[5] Cook S., Jones G., Kent S., Wills A.C.: *Domain-Specific Development with Visual Studio DSL Tools*. Addison-Wesley (2007)

[6] Dąbrowski W., Stasiak A., Wolski M.: *Modelowanie systemów informatycznych w języku UML 2.1*, PWN, Warszawa, 2007

[7] Diu W.: *Custom domain modeling with UML Profiles: Part 1. Generating and deploying tooling*. IBM developerWorks, 2008. *http://www.ibm.com/developerworks/rational /library/08/0506_diu1/*

[8] Dubielewicz I., Hnatkowska B., Huzar Z., Tuzinkiewicz L.: *Języki dziedzinowe w modelowaniu danych w kontekście MDA*, Bazy danych. Rozwój metod i technologii. Architektura, metody formalne i zaawansowana analiza danych. Tom 1, Politechnika Śląska w Gliwicach, WKŁ 2008

[9] Dusko Misic: *Authoring UML Profiles: Part 1. Using Rational Software Architect, Rational Systems Developer, and Rational Software Modeler to create and deploy UML*

Profiles, 2008. *http://www.ibm.com/developerworks/rational/library/08/0429_misic1/index.html*

[10] Dusko Misic: *Authoring UML Profiles: Part 2. Using Rational Software Architect, Rational Systems Developer, and Rational Software Modeler to create and deploy UML Profiles*, 2008. *http://www.ibm.com/developerworks/rational/library/08/0429_misic2/index.html*

[11] Jackson E.K., Sztipanovits J.: *Formalizing the Structural Semantics of Domain-specific Modeling Languages*, Journal of Software and System Modeling, 2009

[12] Fowler M., Parsons R.: *Domain Specific Languages*. Addison Wesley, 2010

[13] Hovater S.: *Implementing a domain-specific constraint in IBM Rational Systems Developer*. IBM developerWorks, 2006. *http://www.ibm.com/developerworks/rational/education/dw-rt-rsdconstraint/*

[14] IBM: *Domain-Specific Modeling with IBM Rational Software Architect V7.5*, 2009

[15] Tolvanen K.S.: *Domain-Specific Modeling: Enabling Full Code Generation*, NJ: Wiley, 2008.

[16] Kleppe A., Warmer J.: *OCL. Precyzyjne modelowanie w UML*, WNT, 2003

[17] Kosior A.: *Projekt serwisu informacyjnego o projektach studenckich realizowanych w WAT.* Praca dyplomowa, Wydział Cybernetyki WAT, 2011.

[18] Kosior A., Stasiak A.: *Wytwarzanie serwisów informacyjnych w oparciu o koncepcję modelowania dziedzin. Metoda KMS*, Biuletyn IAR, NR 1, 2012 (złożono w redakcji)

[19] Kosior A., Dąbrowski W., Stasiak A.: *Wytwarzanie serwisów informacyjnych w oparciu o koncepcję modelowania dziedzin. Budowa transformacji*. Biuletyn IAR, NR 1, 2012 (złożono w redakcji)

[20] OMG. *Catalog of UML Profile Specifications*. *http://www.omg.org/technology/documents/profile_catalog.htm*

[21] OMG. *Documents associated with Meta Object Facility (MOF) 2.0 Query/View/Transformation, v1.1*, 2011, *http://www.omg.org/spec/QVT/1.1/*

[22] OMG. *MDA Specifications. The Architecture of Choice for a Changing World http://www.omg.org/mda/specs.htm*

[23] OMG. *Meta Object Facility (MOF) Core Specification Version 2.4.1*, OMG 2011. *http://www.omg.org/spec/MOF/2.4.1/PDF/*

[24] OMG. *Object Constraint Language, Version 2.2*, 2010, *http://www.omg.org/spec/OCL/2.2/*

[25] OMG. *UML Superstructure Specification v2.4*, OMG 2011 (chapter 18: Profiles). *http://www.omg.org/spec/UML/2.4/*

[26] Schmidt D.C.: *Model-Driven Engineering*. IEEE Computer 39(2), (2006)

[27] White J., Schmidt D.C., Nechypurenko A., Wuchner E.: *Introduction to the Generic Eclipse Modeling System*. Eclipse Magazine 7 (2007)

Gry symulacyjne w zarządzaniu projektami informatycznymi

Karol Domagała, Włodzimierz Dąbrowski

Politechnika Warszawska, Instytut Sterowania i Elektroniki Przemysłowej

Abstract:

The article presents the possibility of using computer games to train managers of IT projects. Described the ideas of the production of games, the realization of the objectives and characteristics. The conception of the possible development of simulation game focused on mapping tasks the project manager. The paper presents a prototype simulation game that allows IT project management training in different scenarios and different methodologies

Keywords:

Software Project Management, Software Games

1. Wprowadzenie

W artykule przedstawiono problematykę budowania, projektowania i wykorzystania gier symulacyjnych w nauczaniu zarządzania projektami informatycznymi. Problem wyszkolenia dobrego kierownika projektów uwzględniających specyfikę projektów informatycznych jest trudny. Autorzy prowadzą szkolenia z obszaru zarządzania projektami informatycznymi od wielu lat prowadząc zajęcia zarówno dla studentów kierunków informatycznych jak i dla osób z doświadczeniem zawodowym. W każdym szkoleniu należy dobrać odpowiednie środki metody zajęć, aby nie tylko przekazać studentom wiedzę z zakresu zarządzania ale i wykształcić w nich odpowiednie umiejętności, zainteresować i zaangażować emocjonalnie w wykonywanie zadań. Jedną z form edukacji aktywizującej opartej na podejściu POL (Project Oriented Learning) są gry symulacyjne [5]. Gry symulacyjne w edukacji znane są od wielu lat. Autorzy w niniejszej pracy prezentują swoje doświadczenia w wykorzystaniu gier symulacyjnych do nauczania wiedzy i kompetencji z obszaru zarządzania projektami informatycznymi, które zdobyli w czasie ponad dziesięcioletniej praktyce prowadzenia zajęć.

Oprogramowanie symulacyjne pozwala na lepsze i bardziej efektywne szkolenia członków zespołu projektowego, w szczególności zaś kierowania projektu. Propozycje przedstawione w artykule koncentrują się na specyfice projektówa) informatycznych. Opracowany przez autorówb) podejście pozwala na generowanie scenariuszy uwzględniających różne metodyki prowadzenia projektówc) informatycznych – zarówno zwinnych jak i prowadzonych metodami tradycyjnymi. Z tego punktu widzenia opracowane narzędzie jest narzędziem generycznym, które może być wykorzystane w szerokim zakresie kształcenia menadżerów, osób zarządzających ryzykiem, jakością itp. i pozostałych członków zespołu projektowego.

W kolejnych punktach przedstawiono: autorską metodę budowy scenariuszy oraz narzędzie do przeprowadzania gry symulacyjnej z zarządzania projektami informatycznymi.

2. Specyfika budowy i wytwarzania oprogramowania gier komputerowych

Granie oznacza szeroką kategorię nieistotnych, a zazwyczaj rekreacyjnych czynności ludzkich, które często mają także znaczenie towarzyskie. Udawanie jest umysłową zdolnością do tworzenia wymyślonej rzeczywistości, o której wiadomo, że różni się od świata realnego. Gry pojawiły się z ludzkiej potrzeby grania oraz ludzkiej zdolności do udawania. Granie i udawanie są podstawowymi elementami gier. E. Adams [1]definiuje grę, jako rodzaj aktywności wykorzystującej udawaną rzeczywistość, w której uczestnik (lub uczestnicy) usiłuje osiągnąć przynajmniej jeden, dowolny, nietrywialny cel poprzez wykonywanie działań zgodnych z regułami. Reguły gry często charakteryzują jej cel, jako warunek zwycięstwa. Jest to jednoznaczna sytuacja w grze, w której jeden lub więcej graczy staje się zwycięzcami. Adams zastrzega jednak, że definicja nie jest ścisła, lecz istnieje, jako wygodny opis zgodny z większością przypadków. Jeśli projektanci gier uważają, że ustalenie określonego zwycięzcy jest ważną sprawą, muszą również stworzyć dodatkowe reguły związane z sytuacjami równowagi. Reguła, która określa sytuację zakończenia gry, jest nazywana warunkiem zakończenia. Jednak nie we wszystkich grach jest to warunek zwycięstwa. Niektóre definiują jedynie warunek porażki [2]. Jest to sytuacja oznaczająca zakończenie gry po określeniu, który z graczy przegrał. Reguły i cel gry zawierają się całkowicie w wirtualnej rzeczywistości, lecz zasada zwycięstwa i porażki wykracza do świata rzeczywistego powodując, że po zakończeniu rozgrywki gracz jest dumny ze zwycięstwa lub smutny po porażce.

2.1. Strategie w grach symulacyjnych

Strategia jest planem umożliwiającym odpowiednie granie w grę [1,2]. Zwykle związana jest z pewną zasadą lub teorią gracza, według której zadanie powinno zostać zakończone sukcesem. Większość gier fabularnych daje graczowi dużą swobodę w opracowywaniu własnej strategii rozgrywki, dzięki czemu umożliwia odczucia większej swobody w świecie gry. Określenie strategii dominującej odwołuje się do klasycznej teorii gier i odnosi się do strategii, która niezawodnie pozwoli graczowi na osiągnięcie sukcesu bez względu na czynności przeciwników. Strategie dominujące są niepożądane w schemacie gry, ponieważ jeśli gracz odkryje jedną z nich najczęściej nie zechce szukać innych sposobów wygrywania. Natomiast, gdy gracz zacznie stosować tego typu strategię w grach wieloosobowych, wówczas nie tylko zmniejszy poziom rozgrywki, ale sprawi, że gra stanie się niesprawiedliwa. Jeśli dla strategicznych opcji gracz istnieje związek przechodni, wówczas opcja A jest lepsza od opcji B, zaś opcja B lepsza od opcji C, inteligentny gracz nigdy nie wybierze strategii C, za to zawsze będzie korzystał z A jako zapewniającej największe korzyści. Jest ona nadrzędna w stosunku do pozostałych. Aby naprawić brak równowagi, można wprowadzić dla każdej ze strategii koszty jej wykorzystania, co zmniejszy przewagę mocniejszych opcji i da graczowi powód do wzięcia po uwagę słabszych. Ustalanie kosztów bezpośrednich, które rekompensują zalety pewnych strategii pozwala na zrównoważenie rozgrywki, lecz powoduje odczucia, że gra jest nijaka. Gracz po pewnym czasie zauważy, że wybory praktycznie niczym się od siebie nie różnią. Aby stworzyć bardziej interesujące możliwości dla gracza można wprowadzić koszty ukryte, o których gracz nie jest powiadamiany w sposób jawny. Służą one do równoważenia gry w sposób dyskretny, niekiedy nawet nieudokumentowany w instrukcji użytkownika. Dzięki temu gracz będzie musiał poznać zalety i wady pewnych rozwiązań próbując każde z nich.

W starannie przemyślanych grach stosowane jest zjawisko nieprzechodniości związków[2]. Najprostszym przykładem jego zastosowania jest gra papier-kamień-nożyce, w której opcja A pokonuje B, opcja B pokonuje C, natomiast C pokonuje A. Reguły te tworzą

zrównoważony trójelementowy związek. Jednak, aby gracz miał bardziej interesujące możliwości dokonywania wyborów oraz dostęp do większej liczby strategii, obiekty nie powinny różnić się jedynie zdolnością do wykonywania pojedynczego zadania, lecz także wykazywać całkowicie odmienne właściwości. Idealną sytuacją byłoby, gdy każdy rodzaj obiektu mógł mieć cechy, których nie posiadają inne, co pozwoliłoby im na odgrywanie różnych ról w grze

3. Wykorzystania gier komputerowych do symulacji procesów zarządzania projektami informatycznymi

Gry symulacyjne w klasycznym znaczeniu są specyficznym sposobem szkolenia w danej dziedzinie tworząc sztuczne środowisko pracy ale pracując nad jak najbardziej realnym problemem. Oprócz poszerzania i kontroli kompetencji, tego typu rozgrywka wzmaga relacje interpersonalne pomiędzy członkami zespołu, a także umożliwia odczucie znacznie większej presji wyniku, czego nie można osiągnąć stosując gry indywidualne. Gry symulacyjne doskonale sprawdzają się przy szkoleniu osób zarządzających ludźmi. Dzięki bezpośredniemu kontaktowi z pozostałymi graczami możliwe jest nie tylko weryfikowanie zdobytej przez siebie wiedzy, ale również podpatrywanie dobrych praktyk u rywali. Gry tego typu są materiałami szkoleniowymi, wykorzystującymi formę rozrywki jako element rozszerzający wiedzę i umiejętności.

3.1. Cechy charakterystyczne symulatorów

Obok tradycyjnych gier symulacyjnych skupionych na bezpośredniej interakcji z innymi ludźmi, coraz częściej pojawiają się specjalistyczne gry wideo będące symulatorami dla określonej specjalności. Jedną z podstawowych cech symulatorów jest możliwie najwierniejsze odwzorowanie rzeczywistości. Ze względu na fakt, że urządzenia czy aplikacje mające spełniać takie zadanie przeznaczone są do nauki i doskonalenia fachowców z danej dziedziny, skupione są na wąskim zakresie operacji związanych tylko z jedną specjalnością. Rzeczywistość przedstawiona za ich pomocą odzwierciedla z możliwie najdrobniejszymi szczegółami prawa logiki, fizyki i ekonomii. Tego typu rozwiązania nie są już tak doskonałymi elementami szkoleniowymi jeśli chodzi o współpracę i integrację zespołów, ale pozwalają na utrwalenie wiadomości i sprawdzenie technik działania. Niepodważalnym atutem gier wideo jest możliwość powtórzenia rozgrywki z niemal identycznymi parametrami startowymi. Jednak nawet najdoskonalsze algorytmy implementujące pracę nie są w stanie dokładnie przewidzieć reakcji podwładnych w skutek niewłaściwej rozmowy motywacyjnej i upadającego morale zespołu.

Zaprezentowane w pracy rozwiązanie jest próbą zaprezentowania symulatora pracy menedżera projektów informatycznych pod kątem niewielkich – kilkuosobowych zespołów projektowych. Z tego względu symulator został wyposażony w możliwości zarządzania kadrami z perspektywy wyboru pracowników do zespołu, przydzielania zadań do realizacji oraz ustalania i korygowania harmonogramu prac. W przypadku symulacji środowisk, gdzie konieczna jest praca z innymi ludźmi – jak w przypadku menedżera projektów, niezbędne wydaje się opracowanie modeli psychologicznych różnych postaci, co znacznie utrudnia tego typu konstrukcje. Rozwiązania takie wymagają nie tylko znajomości dziedzinowej ale również współpracy z socjologami i filozofami społecznymi, którzy byli by w stanie nakreślić możliwe zachowania kadr i pożądane reakcje.

3.2. Specyfika pracy project managera w branży IT

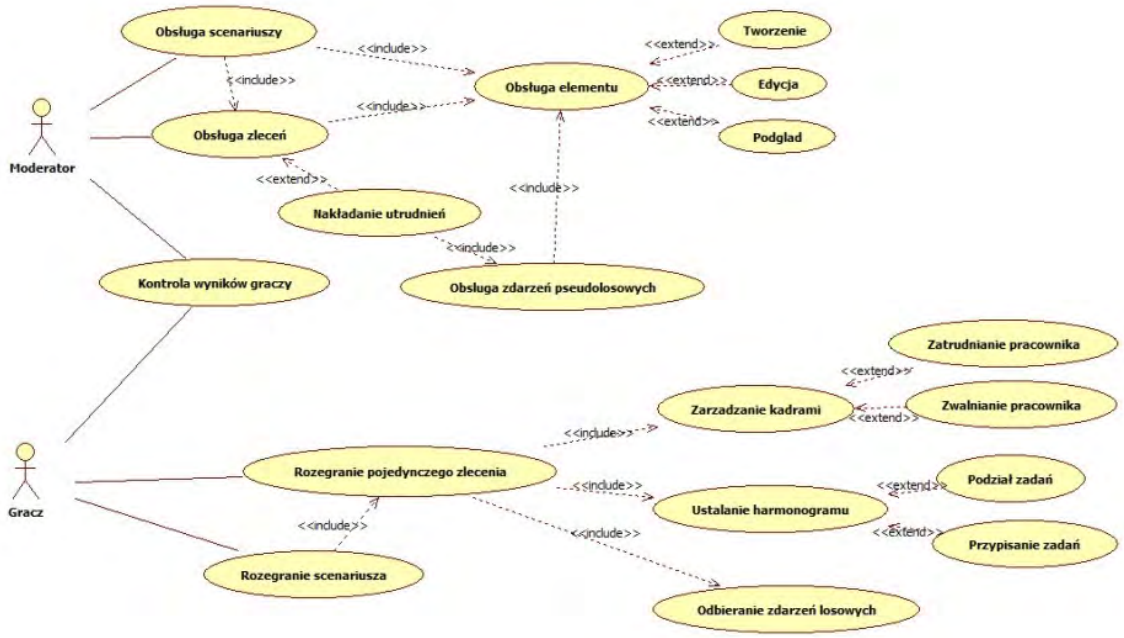
Przy każdym projekcie niezbędna jest osoba, która będzie kierownikiem całego przedsięwzięcia. Większość przedsiębiorstw zajmujących się realizacją projektów IT wykorzystuje przetestowane i udokumentowane standardy postępowania. Metodyki te często wypracowane i dostosowane wskutek dawnej działalności organizacji, określają dokładnie jakie zadania pełni kierownik projektu. W przypadku niewielkich przedsiębiorstw zdarza się, że menedżer oprócz funkcji zarządczych musi uczestniczyć w pracy nad projektem jako jeden z analityków, bądź testerów. W każdym jednak przypadku do zadań kierownika należy dbanie o prawidłowy rozwój projektu, terminowość realizacji zadań i kontrola wydatków. Takie też funkcje zostały przydzielone graczowi w zastosowanym rozwiązaniu. Gracz podczas realizacji zleceń odpowiada za prawidłowy przebieg prac i generowanie możliwie najniższych kosztów uzyskania przychodów. Oprócz zadań typowo realizacyjnych na menedżerze spoczywa również obowiązek dbania o zespół, przez utrzymywanie morale na odpowiednio wysokim poziomie, dobrze motywując zespół jednocześnie zapobiegając wewnętrznym konfliktów. Oprócz organizacji pracy zespołu kierownik projektu informatycznego bardzo często ma bezpośredni kontakt z klientem. Najbardziej jest to uwidocznione w przypadku stosowania metodyk zwinnych, gdzie konsultacje ze zleceniodawcą odbywają się regularnie. W takich przypadkach menedżer powinien mieć zdolności do prowadzenia negocjacji z osobami związanymi bezpośrednio z projektem, oraz potrafić podejmować szybkie i poprawne decyzje w sytuacjach krytycznych.

4. Oprogramowanie symulacyjne

Propozycje Autorów zostały zweryfikowane za pomocą oprogramowania symulacyjnego opracowanego na Politechnice Warszawskiej. Oprogramowanie to umożliwia zdefiniowanie scenariuszy rozgrywki dostosowanych do wybranej metodyki prowadzenia projektu informatycznego a następnie do przeprowadzenia rozgrywki z udziałem graczy.

Aplikacja oparta jest na architekturze klient-serwer, przy czym serwer zajmuje się przechowywaniem bazy danych, natomiast każdy z użytkowników korzysta z aplikacji desktopowej łącząc się z bazą w celu pobrania podstawowych informacji o rozgrywce. Aplikacja kliencka napisana jest w technologii Java z wykorzystaniem frameworka Vaadin umożliwiającego implementację widgetów i wykorzystanie styli CSS. Start aplikacji możliwy jest dzięki zastosowaniu lekkiego serwera Jetty. Choć domyślnie program współpracuje z bazą MySQL-ową, to dzięki korzystaniu z interfejsu JDBC i przy dość niewielkich modyfikacjach powinien równie dobrze dać się dostosować do konkurencyjnych produktów.

Aplikacja podzielona jest na dwa główne pakiety – moderatora i gracza. Po zalogowaniu się jako gracz możliwe jest zarówno rozgrywanie pojedynczych misji, jak i ciągu następujących po sobie zadań przygotowanych jako scenariusze.



Rysunek 1. Funkcjonalność symulatora

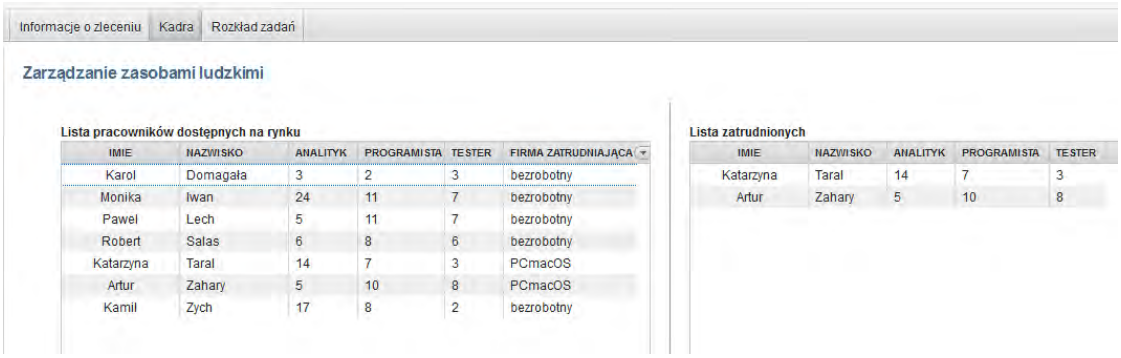
Gracz wciela się w postać menadżera projektów branży IT w stworzonej przez siebie firmie. Jego zadaniem jest uzbieranie możliwie jak największej liczby punktów przydzielanych za każde zlecenie zakończone sukcesem. Aby projekt został zakończony sukcesem należy tak manewrować pracownikami i terminarzem aby zmieścić się w określonym przedziale czasowym i finansowym. Dla każdego projektu jest bowiem określony sztywny budżet oraz termin realizacji. Rozgrywka pojedyncza przeznaczona jest przede wszystkim do gry wieloosobowej, gdzie gracz sam decyduje, który projekt chce realizować wybierając z przygotowanej przez moderatora puli zleceń, jednocześnie rywalizując z pozostałymi graczami zarówno w kwestii wyboru projektu jak i dostępnych zasobów. Podczas tej rozgrywki moderator może na bieżąco śledzić postępy graczy i przydzielać im utrudnienia przez generowanie zdarzeń imitujących wydarzenia mogące się wydarzyć w rzeczywistości – jak odejście pracownika, czy dodatkowe zadanie. Gra toczy się do momentu wyczerpania dostępnych zleceń. Rozgrywka według przygotowanych scenariuszy umożliwia, realizację projektów według ustalonej kolejności i każdy z graczy może zmierzyć się z takimi samymi wyzwaniami i identycznymi zdarzeniami pseudolosowymi. Zadania układane są przez moderatora, toteż powinien on zwrócić uwagę by ich poziom trudności następował rosnąco, wraz ze zdobywanym doświadczeniem. Każdy projekt dzielony jest na kilka modułów reprezentujących zbiory zadań i może być realizowany przez gracza według dwóch możliwych metodyk realizacji projektów: Agile SCRUM oraz PRINCE2. Jej rodzaj określany jest przez moderatora podczas projektowania projektu. W zależności od wybranego typu metodyki realizacja zlecenia ma odmienny przebieg odzwierciedlając podstawowe właściwości każdej z metodyk. Ogólny proces realizacji pojedynczego projektu można przedstawić w kilku krokach:

1. Sprawdzenie poziomu ryzyka zadania
2. Wybranie i zatrudnienie pracowników
3. Poddanie projektu analizie
4. Przypisanie pracowników do zadań
5. Umieszczenie punktów kontrolnych
6. Uruchomienie procesu realizacji i reagowanie na otrzymywane zdarzenia

Agile SCRUM wykorzystuje zasadę samoorganizacji zespołu projektowego dając pracownikom swobodę w wyborze swoich zadań, przez co nie można wymuszać przydzielenia zadania. Gracz nie musi wykonać wszystkich modułów jakie klient przewiduje, ale konieczne jest zrealizowanie wszystkich zadań z podjętych części projektu. Przydzielanie kolejnych modułów następuje w momencie gdy linia czasu osiąga punkt kontrolny symbolizujący Sprint. Aby gracz nie musiał rozgrywać wszystkich modułów powinien ustawić termin ostatniego Sprintu w ostatnim slocie czasowym realizacji projektu. Rezygnacja z realizacji modułów wiąże się jednak z otrzymaniem niepełnego wynagrodzenia za projekt, ponieważ gratyfikacje finansowe wpływają z każdym rozpoczynanym modułem. Zdarzenia otrzymywane podczas procesu wynikające ze zmiany założeń projektowych lub generowaniu dodatkowych zadań przez klienta nie są związane z dodatkowym wynagrodzeniem. Jest to element ciągłego doskonalenia projektu wynikającego z regularnych konsultacji z klientem i rozwijaniem zlecenia.

W przypadku metodyki PRINCE2 od gracza wymaga się zrealizowania wszystkich przewidzianych modułów. Są one jednak znane od samego początku realizacji projektu a wynagrodzenie przydzielane jest od razu na cały projekt. Metodyka przewiduje trzymanie się ścisłego harmonogramu prac i założeń toteż w przypadku otrzymania zdarzeń zadań dodatkowych klient musi przyznać premie za dodatkową pracę poza pierwotnymi ustaleniami. Ponieważ w PRINCE2 kierownik projektu jest odpowiedzialny za dopilnowanie realizacji wszystkich zadań, to podczas procesu rozgrywki konieczne jest ręczne przydzielanie zadań pracownikom wedle własnego uznania.

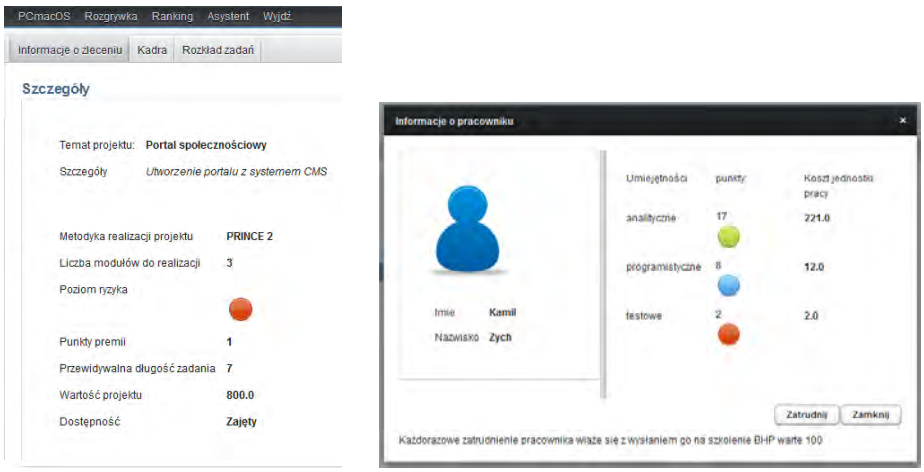
Analizując każde zadanie należy zwrócić uwagę na określony dla niego poziom ryzyka. Informacja ta ma odzwierciedlać przewidywalną możliwość niepowodzenia projektu wyznaczaną zwykle przez analityków podczas procesu przygotowania projektu. Analiza tego typu przeprowadzana jest przy ogromnej większości projektów. W tym przypadku moderator narzuca swoją ocenę ryzyka dla całego zlecenia. Poziomem ryzyka obarczone będzie również każde pojedyncze zadanie przy realizacji projektu, lecz wówczas taka ocena jest symulowana przez odpowiedniego pracownika pełniącego funkcję analityka. To właśnie analityk będzie osobą odpowiedzialną za podział zadań na mniejsze pakiety, analizę ryzyka i przewidywalne obciążenie czasowe. W rzeczywistości system wyznacza wynik analizy zadania a informacja przekazana graczowi od pracownika jest jedynie maską tych wyników przeliczonych na zdolności danego pracownika. W zależności od poziomu umiejętności pracownika na tym stanowisku jego analizy będą trafniejsze - to znaczy bardziej zbliżone do zasymulowanych, dzięki czemu gracz będzie miał pewność, że przypisując zadania tego typu osobie o większych umiejętnościach umożliwi mu uzyskanie bardziej wiarygodnych wyników. Poziomy ryzyka określone są kolorami, wśród których czerwony – oznacza zadanie o wysokim, żółty – o średnim, natomiast zielony o niskim poziomie ryzyka. Podobna sytuacja ma miejsce przy realizacji wygenerowanych zadań dla programisty czy testera. Wprawdzie w tym przypadku gracz nie otrzymuje już tak symulowanych wyników, jednak stosunek poziomu umiejętności pracownika na dane stanowisko do poziomu ryzyka przekłada się na rzeczywisty czas potrzebny na zrealizowanie zadania.



Rysunek 1. Wybór projektu do realizacji przy rozgrywce pojedynczej

TYTUŁ	PRZEWIDYWALNA DŁUGOŚĆ ZADANIA	WARTOŚĆ PROJEKTU	POZIOM RYZ' ▲	PUNKTY PREMII	DOSTĘPNOŚĆ
Nowy system CMS	15	150.0	Niski	1	Dostępny
Portal społecznościowy	7	800.0	Niski	1	Dostępny
proto	6	250.0	Niski	2	Dostępny
Gra RPG	35	9900.0	Wysoki	6	Dostępny
Mobilny Mini klient CRM	32	1100.0	Wysoki	3	Zakończony
Moduł finansowo-księgowy	28	1300.0	Średni	4	Zajęty
Opracowanie MINI systemu ERP	18	1500.0	Średni	2	Dostępny

Rysunek 2. Okno zatrudnienia pracowników



Rysunek 3. Informacje szczegółowe o danym zleceniu i poziomy trudności gracza

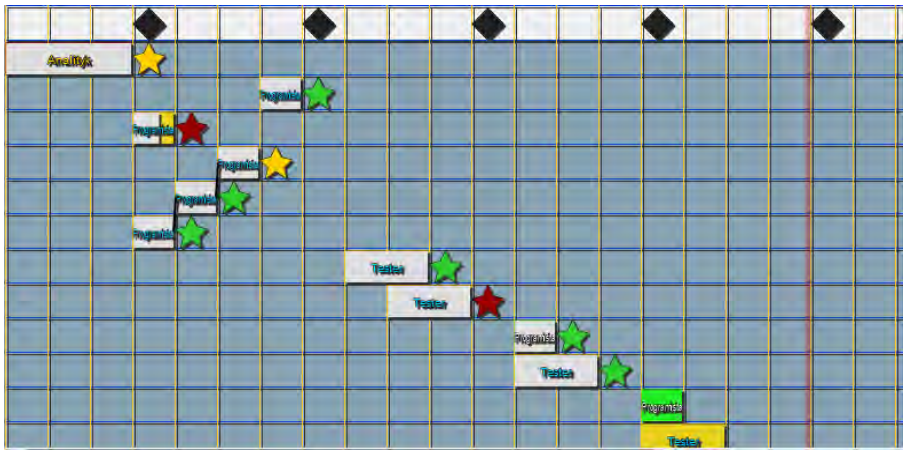
Należy więc tak kierować się wyborem pracowników na dane stanowisko, aby mogli sprostać wyznaczonym im zadaniom. Stopień dopasowania pracownika do powierzonego mu elementu określony jest przez kolor gwiazdki umieszczonej obok zadania na diagramie Gantta. W tym przypadku kolor zielony odpowiada za doskonałe, żółty – za średnie, a czerwony za kiepskie dopasowanie pracownika. Oczywiście możliwe jest wykonanie zadania przy kiepskim dopasowaniu jednak zajmie mu to znacznie większą liczbę jednostek czasu.

Każdy z potencjalnych zatrudnionych charakteryzuje się trzema informacjami punktowymi ściśle związanymi z możliwym stanowiskiem. Są to umiejętności analityczne, programistyczne i testowe. Zgodnie z ich wartością otrzymywane są wyniki pracy przy danym zadaniu. Im większa liczba punktów w danej kategorii tym pracownik działa szybciej i jest

bardziej odporny na poziom ryzyka związany z powierzonym mu zadaniem. Umiejętności dla każdego osoby zwiększają się wraz z kończeniem kolejnych zadań – oczywiście o ile są one dla pracownika choć minimalnym wyzwaniem, czyli jeżeli dopasowanie pracownika do stanowiska jest przeciętne lub kiepskie. W innym przypadku przyjęte jest, że pracownika wykonuje zadanie zbyt łatwe i nie rozwija swoich zdolności. Wartości punktowe przedstawiają osobę jako nowicjusza, praktyka lub profesjonalistę w danej dziedzinie. W zależności od tego statusu ustalane jest oczekiwane wynagrodzenie za jednostkę czasu. Gra nie uwzględnia elementów psychologicznych potencjalnych pracowników i zakłada, że każdy z nich pracuje na 100% swoich możliwości, wykorzystując całą jednostkę czasową tylko na pracę. Nie są tu również uwzględniane relacje interpersonalne, czy determinacja w działaniu. Przykład dopasowywania i ogólnie wyboru pracowników jest typowym elementem strategicznym w teorii gier. Gracz nie może wykorzystać samych najlepszych opcji ponieważ wiąże się to z ogromnymi kosztami utrzymania. Ma więc do wyboru zatrudnienie większą liczbę średniej jakości specjalistów lub mniej fachowców. Dodatkowo wzrastanie stawek płacowych i douczanie pracowników zapobiega strategiom dominującym, gdzie gracz za każdym razem będzie mógł wybrać ten sam personel by osiągnąć sukces.

Właściwie zatrudnianie i zwalnianie pracowników przy każdym projekcie jest dla gracza czynnością swobodną i ograniczoną jedynie budżetem, ale każde zatrudnienie wiąże się z wydaniem na pracownika sumy związanej z wysłaniem go na wstępne przeszkolenie, co przy ciągłych manipulacjach kadrowych może doprowadzić do załamania budżetu projektowego.

Główna część rozgrywki opiera się o uproszczony diagram Gantt'a. Jest to element przedstawiający postęp prac jednocześnie obserwując upływ czasu życia projektu. Uwzględnia on tylko pracujące jednostki czasu, bez urlopów i okresów wolnych od pracy. Na szczególną uwagę zasługuje tu pierwszy wiersz zawierający punkty kontrolne. Są to elementy, w których czas niejako się zatrzymuje a gracz ma wtedy czas na podjęcie odpowiednich modyfikacji czy reakcji na otrzymane zdarzenia. Punkty kontrolne dzielą się na dwa typy – Sprinty oraz spotkania pracownicze. W przypadku tych pierwszych odbywa się konsultacja z klientem i odbieranie od niego uwag, natomiast przy zwykłym spotkaniu pracowniczym gracz może skontrolować postępy i wprowadzić zmiany. Jest to uproszczony model metodyki Agile SCRUM wprowadzony na stałe w celu ułatwienia graczowi pracy również przy metodyce PRINCE2. W pozostałych wierszach znajduje się lista zadań, które gracz musi rozmieścić w taki sposób, by pracownicy je wykonujący zdążyli z realizacją przed kolejnym Sprintami.



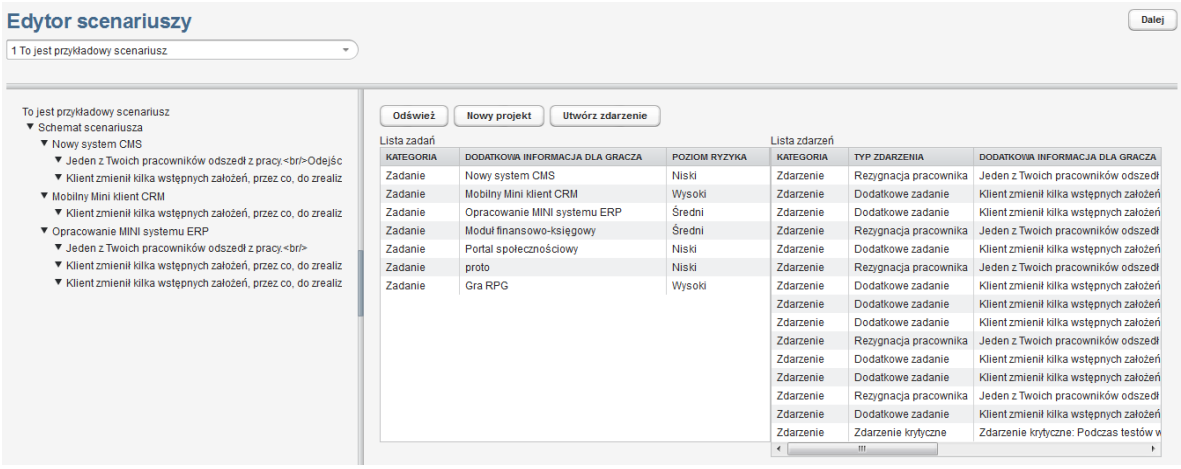
Rysunek 4. Uproszczony diagram Gantt'a z naniesionymi zadaniami do wykonania i przypisanymi współczynnikami dopasowania pracowników

Każde zadanie zawiera się w cyklu analiza – implementacja – testy. Wynikiem pracy analityka są zadania programistyczne, z czym zwykle pod wpływem analizy pojedynczego zadania wygenerowane zostanie wiele zleceń dla programistów. W wyniku działań programistycznych powstaje zlecenie dla testera, który najczęściej określa zadanie zwrotne z koniecznością implementacji poprawek. Jeśli zdaniem gracza któreś otrzymane zadanie jest obciążone zbyt wysokim ryzykiem niepowodzenia przy określonej kadrze może zlecić analitykowi jego podział. Dzięki temu otrzymanych zostanie kilka podrzędnych zadań o mniejszym ryzyku.

Pakiet Moderatora

Moduł moderatora pozwala na nadzorowanie przebiegu i uatrakcyjnianie gry, zwłaszcza w przypadku większej liczby graczy. Umożliwia zarządzanie projektami, pracownikami, graczami, a także wprowadzanie do rozgrywki zdarzeń pseudolosowych.

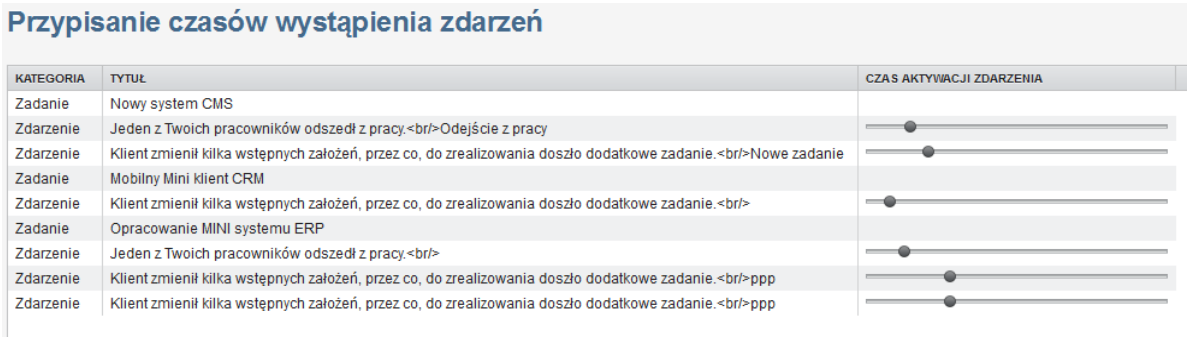
Sterowanie rozgrywką udostępnia możliwość tworzenia nowych zleceń, które będą mogły być realizowane przez graczy. Zlecenie charakteryzuje się tytułem, krótkim opisem, wartością pieniężną, przewidywaną ilością jednostek roboczych, określonym poziomem ryzyka niepowodzenia i liczbą punktów za poprawnie zrealizowany projekt. Dodatkowo dla każdego zlecenia należy narzucić model realizacji projektu – Agile SCRUM lub PRINCE2. Określenie metodyki ma na celu umożliwienie graczowi porównania różnych sposobów pracy i choć w minimalnym stopniu przedstawić wady i zalety popularnych metodyk realizacji projektów. Pojedyncze zlecenia można ze sobą grupować tworząc całe scenariusze. Gracz podejmujący rozgrywkę według danego scenariusza będzie otrzymywał kolejne projekty do realizacji wedle określonego schematu. Dzięki wygenerowaniu scenariuszy daje się możliwość wielu graczom stawienia czoła takim samym wyzwaniom. Nanoszenie tego typu elementów odbywa się metodą „drag-and-drop” z zamieszczonych w tabelach rekordów na strukturę drzewa, w którym głównym korzeniem jest cały scenariusz, gałęziami są kolejne zlecenia a liśćmi zdarzenia losowe.



Rysunek 5. Skomponowany przykładowy scenariusz rozgrywki

Podczas rozgrywki moderator może generować utrudnienia jako zdarzenia imitujące prawdziwe zdarzenia losowe, jak na przykład odejście pracownika czy zadanie serwisowe. O ile podczas gry „na żywo” moderator może śledzić realizowane przez graczy zlecenia

i generować utrudnienia dla każdego użytkownika, to w przypadku scenariusza utrudnienia takie są przydzielane w określonym czasie względem przewidzianej długości projektu.



Rysunek 6. Generowanie utrudnień podczas rozgrywki

Kolejnym zadaniem moderatora jest kreowanie nowych postaci, jako potencjalnych pracowników. Każdy z nich ma pewne umiejętności, które ulegają rozwojowi wraz z przebiegiem gry i w zależności od angażowania danego bohatera w realizację zadań. Od poziomu tych danych zależy koszt jednostki pracy na konkretnym stanowisku. Po pewnym czasie może wystąpić sytuacja, gdzie wszystkie postacie osiągną maksimum wiedzy i gra stanie się monotonna lub pracownicy zbyt drodzy w utrzymaniu. Dlatego zadaniem moderatora jest dodawanie nowych bohaterów, którzy będą mogli się rozwijać, lub edycję danych już istniejących by sztucznie zmienić ich poziomy zaawansowania.

W celu oceny wyników graczy i porównaniu ich osiągnięć moderator ma możliwość podglądu danych rankingowych zarówno dla rozgrywki pojedynczej jak i rywalizacji scenariuszy. Dzięki temu jeśli aplikacja zostałaby wykorzystana jako platforma szkoleniowa, administrator ma możliwość sprawdzenia, którzy uczestnicy uczynili największe postępy.

5. Weryfikacja metody i podsumowanie

Opracowany prototyp aplikacja jest próbą przeniesienia klasycznej gry symulacyjnej w przestrzeń wirtualną. W stworzonym rozwiązaniu gracz ma przede wszystkim zarządzać zespołem. Rozwijanie elementów sztucznej inteligencji i zaimplementowanie ich do działań podwładnych mogłoby znacznie zwiększyć atrakcyjność gry. W klasycznych rozwiązaniach występuje dość precyzyjny podział na role i odpowiedzialności poszczególnych osób pracujących nad projektem. Dla potrzeb gry, schemat ten został uproszczony i kompetencje menedżera zostały nieco zmienione. Zadania raportowania zostały zminimalizowania do określania kamieni milowych. Gracz nie ma fizycznej możliwości tak dogłębnego wcielenia się w swojego bohatera by móc dokładnie analizować powierzony projekt. Dlatego też proces wstępnego szacowania pracochłonności czy określenie terminu realizacji dokonywany jest przez moderatora i z góry narzucony dla każdego zlecenia. Za każde wykonane przez menedżera zadanie przyznawany był czas bonusowy lub karny. W zaimplementowanej grze czas jest wartością, która jest niezależna od naszych poczynań natomiast gracz rozlicza się według przyznanego mu budżetu pieniężnego. Elementem wprowadzonym jest liczba punktów nadawanych za wynik całkowity zakończonego projektu. W przypadku rozbudowy aplikacji i poszerzaniu kompetencji gracza na przykład o dodatkowe raportowanie można wprowadzić dodatkowe elementy związane z upływem czasu – jak punkty za terminowość zadań pomiędzy kamieniami milowymi czy przygotowywanie podań o opóźniania terminu realizacji projektu.

Zbudowany prototyp wymaga dalszego rozwoju. Jego wykorzystanie w czasie zajęć szkoleniowych dowodzi już teraz, że zastosowanie gier symulacyjnych dedykowanych do projektów informatycznych w szkoleniu zespołów projektowych poprawia efektywność takich szkoleń i uatrakcyjnia je. Zastosowanie gier symulacyjnych pozwala też na prowadzenie szkoleń w środowiskach rozproszonych i jest przyszłością szkoleń e-learningowych.

Prototyp gry symulacyjnej wykorzystano do przeprowadzenia szkoleń z czterema grupami studenckimi z przedmiotu Zarządzanie projektem informatycznym. Po zajęciach przeprowadzono badanie jakościowe oceny formy przeprowadzonych zajęć, które ujęte jest w tabeli 1.

Tabela 1. Wyniki ankiety ze szkoleń z zastosowaniem gry symulacyjnej

Lp	Pytanie	Procent odpowiedzi pozytywnych
1	Czy wykorzystanie gry symulacyjnej uatrakcyjniło formę zajęć	85
2	Czy wykorzystanie gry symulacyjnej wpłynęło Twoim zdaniem na zrozumienie procesów zarządzania projektem	86
3	Czy wykorzystanie gry symulacyjnej pomogło Ci w nauce	84

Wyniki zebrane w tabeli 1 wyraźnie pokazują, że wprowadzenie do edukacji z zakresu zarzadzania projektami informatycznymi jest pozytywnie odbierane przez studentów i poprawia w ich odczuciu jakość procesu edukacji.

W tabeli 2 zebrano wyniki testu kompetencyjnego przeprowadzonego wśród studentów mających zajęcia z wykorzystaniem gier symulacyjnych w porównaniu do wyników tego samego testu wykonane przez studentów kursu tradycyjnego bez stosowania gier symulacyjnych.

Tabela 2. Ocena kompetencji po ukończeniu szkolenia

Metoda	Odsetek studentów z oceną niedostateczną	Odsetek studentów z oceną bardzo dobrą
Metoda tradycyjna	35%	15%
Metoda z grą symulacyjną	20%	40%

Zbudowany prototyp wymaga dalszego rozwoju. Jego wykorzystanie w czasie zajęć szkoleniowych dowodzi już teraz, że zastosowanie gier symulacyjnych dedykowanych do projektów informatycznych w szkoleniu zespołów projektowych poprawia efektywność takich szkoleń i uatrakcyjnia je. Zastosowanie gier symulacyjnych pozwala też na prowadzenie szkoleń w środowiskach rozproszonych i jest przyszłością szkoleń tradycyjnych i e-learningowych.

Bibliografia

[1] Adams E., *Projektowanie gier. Podstawy*. Wydanie II, Helion 2011
[2] McGuire M., Jenkins O. C., *Creating Games. Mechanics, content and technology*, AK-Peters 2009
[3] Cockburn A., *Agile Software Development. Gra zespołowa*, Wydanie II, Helion 2008
[4] Schmidt P., *Agile w TMP, Software Developer's Journal Extra*, nr 2/2011, str. 68-75
[5] Burdewick I. *Aspects Of Methodology And Education Psychology In Project-Oriented Studies*, International Workshop on Project Oriented Learning, March 2003, Hanzehogeschool Groningen

Zarządzanie projektem wdrożeniowym systemu klasy ERP – autorska metodyka

Jan Trąbka

Uniwersytet Ekonomiczny w Krakowie, Katedra Informatyki

Abstract:

In today's economic organizations very important element of infrastructure are the class ERP systems (Enterprice Resource Planning). Essential component becomes ability of choice and introduction of this equipment. Unfortunately the projects of implement of the class ERP systems are unsuccessful understood as introduction of lesser range than assumed or marked overspend of time and financial budget of this project. As a reason for this we can point out lack of use project's management methodology set on ERP systems. In this paper is proposed authorial management methodology of project of class ERP systems implement. It allows to conduct the project in a frame of complex budget and get appropriate quality of the final product.We described source of introduced method and elements of creation of projects range or time and financial budget.

Keywords:

metodyka zarządzania projektem, systemy ERP, proces biznesowy, zdyscyplinowane metodyki zarządzania projektem, zwinne metodyki zarządzania projektem

1. Wprowadzenie

W dzisiejszych organizacjach gospodarczych bardzo istotnym elementem infrastruktury IT są systemy klasy ERP (Enterprise Resource Planning). Wdrożenia systemów klasy ERP stanowią, w chwili obecnej, duży procent wszystkich prowadzonych w przedsiębiorstwach projektów informatycznych. Obecnie bardzo niewiele średnich i dużych przedsiębiorstw może funkcjonować bez wspomnianego narzędzia informatycznego. W związku z tym ważnym elementem staje się umiejętność wyboru i wdrożenia ww. narzędzi. Niestety bardzo często projekty wdrożenia systemów klasy ERP kończą się porażką, rozumianą jako wdrożenie mniejszego niż zakładano zakresu docelowego systemu, czy znaczne przekroczenie budżetu czasowego i finansowego projektu. Praktyka wskazuje również na przypadki całkowicie nieudanych prób wdrożenia. W przyczynach tego stanu rzeczy jednym z ważnych czynników jest niestosowanie lub niekonsekwentne stosowanie metodyk zarządzania projektami. Pośrednią przyczyną jest brak metodyk zarządzania projektami uwzględniających specyfikę systemów ERP. Największe firmy specjalizujące się w produkcji systemów tego typu mają wypracowane takie metodyki (tzw. firmowe) np. firma SAP z metodyką ASAP. Dostawcy z mniejszym potencjałem i doświadczeniem nie stosują żadnych metodyk lub wykorzystują metodyki klasyczne – opisane w literaturze (Prince2, PMI). System klasy ERP jest specyficznym produktem, gdyż z jednej strony może być potraktowany jako gotowy do sprzedaży, dla klientów, którzy nie mają specyficznych wymagań lub mają niewielką skalę działania. Z drugiej strony specyfika wymagań odbiorcy, szczególnie średnich i dużych przedsiębiorstw, najczęściej wymaga wielkiego wysiłku i czasu, aby dostosować system do wymaganej specyfiki biznesu. W takim ujęciu system ERP jest platformą, na której tworzy

się „szyte na miarę" narzędzie realizujące wymagania klienta. Wówczas należy powołać projekt z całą jego infrastrukturą i zasobami. Najczęściej mamy do czynienia z etapami: analizą wymagań, implementacją wymagań (konfiguracją, kastomizacją) oraz szkoleniami. Konfiguracja i kastomizacja w projekcie wdrożeniowym kosztuje czasami tyle samo lub więcej niż licencje na kupowany system ERP. Dostępne na rynku systemy ERP różnią się między sobą nie tylko wbudowanymi, gotowymi funkcjonalnościami, ale możliwością dostosowania ich do procesów biznesowych odbiorcy (np. dedykowanymi, wbudowanymi narzędziami developerskim). W opisywanej w niniejszym referacie metodyce słowem kluczem będzie określenie - proces biznesowy, które będzie wykorzystywane w całym cyklu wdrożenia systemu ERP. W referacie zaproponowano autorską metodykę zarządzania projektem wdrożenia systemu klasy ERP, która może pozwolić na prowadzenie projektu w ramach założonego budżetu oraz uzyskanie odpowiedniej jakości produktu finalnego. Zaproponowana metodyka będzie bazować na klasycznych, opisanych w literaturze metodykach zarządzania projektami oraz stworzonej na Politechnice Poznańskiej metodyce Xprince. Do założeń tej ostatniej dodane pozostaną elementy związane ze specyfiką systemu ERP.

2. Źródła metodyki ERPrince

2.1. Klasyczne metodyki zarządzania projektami informatycznymi

Metodyki zarządzania projektami informatycznymi można podzielić na dwa główne typy: zdyscyplinowane (ciężkie) oraz zwinne (lekkie) [2]. W metodykach zdyscyplinowanych (np. Prince2) projekt ma bardzo wyraźnie określone etapy, które realizowane są kaskadowo (analiza, projektowanie, implementacja). Produkt danego etapu oddawany jest po jego zakończeniu. Zespół projektowy zbudowany jest z kilkunastu ról w kilkupoziomowej hierarchii. Metodyki zdyscyplinowane wymagają prowadzenia bardzo rozbudowanej dokumentacji projektowej [3]. W metodykach zwinnych (RUP for small project, XP, Scrum) produkt dostarczany jest w tzw. iteracjach. Każda iteracja to przejście całego cyklu wytwórczego (analiza, projekt, implementacja) w odniesieniu do małych fragmentów dostarczanego systemu. Iteracje to okresy 1-2 tygodniowe. Takie podejście redukuje czynności zarządcze oraz dokumentację projektową. Zespół projektowy jest podzielony na kilkuosobowe grupy składające się z przedstawicieli odbiorcy oraz developerów, projektantów i programistów. W podejściu zwinnym odbiorca systemu na bieżąco śledzi i kontroluje proces tworzenia. Obydwa rodzaje metodyk projektowych mają swoje mocne i słabe strony, które zaprezentowano w tabeli nr 1.

Tabela 1. Mocne i słabe strony zdyscyplinowanych i zwinnych metodyk zarządzania projektami

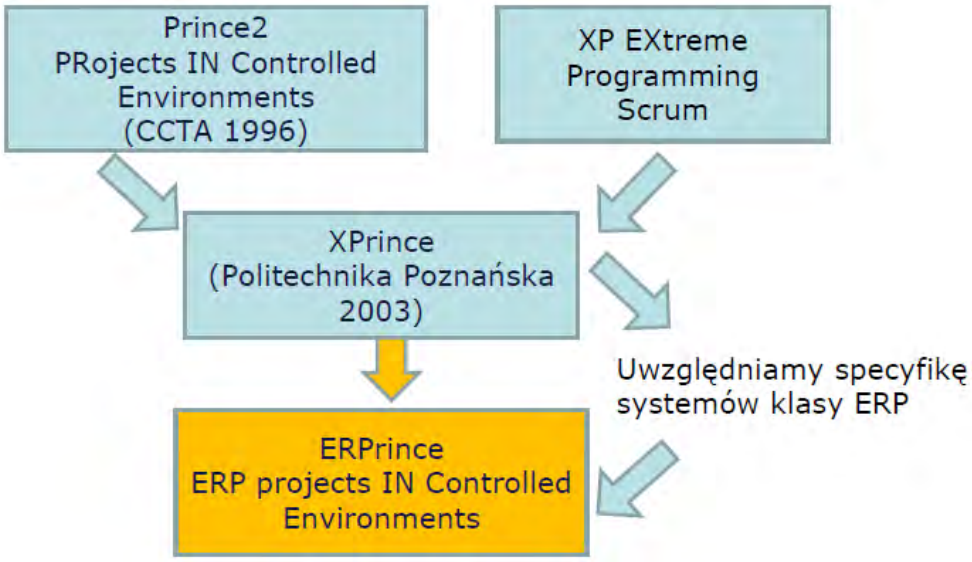
Metody zdyscyplinowane	**Metody zwinne**
Mocne strony	
podejście procesowe [3]	elastyczność - iteracyjna implementacja wymagań z możliwością ciągłego doprecyzowania
budowa budżetu oraz mechanizmów kontrolnych w oparciu o produkty projektowe	bieżąca współpraca z Klientem, możliwość wczesnego wyłapania błędów
pełna (długa) perspektywa planu	
precyzja szacowania	
Słabe strony	
nadmierna ilość dokumentów (papierologia)	niewystarczające zasoby i zaangażowanie po stronie Klienta
powolne procesy decyzyjne	brak spisanej dokumentacji

niezdolność do dostosowania się do zmian w trakcie trwania projektu	zbyt krótka perspektywa planu (najbliższe iteracje)
mała elastyczność	trudności z oszacowaniem kosztów

Źródło: Opracowanie własne na podstawie [4]

2.2. Metodyka Xprince

Metodyką łączącą zalety podejścia zdyscyplinowanego oraz zwinnego jest opisana poniżej metodyka Xprince. Metodykę tę stworzono na Politechnice Poznańskiej przez zespół Prof. J. Nawrockiego w roku 2003.



Rysunek 1. Źródła metodyki Xprince oraz ERPrince
Opracowanie własne

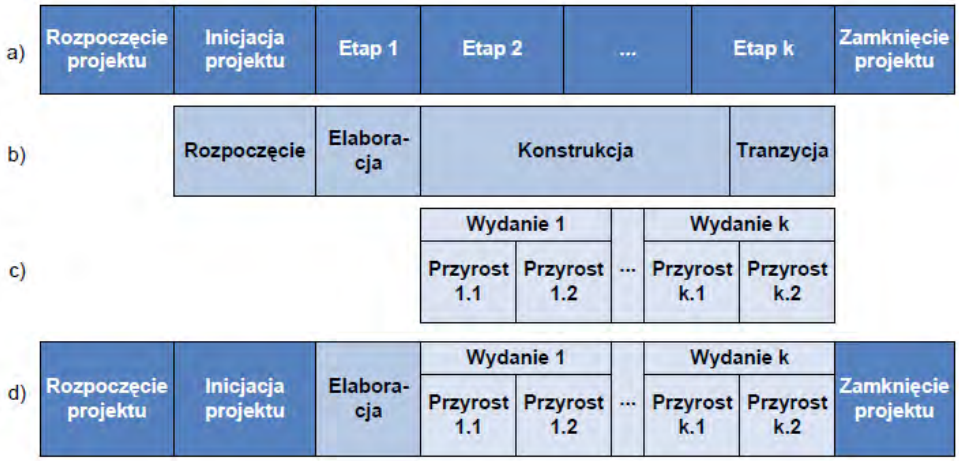
Xprince bazuje na trzech metodykach: XP (Extreme Programming), Prince2 oraz RUP [4]. Aby nie powtarzać opisu poszczególnych elementów metodyki (będą opisane przy omawianiu metodyki ERPrince), w tym miejscu, tylko skrótowo omówiono budowę i role zespołu projektowego oraz cykl życia projektu, ze wskazaniem, że Xprince jest zbilansowanym konglomeratem wskazanych powyżej metodyk.

Zespół projektowy w metodyce Xprince

W metodykach zwinnych struktury zespołów projektowych są płaskie i sprowadzają się do kilkuosobowych zespołów wytwórczy. Zespoły wytwórcze, w kolejnych iteracjach tworzą (realizując cykl: analiza, projektowanie, implementacja) gotowe fragmenty systemu. Stworzone i zintegrowane fragmenty składają się na kompletne rozwiązanie. W metodyce Prince2 zespół projektowy jest kilkustopniową hierarchią, z bardzo rozbudowanym Komitetem Sterującym. Za operacyjne zarządzanie projektem odpowiada rola Kierownika Projektu. Metodyka Xprince przejmuje zdyscyplinowane struktury zarządcze projektu natomiast w warstwie wykonawczej pozostają kilkuosobowe zespoły wytwórcze, występujące w podejściu zwinnym.

Cykl życia projektu w metodyce Xprince

W kontekście etapów prowadzenia projektu widać wyraźnie, że ich struktura została przejęta z metodyki Prince2. Elementem odziedziczonym po metodykach zwinnych jest etap wykonawczy podzielony na wydania czyli iteracje, w których klient otrzymuje gotowy fragment systemu. W zwinnych metodykach w fazie wytwórczej wykorzystuje się prototypowanie.



Rysunek 2. Cykl życia projektu w metodykach źródłowych: a. Prince2, b. RUP, c.XP, d. Xprince
Źródło: [4]

3. Metodyka ERPrince

3.1. Założenia ogólne

Na potrzeby zaproponowanej w niniejszym referacie metodyki do założeń metodyki Xprince dodane zostaną elementy charakterystyczne dla wdrożeń systemów klasy ERP. Powstający zestaw założeń utworzy podstawową strukturę pojęciową metodyki nazwanej przez autora ERPrince. ERPrince jest metodyką stworzoną na czterech filarach:

- międzynarodowym standardzie zarządzania projektami - metodyce PRINCE2,
- zwinnych metodykach zarządzania projektami m.in. XP, RUP for small project
- metodyce Xprince będącej konglomeratem metodyk zdyscyplinowanych i zwinnych,
- specyfice przedmiotu wdrożenia, a więc systemie klasy ERP.

W tym miejscu należy zdefiniować najważniejsze pojęcie omawianej metodyki – proces biznesowy. Proces biznesowy to sekwencja uporządkowanych czynności, których celem jest wytworzenie jakiegoś dobra (informacji, towaru, usługi itd.)[5]. Procesy dzielą się na podprocesy. Dekompozycja doprowadza do procesów elementarnych, a więc takich które nie wymagają dalszego podziału - na tym poziomie możemy je dokładnie wyspecyfikować. Każdy proces elementarny stanowi prostą i niepodzielną (w konkretnym modelu procesów) czynność np. wydruk faktury. Na poziomie procesu elementarnego można projektować wsparcie jego realizacji przez system komputerowy. Najczęściej we wsparciu systemowym procesu elementarnego bierze udział od jednej do kilku funkcjonalności systemu np. do wydruku faktury jedna, do przyjęcia towaru do magazynu już kilka. W metodyce ERPrince poprawnie zaimplementowany proces elementarny jest produktem projektownym. Aby produkt projektowy można było uznać za dostarczony musi przejść przez zespół testujący po

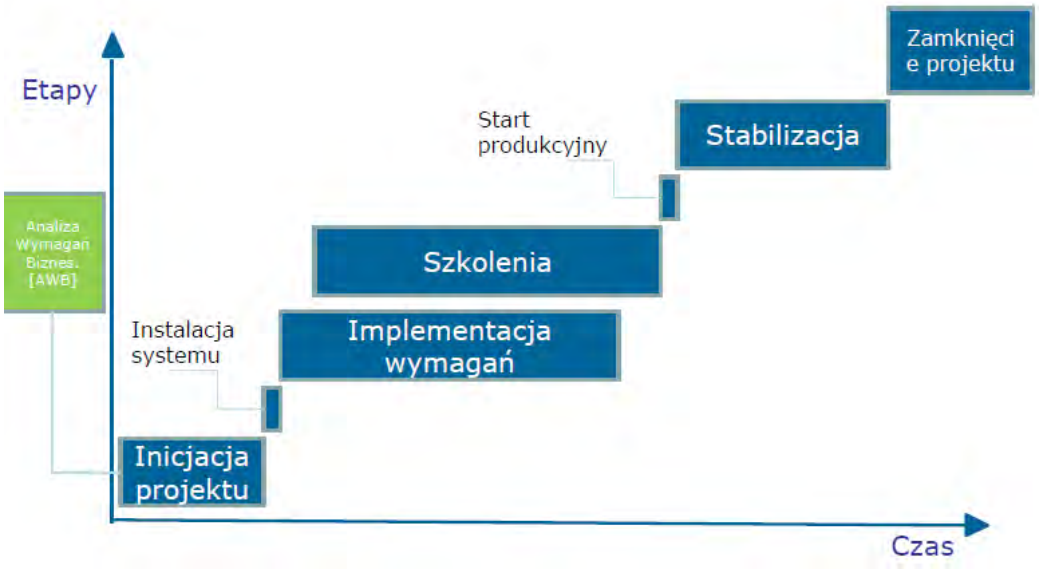
stronie odbiorcy systemu. W kontekście tak zdefiniowanego produktu projektowego dokonuje się szacowania, harmonogramowania oraz wyceny całości projektu.

ERPrince oparto na trzech podstawowych zasadach:

- etapowym podejściu do planowania i kontroli projektu (zasada zaczerpnięta ze zdyscyplinowanych metodyk zarządzania projektem). Bardzo istotnym elementem jest zaplanowanie wszystkich etapów projektu, a więc w perspektywie kilku, kilkunastu miesięcy. Fazę implementacji planuje się rozpisując, które procesy biznesowe będą wdrażane w poszczególnych iteracjach. Takie podejście daje odpowiedź kiedy będą zakończone poszczególne etapy, czego nie można określić wykorzystując tylko zwinne metodyki projektowe,
- kontroli postępu prac w oparciu o produkty projektowe (zasada zaczerpnięta z metodyki Prince2). Produkty projektowe tworzone są z procesów elementarnych opisanych na etapie Analizy Wymagań. Uznaje się, iż produkt projektowy jest dostarczony, gdy proces elementarny jest poprawnie zaimplementowany w systemie i zaakceptowany przez odbiorcę,
- prototypowaniu w fazie implementacji wymagań użytkownika (zasada zaczerpnięta ze zwinnych metodyk tworzenia systemów informatycznych). Małe zespoły projektowe tworzą w ramach iteracji obsługę procesu bądź procesów. Developerzy (w ramach jednej iteracji) przekazują przygotowany fragment systemu (prototyp) do testów przez przedstawicieli odbiorcy. Ci ostatni testują i akceptują bądź odrzucają prototyp z uwagami. W takim trybie po ostatecznej akceptacji oddawany jest gotowy fragment systemu.

W dalszej części opisano szczegółowo cztery aspekty metodyki ERPrince: etapy prowadzenia projektu, zasady weryfikacji postępu prac, budowy harmonogramu oraz budowy zespołu projektowego.

3.2. Etapy realizacji projektu oraz zasady weryfikacji postępu prac



Rysunek 3. Etapy realizacji projektu według metodyki ERPrince
Źródło: Opracowanie własne

Etapy prowadzenia projektu:

- analiza wymagań biznesowych – etap rozpoczyna się od ustanowienia wspólnego zespołu projektowego, omówienia wykorzystywanych narzędzi i notacji oraz ustalenia planu i terminarza całego etapu. W trakcie analizy specyfikowane są procesy będące w zakresie projektu. Punktem wyjścia do stworzenia hierarchicznej listy procesów jest mapa procesów najwyższego poziomu (obsługa sprzedaży produktów, obsługa zaopatrzenia itd.). Następnie procesy główne dekomponuje się do poziomu procesów elementarnych, reprezentujących proste czynności, odbywające się w jednym momencie np. wydruk faktury. Proces elementarny obsługiwany jest w docelowym rozwiązaniu najczęściej przez pojedynczą funkcjonalność systemu. Procesy elementarne specyfikuje się poprzez opis kontekstu realizacji, procedury wykonania, dodatkowych wymagań użytkownika, struktur danych, wzorców dokumentów i raportów oraz praw dostępu i realizacji. Dodatkowymi elementami analizy są: słowniki pojęć, słowniki danych oraz uszczegółowiony harmonogram projektu.
- inicjacja projektu – w trakcie realizacji tego etapu potwierdza się skład i odpowiedzialności zespołu projektowego, ostatecznie potwierdza harmonogram projektu, ustala zasady kontroli i komunikacji.
- implementacja wymagań – to najdłuższy etap projektu, realizowany jest prawie równocześnie z etapem szkoleń. Układ taki wynika z prototypowego podejścia do implementacji rozwiązania. Podejście to polega na tworzeniu prototypów funkcjonalności obsługujących pojedyncze procesy elementarne wyspecyfikowane na etapie analizy wymagań biznesowych. Stworzony prototyp zostaje przekazany Kluczowym Użytkownikom (patrz następny podrozdział) do testów. W tym właśnie momencie następuje równoległe uruchomienie etapu szkolenia, gdyż Kluczowi Użytkownicy muszą poznać podstawowe zasady działania oraz interfejs systemu, aby prawidłowo przetestować prototypy funkcjonalności. Uwagi Kluczowych Użytkowników są na bieżąco przekazywane do developerów. Po wprowadzeniu poprawek następuje ponowne testowanie i taka iteracja powtarza się, aż do momentu ostatecznej akceptacji prototypu, który staje się gotową częścią nowego rozwiązania. Pojedynczy, zaimplementowany proces elementarny (po ostatecznym odbiorze) jest dostarczonym produktem projektowym. Na takim poziomie w metodyce ERPrince buduje się mechanizm weryfikacji postępu prac projektowych. Pełny cykl wytworzenia produktu projektowego powinien się zamknąć w dwóch tygodniach. Cykliczna (najczęściej cotygodniowa) kontrola projektu raportuje produkty projektowe zakończone, te, nad którymi aktualnie pracuje zespół projektowy (w tym miejscu raportuje się problemy i ryzyka) oraz potwierdza, które produkty projektowe będą realizowane w ciągu kolejnego okresu kontrolnego. Z podejścia prototypowego wynika również instalacja systemu poprzedzająca omawiany etap implementacji wymagań (patrz rysunek powyżej).
- szkolenia – etap ten można podzielić na dwie części. Pierwsza to szkolenia dla Kluczowych Użytkowników, które odbywają się podczas prototypowej implementacji rozwiązania. Druga część to szkolenia dla pozostałych użytkowników, które to odbywają się już po akceptacji całości rozwiązania.
- start produkcyjny – to najistotniejszy etap projektu, od tego momentu użytkownicy rozpoczynają wykorzystywanie systemu w warunkach bieżącej działalności.
- stabilizacja – etap projektu, w którym developerzy wspomagają użytkowników w pierwszym okresie pracy na nowym systemie.
- zamknięcie projektu – w etapie tym dokonuje się rozliczenia projektu. Od tego momentu rozpoczyna się etap utrzymania systemu.

3.3. Budowa harmonogramu projektu

W przygotowaniu harmonogramu, w szczególności, etapu implementacji podstawą jednostką planowania są opisane w analizie wymagań procesy elementarne. Dla przejrzystości procesy elementarne grupuje się do poziomu procesów głównych (patrz Tabela 2). Dla każdego procesu elementarnego planuje się pełny cykl przygotowania i przekazania prototypu. Cykl ten zawiera trzy zadania:

- przygotowania prototypu – developerzy na bazie dokumentu analitycznego przygotowują pierwszy prototyp,
- przekazania prototypu – developerzy prezentują Kluczowym Użytkownikom przygotowany prototyp
- weryfikacja i oddanie prototypu – Kluczowi Użytkownicy przekazują uwagi z testu prototypu. Uwagi zostają zaimplementowane przez developerów. Zamknięcie całego cyklu następuje po ostatecznym testowaniu wykonanym przez Kluczowych Użytkowników.

Tabela 2. Harmonogram projektu według metodyki ERPrince

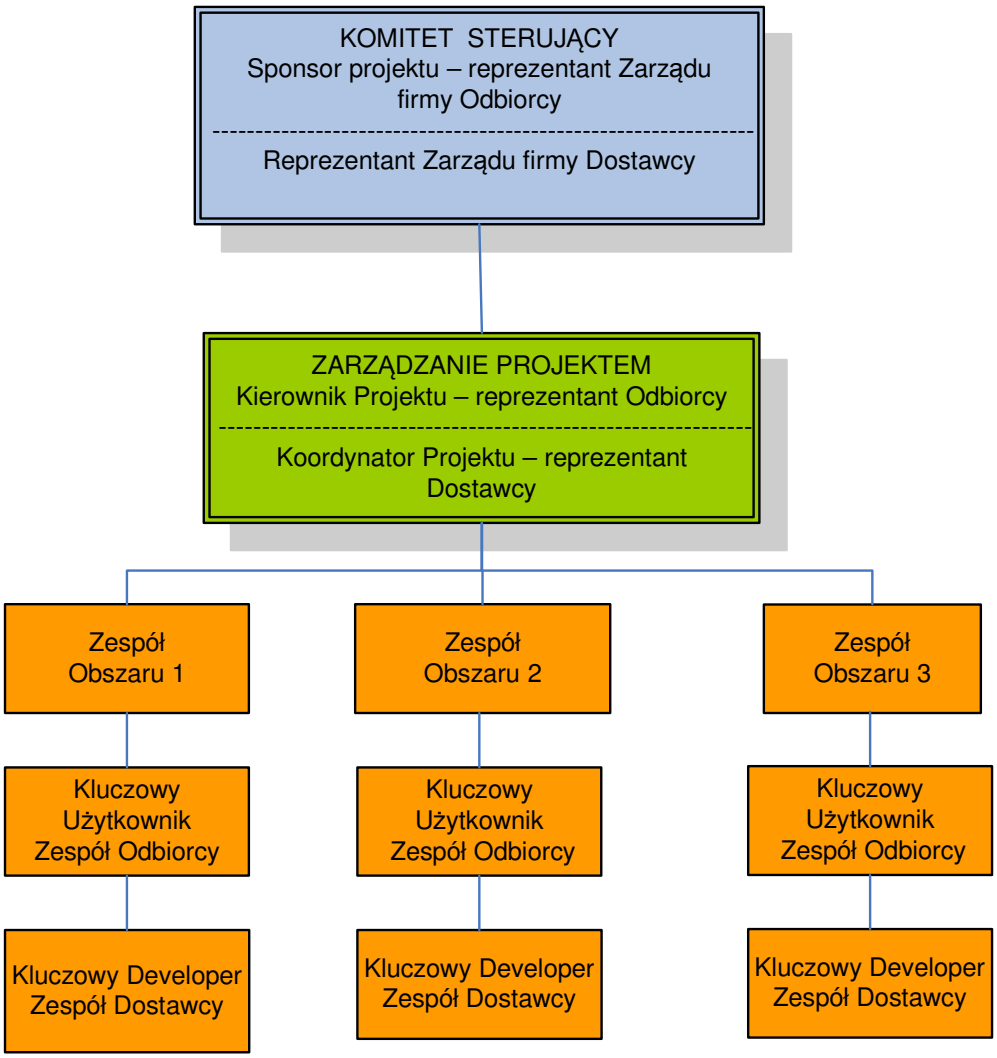
Numer zadania	Zadanie/etap	Czas Dostawcy (rbd)	Czas Odbiorcy (rbd)	Data Rozp.	Data Zak.	Komentarz
1.	Proces nr 5. Logistyka serwisowa					Proces główny
1.1.	Proces 5.1. Przyjęcie zapotrzebowania					Pierwszy proces elementarny
1.1.1.	Przygotowanie prototypu	2	0	2012-04-24	2012-04-26	Przygotowanie prototypu na bazie Analizy Wymagań
1.1.2.	Przekazanie prototypu	1	1	2012-04-27	2012-04-27	Szkolenie dla Kluczowych Użytkowników
1.1.3.	Weryfikacja i oddanie prototypu	2	3	2012-04-28	2012-05-04	Uwagi Kluczowych użytkowników
1.2.	Proces 5.2. Zamówienie do dostawcy					Powtórzenie całego cyklu

Źródło: Opracowanie własne

Najtrudniejszym i najbardziej ryzykownym w harmonogramowaniu jest szacowanie zadania weryfikacji i oddania prototypu, zależy ono od skomplikowania danego procesu elementarnego ale również kompetencji i zaangażowania Kluczowego Użytkownika. W harmonogramie ww. zadanie ma zazwyczaj największy budżet czasowy. Aby zapobiec niedoszacowaniu czas realizacji tego zadania można wyliczać na bazie wdrożeń podobnych procesów elementarnych w innych przedsiębiorstwach.

3.4. Budowa i odpowiedzialności zespołu projektowego

Budowę zespołu projektowego, zgodnego z metodyką ERPrince przedstawiono na rysunku nr 4.



Rysunek 4. Struktura zespołu projektowego
Źródło: Opracowanie własne

Odpowiedzialność głównych ról projektowych w metodyce ERPrince:

- Komitet Sterujący - jest organem reprezentującym najwyższe kierownictwo firm odbiorcy i dostawcy, delegowanym do nadzoru nad realizacją projektu wdrożenia systemu ERP. Komitet jest odpowiedzialny za długoterminowe zarządzanie projektem oraz monitoring. Oznacza to kontrolę realizacji projektu na poziomie strategicznym, weryfikację zgodności projektu z przyjętymi wcześniej celami i utrzymanie założonych ram: zakresu, kosztów i terminów czasowych. Zmiany w projekcie wymagają akceptacji Komitetu Sterującego.
- Kierownik Projektu – odpowiada za nadzór oraz koordynację czynności prowadzonych przez jednostki zaangażowane w projekt po stronie odbiorcy. Kierownik Projektu odpowiada za komunikację wewnątrz zespołu projektowego oraz z Komitetem Sterującym. Kierownik Projektu wraz z Koordynatorem podejmuje decyzje operacyjne projektu,
- Koordynator Projektu – odpowiada za nadzór oraz koordynację czynności prowadzonych przez jednostki zaangażowane w projekt po stronie dostawcy. Koordynator wraz z Kierownikiem Projektu podejmują decyzje operacyjne projektu,

- Kluczowi Użytkownicy – zespół specjalistów dziedzinowych odbiorcy zaangażowany we wszystkie etapy projektu oraz odpowiedzialnych za weryfikację wszystkich wdrożonych rozwiązań (produktów projektowych),
- Kluczowi Developerzy – wdrożeniowcy o dużym doświadczeniu w realizacji wdrożeń w danym obszarze funkcjonalnym. Odpowiadają za tworzenie i dostarczanie produktów projektowych.

Rola zespołu odbiorcy

W metodyce ERPrince ogromną rolę odgrywa zaangażowanie i kompetencje zespołu projektowego. Szczególny nacisk położony jest na stronę odbiorcy rozwiązania. We wszystkich opisanych wyżej etapach Kluczowymi Użytkownicy odgrywają aktywną i ważną rolę:

- w analizie wymagań przedstawiają jak obecnie działa ich organizacja oraz jak chcieliby, aby działała. Następnie weryfikują stworzoną i opisaną przez zespół projektowy dostawcy wizję rozwiązania,
- podczas implementacji odbierają prototypy, zgłaszają uwagi oraz proponują własne rozwiązania. Bez akceptacji Kluczowych Użytkowników prototyp nie może stać się częścią docelowego rozwiązania,
- podczas szkoleń niejednokrotnie występują w charakterze prowadzących - instruktorów, gdyż po implementacji znają już przygotowane rozwiązania a dodatkowo znają środowisko organizacji, reguły biznesowe oraz same osoby szkolone, co bardzo ułatwia komunikację i odbiór przekazywanego materiału.

Aktywny udział użytkowników końcowych w realizacji projektu jest bardzo charakterystyczny dla zwinnych metodyk tworzenia systemów informatycznych. Z tym elementem wiąże się jednak ogromne ryzyko projektowe. W sytuacji, gdy odbiorca nie zapewni wystarczających zasobów, nie będą one kompetentne lub nie będą posiadały umocowań decyzyjnych w zakresie nowego narzędzia oraz nowo wdrażanych procedur, osiągnięcie założonych celów projektu może być niemożliwe. Również istotne, znaczenie dla powodzenia projektu posiada zespół projektowy dostawcy. Kluczowy developer systemu jest, w jednej osobie: analitykiem, projektantem i programistą, dlatego też musi to być pracownik o dużej wiedzy i doświadczeniu biznesowym w obszarach, w których się specjalizuje, jak również o dużych kompetencjach informatycznych.

4. Podsumowanie

W obecnej chwili projekty wdrożeniowe systemów klasy ERP stanowią duży procent pośród wszystkich przedsięwzięć informatycznych w średnich i dużych przedsiębiorstwach. Niestety widać również duży procent wdrożeń nieudanych. Jednym z powodów i jednocześnie główną tezą opracowania jest stwierdzenie, że bardzo niewiele metodyk zarządzania projektem informatycznych uwzględniających specyfikę wdrożeń systemów ERP. W niniejszym opracowaniu pokazano próbę stworzenia takiej metodyki. Na bazie ogólnej metodyki zarządzania projektami wdrożeniowymi Xprince oraz specyfiki produktu jakim jest system ERP, zapronowano metodykę ERPrince. Głównym elementem charakteryzującym tę metodykę jest wskazanie na proces biznesowy, jako podstawowy element budowy zakresu, harmonogramu, wyceny oraz kontroli projektu. Prace nad pokazaną metodyką będą podążać w kierunku ustalenia bardziej precyzyjnych metod szacowania czasu realizacji poszczególnych zadań oraz związanych z tym kosztów realizacji. Wymienione elementy zostaną omówione w następnych pracach autora.

Bibliografia

[1] Flasiński M. *Zarządzanie projektami informatycznym*,. PWN, 2006
[2] Wrycza S., *Analizowanie i projektowanie systemów informatycznych zarządzania*, PWN, 1999
[3] *Prince2 – Skuteczne zarządzanie projektami*, Crown Copyright, 2009
[4] Nawrocki J., Olek Ł., Jasiński M., Paliświat B., Walter B.,Pietrzak N., Godek P, *Równowaga między zwinnością a dyscypliną z wykorzystaniem Xprince*, www.inmost.org.pl/articles/Rawnowaga_midzy_zwinnoAci_a_dyscyplin,
[5] Drelewicz S., *Zrozumieć BPMN.Modelowanie procesów biznesowych*, Helion, 2012

Tworzenie języków specyfikacji dla zagadnień numerycznych

Andrzej Grosser[1], Norbert Sczygiol[2]

[1]*Politechnika Częstochowska, Wydział Inżynierii Mechanicznej i Informatyki*
[2]*Politechnika Częstochowska, Wydział Inżynierii Mechanicznej i Informatyki*

Abstract:

Artykuł jest poświęcony zagadnieniu tworzenia języków specyfikacji dziedzinowej dla metod numerycznych. Pokazuje autorskie środowisko wspierające tworzenie takich języków ze szczególnym uwzględnieniem parserów i szablonów kodu. Opisano także wszystkie etapy budowy typowego, zewnętrznego języka specyfikacji. Zaprezentowano również przykładowy język specyfikacji Finele, utworzony w wcześniej wymienionym środowisku, opisujący problemy rozwiązywane za pomocą metody elementów skończonych.

Keywords:

tworzenie języków specyfikacji, generowanie kodu, gramatyki wyrażeń parsujących, metoda elementów skończonych

1. Wprowadzenie

Tworzenie poprawnie działającego oprogramowania wymaga przede wszystkim opanowania trudności związanych ze złożonością rozwiązywanego problemu. Narzuca to konieczność dogłębnego zrozumienia często nie do końca jasnej dziedziny oraz dopasowania jej do możliwości oferowanych przez sprzęt, środowisko programowania (w szerokim rozumieniu - system operacyjny, biblioteki, zastosowane języki programowania itd.), a także będącej do dyspozycji kadry. Ten proces przekładu powoduje powstawanie różnego rodzaju błędów związanych zarówno z projektem, jak i implementacją.

Liczba błędów logicznych rośnie wraz ze wzrostem liczby detali, o jakich musi pamiętać programista, w szczególności tych związanych ze sprzętem i systemem operacyjnym na jakich ma działać powstająca aplikacja. Zarządzanie pamięcią, sposób wykorzystania rejestrów mikroprocesora, mechanizmy dostępu do plików potrafią skutecznie zaciemnić rozwiązywany problem.

Rozwój narzędzi i metod programowania doprowadził więc do ukrycia większości niskopoziomowych mechanizmów poprzez wprowadzanie kolejnych warstw abstrakcji wyższego rzędu. Programowanie za pomocą kodów bitowych w języku maszynowym zastąpiła, zastosowana w asemblerach, koncepcja zapisu instrukcji procesora w postaci skrótów mnemonicznych. Następnie pojawiły się języki wysokiego poziomu wyprowadzające zrozumiałe dla człowieka instrukcje, tłumaczone potem przez kompilatory na słowa maszynowe. Kolejnym krokiem było wsparcie przez języki programowania strukturalnego, a następnie obiektowego.

Ułatwiło to programowanie w porównaniu do lat wcześniejszych, jednakże nadal należy przyjęte rozwiązanie problemu dopasować do sztywnych często wymogów języka programowania. Jednocześnie wraz ze wzrostem możliwości kompilowanych języków programowania

wzrosła również ich złożoność. Widać to na przykładzie języka C++ - efektywne korzystanie z wzorców (element języka - ang. *template*), zrozumienie zasad powstawania obiektów tymczasowych oraz posługiwanie się przejętymi z języka C wskaźnikami, wymaga sporego nakładu czasu i dużego wysiłku.

Dlatego też pojawiła się, między innymi, koncepcja uproszczenia przekładu problemu na zapis komputerowy poprzez tworzenie języków, których notacja oraz stopień abstrakcji są skoncentrowane na wyróżnionej dziedzinie zastosowań [1]. Są to tzw. języki specyfikacji dziedzinowej (ang. *domain specific languages*). Dzięki zapisowi specyfikacji wymagań z wykorzystaniem słownictwa oraz idei stosowanych w określonej dziedzinie problemowej możliwe jest doprecyzowanie rozwiązywanego problemu.

Najważniejszy podział tych języków jest przeprowadzony ze względu na sposób ich tworzenia. Języki dzieli się na zewnętrzne i wewnętrzne [2]. Języki wewnętrzne (zwane również osadzonymi) są przygotowane z wykorzystaniem infrastruktury tzw. języka hosta. Składnia, jaką można zaoferować dla tak zbudowanego języka, ogranicza się do możliwości jakie oferuje język hosta. Możliwe jest jednak wykorzystanie przy ich budowie narzędzi opracowanych dla języka, w którym dokonuje się implementacji. W przeciwieństwie do nich języki zewnętrzne (samodzielne) są zazwyczaj implementowane z wykorzystaniem zewnętrznych narzędzi, takich jak generatory analizatorów leksykalnych, parserów, interpreterów. Proces ten można przyrównać do implementacji języka programowania ogólnego przeznaczenia. Wymagane jest więc nie tylko doświadczenie w zakresu rozpatrywanej dziedziny, ale także i wiedza z zakresu praktycznej i teoretycznej realizacji języków programowania. Daje to jednak szereg korzyści, przede wszystkim umożliwia pełne dostosowanie składni do wymagań użytkownika, stąd też wynika ich wybór jako podstawy opisywanych rozwiązań.

Ważnym problemem w budowie zewnętrznych języków specyfikacji jest niewielki wybór zintegrowanych narzędzi wspierających ten proces. Przedstawione w artykule środowisko ułatwia implementację języków specyfikacji z uwzględnieniem zastosowań w dziedzinie metod numerycznych. Wybór ten był związany z wysokimi wymaganiami dotyczącymi mocy obliczeniowych i pamięci operacyjnej, które dają możliwość przetestowania wydajności stworzonych rozwiązań. Ponadto istnieje bogata baza już istniejących programów i bibliotek pozwalających na weryfikację rozwiązań, a także na wykorzystanie części stworzonego oprogramowania do budowy generatorów kodu.

2. Środowisko tworzenia języków specyfikacji

Opisywane w artykule środowisko do tworzenia języków specyfikacji dziedzinowej składa się z rdzenia oraz elementów pomocniczych, wspierających pracę nad językiem. Rdzeń frameworku do budowy języków specyfikacji dziedzinowej jest zbudowany z autorskiego generatora parserów PegLi, służącego do opisu wzorców rozpoznających języki specyfikacji dziedzinowej, który bazuje na gramatykach wyrażeń parsujących (ang. *parsing expression grammar*) [3]. Ważną częścią jest też biblioteka szablonów kodu TempLi. Dodatkowo, pracę nad budową języka oraz szablonów kodu wspiera zintegrowane środowisko programowania. Ponieważ często jest tutaj potrzebne wyprowadzenie pewnych wielkości numerycznych (całek, wektorów, macierzy) za pomocą przekształceń symbolicznych, udostępniono także interfejs pozwalający na korzystanie z środowiska obliczeń symbolicznych Maxima [4].

Tabela 1. Działanie operatorów wyrażeń parsujących

Rodzaj	Rozpatrywany tekst	Konsumowany tekst	Dopasowanie
Symbol pusty ϵ	x	Żaden	Tak
Symbol terminalny a	ax	a	Tak
Symbol terminalny a	$bx\ (b \neq a)$	Żaden	Nie
Symbol nieterminalny A	$x\ (A \leftarrow e \wedge e \sim o)$	o	Tak
Sekwencja e_1e_2	$xy\ (e_1 \sim x \wedge e_2 \sim y)$	xy	Tak
Sekwencja e_1e_2	$xy\ (e_1 \nsim x \vee e_2 \nsim y)$	Żaden	Nie
Uporządkowany wybór e_1/e_2	$x\ (e_1 \sim x)$	x	Tak
Uporządkowany wybór e_1/e_2	$x\ (e_1 \nsim x \wedge e_2 \sim x)$	x	Tak
Uporządkowany wybór e_1/e_2	$x\ (e_1 \nsim x \wedge e_1 \nsim x)$	Żaden	Nie
Iteracja e^*(repetycja)	$x_1x_2\ (e \sim x_1 \wedge e^* \sim x_2)$	x_1x_2	Tak
Iteracja e^*(warunek końca)	$x\ (e \nsim x \wedge e^* \sim \epsilon)$	Żaden	Tak
Negacja $!e$	$y\ (e \nsim x)$	Żaden	Tak
Negacja $!e$	$y\ (e \sim x)$	Żaden	Nie

2.1. Generator parserów i biblioteka PegLi

Generator parserów PegLi (skrót od **P**arsing **e**xpression **g**rammar **Li**brary) został napisany przy użyciu języka programowania Python. Jako podstawę do opisu tworzonych języków specyfikacji wykorzystuje on, jak wcześniej wspomniano, wyrażenia parsujące. Formalnie gramatyką wyrażeń parsujących G [3] jest uporządkowana czwórka (V_N, V_T, R, e_s), gdzie: V_N jest skończonym zbiorem symboli nieterminalnych, V_T jest skończonym zbiorem symboli terminalnych, R jest skończonym zbiorem reguł, natomiast e_s jest wyróżnionym wyrażeniem parsującym, nazywanym wyrażeniem startowym. Zbiory V_N oraz V_T są zbiorami rozłącznymi. Każda reguła jest parą w postaci (A, e), zapisywaną zwykle w formie $A \leftarrow e$, gdzie A jest symbolem nieterminalnym, natomiast e jest wyrażeniem parsującym. Dla każdego symbolu A można zapisać dokładnie jedną regułę R. Wyrażenia parsujące definiuje się indukcyjnie w postaci (e, e_1 oraz e_2 są wyrażeniami parsującymi):

- ϵ - łańcuch pusty,
- $a \in V_T$ - dowolny terminal,
- $A \in V_N$ - dowolny nieterminal,
- e_1e_2 - sekwencja,
- e_1/e_2 - uporządkowany wybór,
- e^* - zero lub więcej powtórzeń,
- $!e$ - operator negacji.

Dodatkowo wyprowadza się dla wygody zapisu, jako pochodne do wymienionych powyżej, następujące symbole:

- $e?$ - wyrażenie opcjonalne (e/ϵ),
- e^+ - jedno lub więcej powtórzeń (e/e^*),
- $\&e$ - predykat and ($!(!e)$).

Dla zachowania zgodności z zapisem wyrażeń regularnych wprowadza się także:

- dopasowanie dowolnego znaku .,
- grupowanie wyrażeń (e),
- dopasowanie do klasy znaków $[s]$,
- dopasowanie łańcucha znaków $'s'$ - literał znakowy.

Rozpoznawanie łańcucha wejściowego z pomocą gramatyki wyrażeń parsujących rozpoczyna się od wyróżnionej pojedynczej reguły zwanej symbolem startowym. Pozycja w wej-

ściowym strumieniu znaków, od której rozpoczyna się parsowanie, zostaje ustawiona na początku łańcucha wejściowego. Jeśli wyrażenie parsujące dopasowało niepusty fragment ciągu wejściowego do opisywanego przez siebie wzorca, wówczas pozycja w strumieniu znaków (kursor), od której rozpatruje się kolejne reguły, przesunie się do przodu o długość rozpoznanego wcześniej fragmentu tekstu. Proces dopasowania i konsumpcji fragmentów ciągu wejściowego dla różnych operatorów wyrażeń parsujących opisano w tabeli 1. W kolumnie pierwszej przedstawiono rodzaj operatora wraz z jego operandami. W kolumnie drugiej wymieniono rozpatrywany tekst z dodatkowymi warunkami zapisanymi w nawiasach okrągłych. Notacja $\sim$ i $\nsim$ użyta do zapisu tychże warunków oznacza odpowiednio dopasowanie lub jego brak dla poszczególnych składowych wyrażenia. Spełnienie tych wymagań decyduje o konsumpcji wymienionego w kolumnie trzeciej tekstu ze strumienia znaków, a także określa, czy nastąpiło dopasowanie do wzorca tekstowego budowanego przez rozpatrywany operator, wymienione w kolumnie czwartej.

Jako przykład ilustrujący użycie gramatyki wyrażeń parsujących można podać zapis ich składni z wykorzystaniem ich samych:

```
pegs    <- spacing pattern+ EOF
pattern <- rule '<-' spacing expression
expression <- alternative ('/' spacing alternative)*
alternative <- (prefix? suffix)*
prefix <- ('&' / '!') spacing
suffix <- primary (('*' / '+' / '?') spacing)?
primary <- '(' spacing expression ')' spacing / literal /
           '.' spacing / charclass / rule !'<-'
literal <- ['] (!['] .)* ['] spacing
charclass <- '[' (!']' (. '-' . / .))* ']' spacing
rule <- [a-z,A-z]([a-z,A-z,0-9])* spacing
spacing <- (' ' / '\t' / EOL)*
EOL <- '\r' / '\n' / '\r\n'
EOF <- !.
```

2.1.1. Opis gramatyk wyrażeń parsujących

Wyrażenia parsujące w ujęciu generatora PegLi opisuje się za pomocą dwóch sekcji. Pierwsza (opcjonalna) sekcja dołącza gramatykę zdefiniowaną wcześniej, której można użyć do opisu składni nowego języka (częściowo lub w całości). Kolejna sekcja jest opisem reguł. Składnia używana w niej pokrywa się z wcześniej opisaną składnią wyrażeń parsujących. Jedynymi rozszerzeniami są: możliwość użycia reguł zdefiniowanych w gramatykach zewnętrznych, umożliwienie definiowania akcji semantycznych i instrukcje związane z budową abstrakcyjnego drzewa składni.

2.2. Biblioteka szablonów TempLi

TempLi (skrót od **Temp**late **Li**brary) jest autorskim językiem specyfikacji dziedzinowej oraz biblioteką oprogramowania stworzoną do generowania tekstu z wykorzystaniem zewnętrznych struktur danych.

Język specyfikacji TempLi, podobnie jak biblioteka `StringTemplate` [5], zakłada oddzielenie modelu generowanej aplikacji od jego formy w postaci kodu źródłowego (widoku).

Szablony kodu biblioteki TempLi składają się z dwóch najważniejszych części:

```
// I sekcja - szablonów i gramatyk
  using template nazwa;
// II sekcja - definicji atrybutów i wzorców generacyjnych
  //deklaracja atrybutu
  $x
  //definicja wzorca generacyjnego
  $zmienna($typ, $nazwa) -> $typ $nazwa';' $end
```

Pierwsza (opcjonalna) część przyłącza wcześniej zdefiniowane szablony kodu, co umożliwia korzystanie z zaimplementowanych wcześniej bibliotek kodu. Druga sekcja definiuje atrybuty i wzorce generacyjne, które pozwalają na implementację szablonów kodu.

2.2.1. Atrybuty szablonów kodu

Atrybuty (`$atrybut`) są danymi szablonów kodu. Są one powiązane z modelem wspierającym generowanie kodu źródłowego. Atrybuty przekazuje się do wzorców generacyjnych jako argumenty, których następnie używa się w ciele tychże wzorców. Możliwe jest używanie w ciele wzorca generacyjnego atrybutów zadeklarowanych w sekcji deklaracji atrybutów, na podobieństwo zmiennych zdefiniowanych w module języka programowania (np. Pythona). Atrybuty mogą być zarówno pojedynczymi zmiennymi, jak i listami zmiennych. Możliwe jest zagnieżdżanie atrybutów (podobnie jak ma to miejsce w przypadku pól składowych struktur i klas). Do składowych zagnieżdżonych w innym atrybucie należy odwołać się za pomocą operatora dostępu do składowej, np. `atrybut.pole`.

Wartości atrybutów są nadawane przed generowaniem kodu. Nie jest możliwe wyznaczenie wartości atrybutu w trakcie działania generatora. Ta właściwość cechująca języki funkcyjne upraszcza tworzenie generatorów kodu dzięki łatwiejszemu śledzeniu wartości atrybutów.

2.2.2. Wzorzec generacyjny

Wzorce generacyjne są elementem języka odpowiedzialnym za tworzenie fragmentów kodu źródłowego. Każdy wzorzec generacyjny jest sekwencją literałów tekstowych (zapisanych w apostrofach), wywołań innych wzorców generacyjnych i atrybutów, które dzięki operacji konkatenacji tworzą generowany tekst. Wzorce generacyjne mogą, podobnie jak funkcje w językach programowania, przyjmować argumenty. Argumentami mogą być zarówno atrybuty, jaki i inne wzorce generacyjne. Dla przykładu fragment:

```
$deklaracjaFunkcji($nazwaFunkcji)->
  'void ' $nazwaFunkcji '(' ');'
$end
```

definiuje wzorzec generacyjny przeznaczony do generowania deklaracji funkcji w języku C (w przykładzie nie uwzględniono argumentów funkcji i założono, że nie będzie ona zwracać wartości).

2.2.3. Reprezentacja szablonów w postaci kodu Pythona

Szablony kodu są transformowane do kodu źródłowego (w pracy wykorzystano język programowania Python), dzięki czemu możliwe jest wykorzystanie utworzonego w ten sposób kodu również w aplikacjach nie korzystających bezpośrednio z szablonów.

Każdemu zdefiniowanemu szablonowi kodu biblioteki TempLi odpowiada bezpośrednio wygenerowana klasa. Atrybuty są przechowywane jako pola składowe klasy, natomiast wzorce generacyjne są reprezentowane w postaci metod zwracających wygenerowany łańcuch znaków na podstawie przekazanych do funkcji argumentów oraz składowych klasy.

2.3. Środowisko programowania

Ręczne tworzenie i pielęgnacja języków specyfikacji za pomocą gramatyk wyrażeń parsujących, pomimo jasnego przełożenia gramatyki na odpowiadający jej parser, jest procesem żmudnym i podatnym na błędy. Dlatego też, żeby usprawnić ten proces stworzono zintegrowane środowisko programowania Gladi (rys. 1). Środowisko to wprowadza szereg usprawnień do edycji skryptów parserów, generowania kodu źródłowego oraz debugowania stworzonych za jego pomocą interpreterów. W ten sposób umożliwia ono tworzenia języków specyfikacji dla osób mniej obeznanych z teorią języków programowania.

Edytor kodu umożliwia podświetlanie składni wyrażeń parsujących, rozróżnia definicje i wywołania reguł, definicje akcji semantycznych i literałów znakowych. Dzięki temu kod gramatyki jest łatwiejszy do czytania i zrozumienia. Edytor podpowiada także, na podstawie pierwszych wprowadzonych liter, nazwy reguł, jakie można zastosować w danym momencie, co może znacznie przyśpieszyć proces tworzenia gramatyki. Ponadto, dodatkowy panel nawigacyjny pozwala, po kliknięciu na nazwie reguły, na szybkie znalezienie i przejście do fragmentu kodu odpowiedzialnego za definicję tej reguły. Dla dużych i złożonych gramatyk przydatną opcją jest przejście do definicji reguły po kliknięciu prawym klawiszem myszy w oknie edycyjnym i wyborze pozycji w menu odpowiedzialnej za przejście do stosownej definicji.

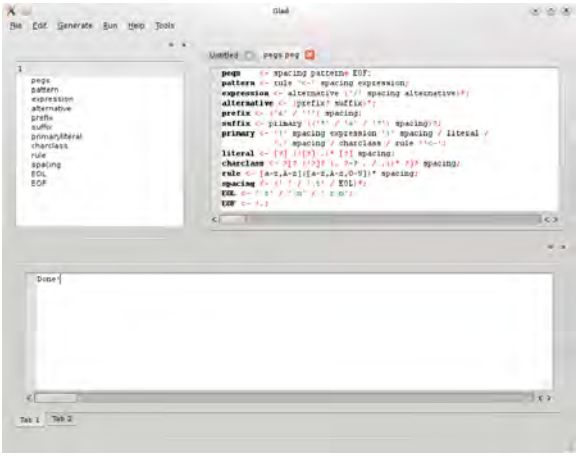
Podobne udogodnienia są udostępniane dla wzorców generacyjnych, również możliwe jest przejście do definicji wzorca generacyjnego po wyborze opcji z menu lub z panelu nawigacyjnego. Ponadto, edytor rozróżnia pomiędzy instrukcjami języka TempLi a instrukcjami języka, dla którego będzie generowany kod źródłowy, podświetlając tło instrukcji języka TempLi i języka generowanego za pomocą innych kolorów tła. Możliwe jest także podejrzenie wygenerowanego kodu źródłowego generatora w osobnym oknie w trybie tylko do odczytu.

Cały proces generowania interpreterów języków specyfikacji dziedzinowej jest w pełni zautomatyzowany. Zadaniem programisty jest jedynie wybranie stosownej opcji z menu, resztą, tzn. szczegółami wywołania generatorów kodu, zajmuje się środowisko programowania. Jeśli w specyfikacji gramatyki lub szablonów zostaną wykryte błędy składniowe lub semantyczne, środowisko pozwala na szybki wybór linii zawierającej taki błąd poprzez kliknięcie konsoli zawierającej komunikat o błędzie.

Środowisko programowania umożliwia uruchomienie wewnątrz siebie tworzonego interpretera, przy czym istnieje możliwość wyboru reguły startowej. Pozwala to na uproszczenie testowania interpretera oraz na szybkie zlokalizowanie i usunięcie błędów pojawiających się w procesie tworzenia języka specyfikacji. Sterowanie tym procesem jest realizowane z wbudowanej konsoli Pythona. Taki sposób debugowania interpretera jest możliwy na podstawie zarówno danych wczytanych z plików, jak i danych ręcznie wprowadzonych do środowiska jako dane testujące.

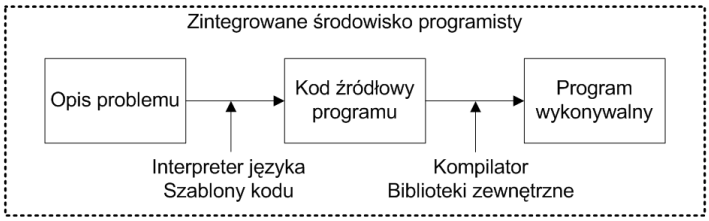
3. Charakterystyka budowanych języków specyfikacji

Języki specyfikacji dziedzinowej tworzone za pomocą opisywanego w pracy autorskiego środowiska są językami zewnętrznymi. Typowy język specyfikacji dziedzinowej, zbudowany



Rysunek 1. Zrzut ekranu przedstawiający panel główny zintegrowanego środowiska programowania Gladi

za pomocą tego szkieletu aplikacyjnego, składa się z dopasowanego do danego języka środowiska programowania nadzorującego pracę interpretera tego języka (rys. 2). Zadaniem interpretera jest kierowanie procesem generowania kodu aplikacji ze specyfikacji problemu przy użyciu stworzonych szablonów kodu. Zastosowanie szablonów ułatwia używanie dobrze przetestowanych i wydajnych bibliotek. Środowisko programowania automatyzuje tworzenie, z wygenerowanego przez interpreter kodu źródłowego za pośrednictwem kompilatora odpowiedniego dla tego kodu, gotowego do użycia programu wykonywalnego.



Rysunek 2. Składniki języka specyfikacji wraz z etapami generowania kodu

3.1. Zakres możliwych zastosowań

Rozpatrywane środowisko zostało utworzone na potrzeby implementacji języków specyfikacji dla metod numerycznych. Ułatwia więc obliczanie wielkości symbolicznych koniecznych do właściwego opisu rozpatrywanych zagadnień. Ponadto, ogranicza wpływ wprowadzonego poziomu abstrakcji na wydajność utworzonego kodu. Jest to związane z użyciem szablonów, które po zastosowaniu nie wprowadzają narzutów do czasu wykonania docelowego programu. Szablony umożliwiają również zastosowanie dobrze przetestowanych i wydajnych bibliotek, co wpływa pozytywnie na jakość utworzonego oprogramowania. Oferowane udogodnienia uwzględniają również budowę prostych środowisk programowania z wsparciem do podświetlania słów kluczowych, operacji edycyjnych i automatycznego uruchamiania interpretera. Opisywane właściwości nie uniemożliwiają więc zastosowań środowiska do budowy języków również dla innych dziedzin. Wymagane jest jednak przyjęcie podobnej metodyki budowy języka.

Należy zwrócić również uwagę na ograniczenia - przede wszystkim środowisko w bezpośredni sposób wspiera jedynie tworzenie języków tekstowych. Nie jest ono uciążliwe dla metod numerycznych, gdzie większość elementów jest przedstawiana w postaci równań matematycznych, uniemożliwia jednak zastosowania w dziedzinach zależnych od reprezentacji graficznej zagadnienia, np. przy budowie interfejsu użytkownika. Skomplikowane jest również utworzenie translatora, gdy istnieje duża różnica pomiędzy reprezentacją w postaci skryptu a kodem docelowym (np. gdy językiem docelowym byłby język maszynowy komputera). Wynika to z faktu, że środowisko nie umożliwia, w chwili obecnej, transformacji reprezentacji wewnętrznej kodu. Ponadto wygenerowane, dedykowane dla zagadnienia środowiska programowania ograniczają się tylko do wyżej wymienionych operacji (operacje edycyjne, uruchomienie interpretera) - nie ma możliwości z poziomu środowiska dołączenia dodatkowych narzędzi, takich jak debuggery tworzonych skryptów.

3.2. Etapy tworzenia języków specyfikacji dziedzinowej

Tworzenie języków specyfikacji dziedzinowej w zaprezentowanym w pracy frameworku składa się z trzech przeplatających się ze sobą części. Pierwszą z nich jest wyrażenie dziedziny problemowej w postaci gramatyki języka specyfikacji dziedzinowej. W zaprezentowanym podejściu używa się gramatyk wyrażeń parsujących. Dostarczają one sformalizowaną formę zapisu języka specyfikacji, pozwalają na uniknięcie wielu problemów powstających podczas implementacji i rozszerzania gramatyk implementowanych za pomocą tradycyjnych narzędzi [6].

Drugą częścią, ściśle związaną z tworzeniem gramatyki, jest implementacja szablonów kodu. Jest ona o tyle istotna, że języki specyfikacji dziedzinowej dla metod numerycznych powstają zazwyczaj przy współpracy z zewnętrzną biblioteką oprogramowania. Etap generowania kodu jest również ważny dla dostarczenia środowiska oprogramowania dostosowanego do języka specyfikacji dziedzinowej. Ponadto, możliwe jest również wygenerowanie pluginów do współpracy z zewnętrznymi aplikacjami inżynierskimi.

Ostatnią częścią tworzenia języka specyfikacji dziedzinowej jest testowanie powstałego interpretera. Testowanie polega na debugowaniu powstałego języka w środowisku programowania. Możliwe jest także przeprowadzenie testów jednostkowych, polegających na weryfikacji rzeczywistych rezultatów działania reguł gramatycznych z spodziewanym wynikiem tychże operacji.

4. Język specyfikacji - Finele

Wcześniej opisane środowisko znalazło zastosowanie przy implementacji języka specyfikacji Finele. Został on zaprojektowany dla metody elementów skończonych. Wybór wynika z szerokiego zastosowania wcześniej wymienionej metody w pracach realizowanych w instytucie, w którym pracują autorzy, co pozwala na weryfikację uzyskanych rozwiązań.

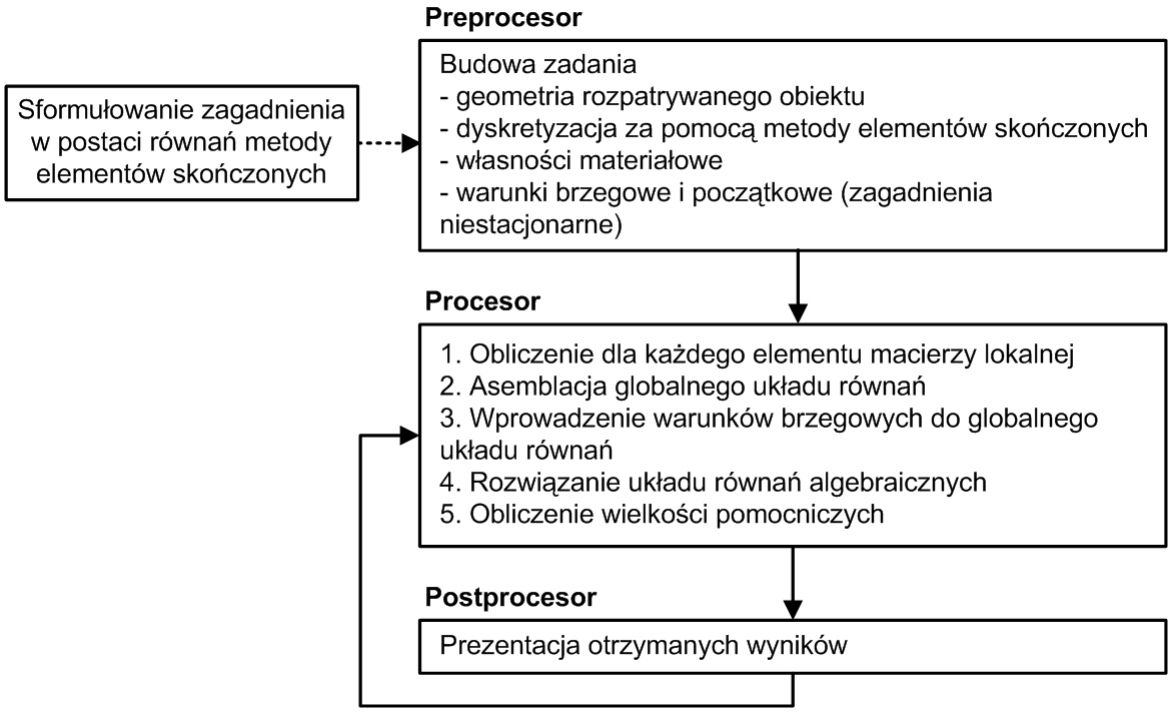
Rozpatrywany język Finele nie jest pierwszym językiem specyfikacji dedykowanym metodzie elementów skończonych. Wcześniejsze jednak języki były przede wszystkim wewnętrznymi językami specyfikacji (np. PIER [7] - Common Lisp, Analysa [8] - Scheme, MyFem [9] - Python, Feel++ [10] - C++). Popularność tego typu rozwiązań wynika ze względnie łatwego przeniesienia biblioteki numerycznej na poziom języka specyfikacji - granica pomiędzy biblioteką a językiem osadzonym jest umowna. Ogranicza to jednak składnię projektowanego języka do składni języka hosta. Dodatkowym utrudnieniem mogą być w niektórych przypadkach komunikaty kompilatora języka hosta używanego w translacji, widać to szczególnie w

przypadku wykorzystania technik metaprogramowania w języku C++. Utrudniona jest również, ze względu na ścisły związek z językiem, wymiana bibliotek wspomagających tworzenie oprogramowania.

Do budowy języków specyfikacji dla metody elementów skończonych używano również tradycyjnych narzędzi do budowy języków, takich jak bison (np. FreeFEM [11], GetDP [12]). Przy czym rozwiązanie to wymaga dobrej znajomości działania zastosowanych w narzędziu algorytmów parsowania, ze względu na możliwe konflikty pomiędzy regułami wynikające z niejednoznaczności gramatyki.

4.1. Metoda elementów skończonych

Rozwiązywanie dowolnego zagadnienia za pomocą metody elementów skończonych następuje w kolejnych krokach, przypisanych odpowiednio do preprocesora, procesora oraz postprocesora (rys. 3). Celem działania preprocesora jest przygotowanie danych do obliczeń, może to być na przykład wyznaczenie wielkości związanych z warunkami brzegowymi (np. zakłada się w sposób graficzny brzeg z określoną temperaturą) oraz, jeśli zagadnienie nie jest stacjonarne, określenie warunków początkowych. Ponadto, w tym etapie wprowadza się wartości własności materiałowych. Zdefiniowane w ten sposób dane zadania są zazwyczaj zapisywane do plików, które mogą być już użyte do obliczeń.



Rysunek 3. Etapy uzyskiwania rozwiązania za pomocą metody elementów skończonych (zagadnienia liniowe)

Celem działania procesora jest wyliczenie odpowiednich wielkości fizycznych, przyporządkowanych węzłom siatki elementów skończonych. Analizowany obszar (objętość) dzielony jest (wirtualnie) na elementy skończone. W ten sposób powstaje siatka elementów skończonych. Każdy element skończony scharakteryzowany jest poprzez jego funkcje kształtu. Liczba funkcji kształtu jest równa liczbie węzłów utworzonych dla elementu skończonego. Jeśli węzły utworzone są tylko w wierzchołkach elementu, to jest on elementem liniowym.

Wartości wielkości poszukiwanej wewnątrz (i na powierzchniach) elementu skończonego wyznacza się z zależności:

$$u(x_i) = N_i u_i \tag{1}$$

gdzie:
— N_i – są funkcjami kształtu,
— u_i – są wartościami wielkości poszukiwanej w węzłach,
— x_i – są współrzędnymi przestrzennymi,
natomiast i zmienia się od 1 do liczby węzłów w elemencie skończonym.

Dla zagadnień liniowych obliczenia przeprowadza się najpierw dla każdego elementu. Wyznacza się, zgodnie z uzyskanym za pomocą dyskretyzacji równaniem algebraicznym, macierze lokalne oraz lokalne wektory prawych stron. Następnie współczynniki te są asemblowane do globalnego układu równań:

$$\mathbf{A}\mathbf{u} = \mathbf{b} \tag{2}$$

gdzie:
— **A** – jest globalną macierzą współczynników,
— **u** – jest wektorem wielkości poszukiwanej,
— **b** – jest wektorem prawych stron.

Powstały po tych wszystkich etapach globalny układ równań rozwiązuje się za pomocą wybranej metody numerycznej. Uzyskane dla węzłów rozwiązanie może być jeszcze wykorzystane do wyznaczenie dodatkowych wielkości, dla przykładu w transporcie ciepła mogą to być strumienie ciepła. Jeśli zagadnienie nie jest zagadnieniem stacjonarnym, wówczas przeprowadzone obliczenia powtarza się dla następnych kroków czasowych. Wyznaczone za pomocą procesora wielkości mogą być następnie zwizualizowane za pomocą postprocesora.

4.2. Charakterystyka języka specyfikacji Finele

Język specyfikacji Finele (bazujący na wcześniejszych pracach - [13, 14]) został zaprojektowany do wspierania tworzenia opisu zadań rozpatrywanych za pomocą metody elementów skończonych dla zagadnień liniowych. Bazuje on na spostrzeżeniu, że niektóre etapy tworzenia rozwiązania są wspólne dla różnych zastosowań (odczyt danych wejściowych, zapis wyników, asemblacja macierzy globalnej) i mogą być automatycznie wygenerowane, bez potrzeby jawnej ich implementacji. Język jest zaprojektowany do budowy procesorów metody elementów skończonych, natomiast zadania preprocesora i postprocesora są realizowane za pomocą zewnętrznego oprogramowania (np. programu GiD).

Skrypt w języku Finele składa się z sekcji opisujących poszczególne składowe zadania. Sekcja rozpoczynająca się od słowa kluczowego `MaterialProperty` wyszczególnia wszystkie własności materiałowe stosowane do budowy zadania, które są wprowadzane za pomocą preprocesora.

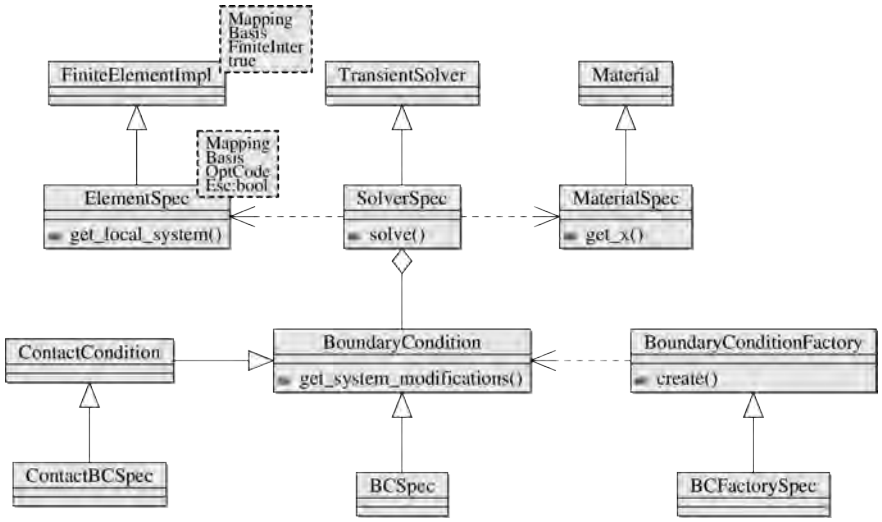
Następna część jest odpowiedzialna za opis solwera (solwer - program lub część programu realizująca obliczenia). Wymieniane są tutaj niewiadome uzyskiwane po rozwiązaniu układu równań (podsekcja niewiadomych), deklaracje pomocniczych macierzy, wektorów i wielkości skalarnych (sekcja deklaracji), a także sposób budowania macierzy lokalnej dla elementu (podsekcja budowniczego). Przyjęto tutaj konwencję macierzową, słowo kluczowe `Nabla` opisuje operator Hamiltona (∇), słowo kluczowe `Trans` oznacza transpozycję macierzy zapisanej po lewej stronie tego operatora. Całki po obszarze elementu $\int_\Omega w^T N d\Omega$ są zapisywane

w postaci: $(w\ Trans, N) * dx$. Słowa kluczowe dt, w oraz N opisują odpowiednio krok czasowy, funkcje wagowe oraz funkcje kształtu. W nawiasach trójkątnych zapisuje się stałe, które można wyliczyć jeszcze przed rozwiązaniem globalnego układu równań. Wartości wyliczone w poprzednich krokach czasowych są zapisywane w postacie nazwa[-liczba], gdzie nazwa to identyfikator wyliczonej w węźle wartości, natomiast poprzedzona znakiem minusa liczba w nawiasie kwadratowym oznacza, z którego poprzedniego kroku czasowego ma być wzięta ta wartość.

Ostatnia sekcja `BoundaryConditions` opisuje warunki brzegowe. Bez tej sekcji zakłada się tylko wprowadzenie warunku brzegowego pierwszego rodzaju. Warunki brzegowe posiadają własne współczynniki (sekcja `coeff`). Sposób ich wprowadzania do globalnego układu równań ujmuje sekcja `builder`. Ponadto, można zdefiniować warunek brzegowy na styku dwóch obszarów, informuje o tym słowo kluczowe `contact`, zapisane w sekcji początkowej tegoż warunku brzegowego. Warunki brzegowe definiowane w opisywanej sekcji nie są jawnie wprowadzane do definiowanego zadania. Są ustawiane za pomocą preprocesora, natomiast wielkości je opisujące są zapisywane do plików, które mogą być już wykorzystane jako dane dla solwera.

4.3. Generowanie aplikacji

Stworzony język specyfikacji dziedzinowej jest zgodny z wcześniej opisanym schematem - z opisu problemu tworzony jest kod źródłowy, następnie jest on kompilowany i linkowany z zewnętrznymi bibliotekami. W przypadku języka `Finele` zewnętrzną biblioteką ułatwiającą tworzenie oprogramowania jest biblioteka `eNsc` – gałąź biblioteki `nsc` rozwijanej w Instytucie Informatyki Teoretycznej i Stosowanej Politechniki Częstochowskiej.



Rysunek 4. Implementacja zagadnienia z wykorzystaniem biblioteki eNsc

Najważniejsze dla generowania programu klasy oraz związki pomiędzy nimi przedstawia rysunek 4. Klasy bez przyrostka `Spec` są klasami bibliotecznymi, natomiast klasy z tym przyrostkiem są klasami tworzonymi dla zagadnienia (ich nazwy będą do niego dostosowane). Generowany kod źródłowy języka C++ jest podzielony na pliki (nazwy plików, poza plikiem main.cpp, są dostosowywane do rozpatrywanego zagadnienia):

— main.cpp - zawiera funkcję main(),
— solver.h - zawiera definicję klasy solwera,

— solver.cpp - zawiera implementacje metod solwera,
— elements.h - zawiera definicje klas opisujących elementy skończone oraz definicję ich metod,
— bcs.h - zawiera definicje klas implementujących warunki brzegowe,
— bcs.cpp - zawiera definicje metod klas warunków brzegowych,
— materials.h - zawiera definicje klas obsługujących własności materiałowe,
— materials.cpp - zawiera definicje metod klas własności materiałowych,
— Makefile - zawiera skrypt budujący gotowy do działania program.

Generacja pojedynczego pliku jest realizowana za pomocą zaimplementowanego do tego celu wzorca generacyjnego. Wzorce generacyjne przedstawione tutaj działają dla zagadnień liniowych rozwiązywanych za pomocą metody elementów skończonych z wykorzystaniem biblioteki `eNsc`. Oprogramowanie to zostało napisane w języku C++, stąd wzorce generacyjne opisują sposób generowania kodu źródłowego języka C++.

Plik main.cpp zawiera implementację funkcji main(). Jej zadaniem jest tworzenie obiektu klasy solwera, a następnie za pośrednictwem tegoż obiektu wczytywanie danych z plików siatki elementów skończonych, własności materiałowych, warunków brzegowych (dzięki metodzie `read_problem_data()`) oraz rozwiązanie zadania (za pomocą metody `solve()`). Ponadto, w funkcji main() może odbywać się obsługa wyjątków powstających podczas działania programu (np. przez wczytanie pliku siatki z niepoprawnym formatem danych). Zadaniem generatora jest w tym przypadku dostarczenie dwóch parametrów do wzorca generacyjnego, które zastąpią atrybuty `$SolverHeader` (nazwa pliku nagłówkowego solwera) i `$Solver` (nazwa klasy solwera). Wygenerowane nazwy będą związane z rozpatrywanym zagadnieniem np. dla zagadnienia opisywanego równaniem Fouriera solwer będzie się nazywał `HeatSolver`, natomiast jego plik nagłówkowy `heat_solver.h`.

Kluczową dla tworzonego zadania jest klasa `SolverSpec`. Zapewnia ona budowę globalnej macierzy dla zadania, umożliwia wprowadzanie warunków brzegowych oraz rozwiązanie powstałego w ten sposób układu równań. Celem jej działania jest również wczytywanie danych z plików wejściowych oraz zapis wyników do plików wyjściowych. Ponieważ dla zagadnień niestacjonarnych konieczne jest również wprowadzenie warunków początkowych, zapamiętywanie wyników z poprzednich kroków czasowych, wyznaczanie kryteriów zakończenia obliczeń, stąd stosowane jest dziedziczenie po klasie `TransientSolver`. Wzorce generacyjne biorą pod uwagę tę relację i część zachowań parametryzują w zależności od rodzaju zagadnienia. W tym celu stosuje się przede wszystkim instrukcję warunkową sprawdzającą, czy implementuje się w danym momencie zagadnienie niestacjonarne. Wyznacza się w ten sposób obecność relacji dziedziczenia, deklarację i definicję metody zwracającej referencję do klasy solwera, odpowiedniej wersji destruktora a także sposobu prowadzenia obliczeń, uwzględniającego wprowadzenie warunków początkowych, uwzględnienie wszystkich wymaganych kroków czasowych i zapis danych po osiągnięciu warunku zakończenia obliczeń (można zadecydować o zapisie wyników tylko z niektórych kroków czasowych). Ponadto, wzorce generacyjne uwzględniają zmienione nazwy klas własności materiałowych. Parametryzuje się również sposób wczytywania danych wejściowych, rozwiązywania globalnego układu równań i zapisywania wyników.

Klasą zapewniająca własności elementu skończonego jest klasa `ElementSpec`. Najważniejszym jej zadaniem jest tworzenie macierzy lokalnej i lokalnego wektora prawych stron dla elementu. W tym celu implementuje się metodę `get_local_system()`. W wzorcu generacyjnym tworzenie macierzy lokalnej jest realizowane przez wywołanie argumentu tegoż wzorca – `finiteElementRepr`. Dzięki dziedziczeniu po klasie `FiniteElementImpl`

klasa `ElementSpec` wyznacza również przekształcenia pomiędzy układami współrzędnych, a także pozwala na dostęp do funkcji interpolacyjnych i wagowych. Klasy elementów skończonych są klasami szablonowymi (w sensie elementu składniowego C++), stąd też cała ich definicja musi się znaleźć w pliku nagłówkowym. Stąd też wzorzec generacyjny istnieje tylko dla pliku nagłówkowego.

Warunki brzegowe są implementowane za pomocą klas dziedziczących własności i interfejs po klasie `BoundaryCondition`. Implementują one przede wszystkim wirtualną metodę `get_system_modifications()`. Zadaniem tej metody jest tworzenie tablicy zawierającej lokalne modyfikacje zadania, wprowadzane potem do globalnego układu równań. Hierarchia dziedziczenia uwzględnia również specyfikę warunków brzegowych na stykach obszarów dzięki wprowadzeniu dodatkowej klasy `ContactCondition`. Z warunkami brzegowymi jest stowarzyszona klasa `BCFactorySpec` implementująca wzorzec projektowy *Metoda wytwórcza* [15]. W klasie `BoundaryConditionFactory` określony jest interfejs do tworzenia obiektów warunków brzegowych, natomiast generowana klasa pochodna od tejże klasy decyduje o tym, jakie obiekty warunków brzegowych mają być tworzone. To tworzenie instancji klas warunków brzegowych jest realizowane podczas wczytywania danych.

Klasa własności materiałowych `MaterialSpec` implementuje metody `get_x`, gdzie `x` oznacza nazwę danej własności. Podobnie jak w przypadku warunków brzegowych, dla własności materiałowych jest wyznaczona klasa fabryczna tworząca obiekty warunków brzegowych podczas przetwarzania plików wejściowych.

Wzorce generacyjne dla aplikacji solwera uzupełnia wzorzec generacyjny dla pliku Makefile używanego do automatycznej kompilacji wygenerowanych źródeł programu. Zadaniem generatora jest stworzenie listy plików oraz wstawienie ścieżek do plików nagłówkowych i bibliotecznych, jeśli nie znajdują się one w standardowej ścieżce wyszukiwań.

4.4. Skrypt Finele dla równania Fouriera

Skrypt Finele analizowanego zagadnienia jest tworzony dla jego modelu numerycznego. Model numeryczny dla zagadnień opisywanych równaniem Fouriera został przygotowany na podstawie rozważań zawartych w wcześniejszej pracy [14]. Równaniem wyjściowym, dla niejawnego schematu całkowania po czasie, jest:

$$(\mathbf{M^{n+1}}+\Delta t\mathbf{K^{n+1}})\mathbf{T^{n+1}} = \mathbf{M^{n+1}}\mathbf{T^{n}}+\Delta t\mathbf{b^{n+1}} \tag{3}$$

Macierze $\mathbf{M}$ i $\mathbf{K}$ w tym równaniu oblicza się z zależności:

$$\mathbf{M} = \mathbf{c}\rho \int_{\mathbf{\Omega}} \mathbf{N^T N}\, \mathbf{d\Omega} \tag{4}$$

$$\mathbf{K} = \lambda \int_{\mathbf{\Omega}} \nabla\mathbf{^T N} \cdot \nabla\mathbf{N}\, \mathbf{d\Omega} \tag{5}$$

Powyższe równanie (uzupełnione o jeden warunek brzegowy) jest zapisane z wykorzystaniem języka Finele w sposób następujący:

```
MaterialProperty{
    lambda;
    c;
```

```
    rho;
}

Solver Heat{
    unknowns -> T;
    builder:
        << c * rho >*(w Trans,N)*dx
         + dt * <lambda> * (Nabla w Trans, Nabla N)*dx> * T
        = <<c * rho> * (w Trans,N)*dx> * T[-1];
}

BoundaryConditions{
    bc Neumann{
    coeff -> q;
    builder:
    <q>*(w Trans,N) * ds;
    }
}
```

Skrypt wyróżnia własności materiałowe wprowadzane na etapie budowania zadania - współczynnik przewodności cieplnej (`lambda`), ciepło właściwe (`c`) oraz gęstość (`rho`):

```
MaterialProperty{
lambda;
c;
rho;
}
```

Wprowadzono także niewiadomą uzyskiwaną po rozwiązaniu globalnego układu równań - temperaturę:

```
unknowns -> T;
```

Ponieważ nie było potrzeby wprowadzenia wielkości pomocniczych, więc podsekcja deklaracji w tym wypadku nie została włączona do skryptu. W następnej części zapisano równanie (3) z wykorzystaniem opisanych wcześniej konwencji składniowych:

```
<< c * rho >*(w Trans,N)*dx
   + dt * <lambda> * (Nabla w Trans, Nabla N)*dx> * T
    = <<c * rho> * (w Trans,N)*dx> * T[-1];
```

Ostatnia część, zaczynająca się od nagłówka `BoundaryConditions`, wprowadza jeden warunek brzegowy (warunek brzegowy Dirichleta – I rodzaju – jest dołączony automatycznie): warunek brzegowy II rodzaju – na brzegu Γ jest zadawany stały strumień ciepła. W zapisie języka Finele przyjmuje on postać:

```
bc Neumann{
coeff -> q;
builder:
<q>*(w Trans,N) * ds;
}
```

5. Podsumowanie

Praktyczne zastosowanie narzędzi i technik opisanych w artykule pozwala na szybkie przygotowanie języków specyfikacji dziedzinowej. Wprowadzone ułatwienia obejmują zintegrowane środowisko programowania automatyzujące proces budowy języka specyfikacji dziedzinowej. Ponadto, dzięki opartej na gramatykach wyrażeń parsujących bibliotece i generatorowi PegLi, zmniejszono trudności związane z przygotowaniem parserów dla języków specyfikacji. W ramach pracy stworzono również bibliotekę TempLi upraszczającą tworzenie szablonów kodu.

Wymienione wcześniej narzędzia znalazły również praktyczne zastosowanie przy implementacji deklaratywnego języka specyfikacji Finele. Jest on językiem specyfikacji zaprojektowanym do realizacji zadań za pomocą metody elementów skończonych. Praca pokazuje również prosty przykład implementacji problemu z wykorzystaniem języka Finele.

Bibliografia

[1] A. van Deursen, P. Klint, J. Visser. *Domain-Specific Languages: An Annotated Bibliography. ACM SIGPLAN Notices*, 35(6):26–36, June 2000.

[2] M. Fowler. *Domain Specific Languages*. Addison-Wesley Professional, 2010.

[3] B. Ford. *Parsing Expression Grammars: A Recognition-Based Syntactic Foundation*. *Symposium on Principles of Programming Languages*, pp. 111–122. ACM Press, 2004.

[4] *Maxima, a Computer Algebra System*. *http://maxima.sourceforge.net/*, 2003.

[5] T. Parr. *A Functional Language For Generating Structured Text*. *http://www.cs.usfca.edu/~parrt/papers/ST.pdf*, 2006.

[6] R. Ierusalimschy. *A Text Pattern-Matching Tool based on Parsing Expression Grammars*. *Software: Practice and Experience*, pp. 39(3):221–258, 2009.

[7] N. Sharma, P. Wang. *High-level User Input Specifications for Finite Element Code Generation*. 1992. preprint.

[8] B. Bagheri, R. Scott. *About Analysa*. *http://people.cs.uchicago.edu/~ridg/al/aa.ps*, 2003.

[9] J. Riehl. *Reflective techniques in extensible languages*. PhD thesis, The University of Chicago, Department of Computer, 2008.

[10] C. Prud'homme. *Life: overview of a unified C++ implementation of the finite and spectral element methods in 1D, 2D and 3D*. *Proceedings of the 8th international conference on Applied parallel computing: state of the art in scientific computing*, PARA'06, pp. 712–721. Springer-Verlag, 2007.

[11] F. Hecht, O. Pironneau, A. L. Hyaric, K. Ohtsuka. *Freefem++*. Laboratoire Jacques-Louis Lions, Universit´e Pierre et Marie Curie, Paris, 2008.

[12] P. Dular, C. Geuzaine. *GetDP Reference Manual*. University of Liege, 2012.

[13] A. Grosser. *Finele–Domain Specific Language for Linear Finite Element Method*. *4th Polish and International PD Forum–Conference on Computer Science*, pp. CD–ROM, 2008.

[14] N. Sczygiol, A. Grosser. *Finele–język specyfikacji dziedzinowej do automatycznego generowania kodu metody elementu skończonego*. *Materiały XV Konferencji Komplastech 2008*, pp. 201–206, 2008.

[15] E. Gamma, R. Helm, R. Johnson, J. Vlissides. *Wzorce projektowe: Elementy oprogramowania obiektowego wielokrotnego użytku.* WNT, 2005.

Interaktywne wyszukiwanie informacji w repozytoriach danych tekstowych

Marcin Deptuła, Julian Szymański, Henryk Krawczyk

Politechnika Gdańska, Wydział Elektroniki, Telekomunikacji i Informatyki

Abstract:

W artykule przedstawione zostały architektura oraz projekt systemu, którego celem jest umożliwienie zbudowania platformy pozwalającej na indeksowanie dużych kolekcji tekstowych oraz wyszukiwania w nich, za pomocą autorskich algorytmów, opartych o zysk informacjny oraz interaktywną komunikację z użytkownikiem. Przeprowadzono ocenę skuteczności zastosowanych algorytmów pod względem zarówno klasteryzacji jak i zbieżności algorytmu opartego o zysk informacyjny.

Keywords:

Wyszukiwanie informacji, interaktywne wyszukiwanie, klasteryzacja, zysk informacyjny

1. Wprowadzenie – wyszukiwanie informacji w tekście

Najpopularniejsze silniki wyszukiwania informacji w zbiorach tekstowych wykorzystują prosty model komunikacji. Typowy scenariusz wyszukiwania jest tu następujący:

1. Wysłanie zapytania tekstowego przez użytkownika.
2. Przetworzenie zapytania oraz pobranie wyników w postaci listy rezultatów, np. artykułów lub stron internetowych.
3. Posortowanie wyników względem określonej miary istotności.
4. Przedstawienie wyników użytkownikowi w postaci uporządkowanej listy.

Schemat ten jest powszechnie używany przez różne silniki wyszukiwania i zawiera dwa główne elementy: indeksowanie i sortowanie. Pierwszy element wykorzystuje wynajdywanie podobieństw między frazą zapytania a indeksowanymi zasobami. Do tego celu wykorzystywane jest wyszukiwanie pełnotekstowe oparte na odwróconych indeksach [1] lub bardziej wyrafinowane metody jak np. indeksowanie semantyczne (LSI - Latent Semantic Indexing) [2]. Ten element silnika wyszukującego przetwarza treści indeksowanych zasobów do utworzenia zbioru rezultatów. Drugi element wykorzystuje algorytm rankingowania w celu uporządkowania rezultatów według określonej miary istotności. Algorytm rankingowy typowo wykorzystuje własności struktury powiązań między indeksowanymi zasobami, jak ma to miejsce np. w stosowanym przez Google algorytmie PageRank [3]. Istnieje szereg innych algorytmów rankingujących [4], większość z nich wykorzystuje analizę struktury powiązań jak np.: HITS (Hyperlink-Induced Topic Search) [5].

Najnowszym trendem rozwijanym w publicznych silnikach wyszukiwania jest możliwość wzajemnego polecania pewnych treści przez samych użytkowników (*collaborative and social information retrieval*) [6]. Dzięki temu treści, które przez wielu użytkowników zostały uznane

za interesujące, są traktowane jako lepsze. Przykładem implementacji takiego systemu jest wprowadzony przez wyszukiwarkę Google przycisk „+1".

Bez względu na stopień złożoności przedstawionych algorytmów, schemat komunikacji z użytkownikiem pozostaje taki sam: zapytanie - indeksowanie - rankingowanie. Jedyne dane do wykonania zapytania wyszukiwawczego dostarczone przez użytkownika to zbiór słów kluczowych. Podejście takie nie pozwala na:

— Określenie kontekstu słów, w przypadku słów wieloznacznych, które w różnych kontekstach oznaczają co innego.
— Doprecyzowanie zapytania w przypadku, gdy zapytanie użytkownika jest zbyt ogólne, by udzielić dokładnej odpowiedzi. Może to wynikać z faktu, że użytkownik może nie być w stanie nazwać dokładnie rzeczy, którą szuka i dopiero po zobaczeniu rezultatów może określić, czy są one dla niego istotne.
— Uogólnienie zapytania, w przypadku, gdy jest ono zbyt szczegółowe i użytkownik nie znalazł w wynikach tego, czego szukał, mimo iż zapytanie zostało zadane poprawnie.

Opisany w tym artykule system wyróżnia się od typowych silników wyszukiwania odmiennym podejściem do sposobu, w jaki komunikuje się on z użytkownikiem oraz pobiera od niego informacje. Standardowy scenariusz wyszukiwania został zastąpiony przez następujący schemat:

1. Wysłanie zapytania tekstowego przez użytkownika.
2. Przetworzenie zapytania oraz określenie listy elementów pasujących do zapytania.
3. Pogrupowanie listy elementów w zbiory logicznie spójnych elementów.
4. Określenie grup tematycznych – konceptualnych kierunków, które najlepiej dzielą wyszukane elementy.
5. Pobranie od użytkownika odpowiedzi dotyczącej wybranego kierunku lub też prośby o uogólnienie wyników.
6. Filtracja i/lub dodanie nowych wyników do listy aktualnie przedstawionej użytkownikowi.
7. Powrót do kroku 3.

Zaproponowany scenariusz odróżnia się od standardowej metody wyszukiwania przede wszystkim swoją interaktywnością, a także sposobem przedstawiania wyników w postaci pokategoryzowanej. Interaktywność systemu wynika z założenia, że wysłanie zapytania tekstowego przez użytkownika nie powinno kończyć jego zapytania, a jedynie przedstawiać pierwszą listę wyników. Lista ta będzie podatna na wszystkie wymienione wcześniej potencjalne błędy, jak np. niejednoznaczność haseł, jednakże, udostępnione zostaną użytkownikowi narzędzia pozwalające na wykluczenie tematów go nieinteresujących i doprecyzowanie zbioru rezultatów do tematyki zgodnej z jego oczekiwaniami.

Ze względu na to, że system musi być w stanie zakwalifikować otrzymane wyniki do wielu kategorii tematycznych, co pozwala mu na określenie m. in. kontekstu wyszukanych rezultatów, może on również zwiększyć czytelność przedstawionych wyników. W tym celu, zrezygnowano ze standardowej posortowanej listy wyników, zastępując ją grupami. Każda grupa (klaster) składa się z należących do niego elementów, które stanowią tematyczny podzbiór elementów zbioru wynikowego, a także nazwy, która w najdokładniejszy możliwy sposób charakteryzuje zebrane w grupie elementy. Najdokładniejszy sposób oznacza tu, że nazwanie danego klastra używając innej, bardziej szczegółowej (na podstawie określonego a priori drzewa kategorii) nazwy nie odzwierciedliłoby w pełni wszystkich elementów znajdujących się w tym klastrze.

2. Architektura wyszukiwarki

System został zaplanowany w sposób architekturze modułowej. Podstawowym podziałem charakteryzującym system jest oddzielnie części obliczeniowej od warstwy prezentacji i dalszy ich podział na moduły funkcjonalne. Podział ten został wprowadzony na poziomie procesów, co pozwala na uzyskanie kilku pożądanych efektów:

— Jeden proces obliczeniowy może obsłużyć równolegle wielu klientów (np. aplikacje klienckie lub serwer www) jednocześnie.
— Proces obliczeniowy może znajdować się na przystosowanym klastrze, do którego dostęp jest ograniczony i kontrolowany.
— Klient w postaci serwera www nie zużywa „sztucznie" ogromnych zasobów, gdyż nie hostuje w swoim procesie danych, co uniemożliwiłoby wprowadzenie np. load balancingu.

Również istotny jest podział na moduły wewnątrz głównej części obliczeniowej. Podział ten ma na celu ułatwienie rozszerzalności poszczególnych algorytmów, dzięki ich łatwej wymianie. Pozwala to na łatwe porównywanie jakości działania wielu modułów, zarówno pod względem wydajnościowym, jak i jakości, czy poprawności wyników.

Przykładem zastosowania tej architektury jest moduł klasteryzacji. Dwa identyczne pod względem zwracanych wyników podejścia zostały zaimplementowane w systemie na dwa różne sposoby. Pierwszy z nich działa bezpośrednio na reprezentacji wewnętrznej danych, opisanych w sekcji 2.2, dzięki czemu jest gotowy do użycia zawsze. Drugi, bazuje na specjalnie przygotowanych strukturach danych, których obliczenie jest długotrwałe, jednak dzięki temu sam algorytm został znacząco zoptymalizowany. Dzięki zastosowanej architekturze, system potrafi użyć szybszego algorytmu jeżeli jest to możliwe dla konkretnych danych, traktując wolniejszy algorytm jako „wyjście awaryjne".

2.1. Dane wejściowe

W trakcie przeprowadzanych badań zostały wykorzystane dane pochodzące z baz danych Wikipedii. Wikipedia została wybrana jako źródło danych do badań, ze względu na rozróżnienie baz danych na wersje językowe o różniących się rozmiarach, co pozwoliło na testowanie systemu na różnej wielkości bazach danych bez ich ręcznego dzielenia. Wybór tej bazy danych miał również znaczenie w późniejszej fazie testowania, gdyż możliwe jest przyrównanie otrzymanych wyników (np. klasteryzacji) do struktury artykułów obecnej w bazie danych Wikipedii.

Zaimportowane zostały dane z dwóch baz, pierwszą z nich jest SimpleWiki, która posiada zawiera artykuły napisane w uproszczonym języku angielskim, także ich ilość jest w znaczący sposób ograniczona. Drugą bazą danych była baza polskiej Wikipedii.

W ramach SimpleWiki zaimportowanych zostało 55675 artykułów, podzielonych na 11196 kategorii. W przypadku polskiej Wikipedii, liczby te wynosiły odpowiednio 678893 oraz 63977.

Importowane dane zawierały informacje o:
— nazwach artykułów oraz kategorii,
— treści artykułów,
— przynależności artykułów do kategorii,
— powiązaniach między kategoriami.

Importowane dane wejściowe zawierają informacje wyłącznie o bezpośrednich połączeniach między artykułami a kategoriami, algorytmy wyszukiwania opisane w sekcji 3 bazują

na szerszej i pełniejszej bazie informacji. Z tego względu, w trakcie importu, dane przetwarzane są przez algorytm ekstrakcji przynależności kategorialnej [7], którego dokładny opis tu pominięto. Jego działanie polega w ogólności na wywnioskowaniu połączeń o mniejszych wagach (znaczeniu) między poszczególnymi parami *artykuł - kategoria*, lub *artykuł - artykuł* na podstawie dostępnych danych.

2.2. Reprezentacja danych

Dane używane przez algorytmy zaimplementowane w systemie przechowywane są w dwóch rodzajach struktur - słownikach danych umieszczonych w pamięci oraz macierzach umieszczonych na dyskach twardych, do których dostęp jest przyspieszany dzięki wykorzystaniu buforowania w pamięci podręcznej.

Struktura słowników danych pozwala na pobranie informacji o dowolnym artykule lub kategorii na podstawie ich identyfikatora id w czasie $O(log_M N)$, gdzie: N - liczba elementów w słowniku, M - parametr określający długość listy używanej do indeksowania elementów, został on doświadczalnie dobrany na poziomie ok. 500000.

Wszelkie powiązania między artykułami oraz kategoriami zapamiętywane są w rzadkich macierzach danych. Macierze indeksują relacje obustronnie, tzn. w przypadku gdy reprezentowane są relacje pomiędzy elementami $x \in X$ oraz $y \in Y$, każda relacja R_{ij} między elementem x_i oraz y_j indeksowana jest obustronnie, ze względu na element x_i oraz y_j. Każda macierz podzielona jest na 3 części:

— TOC - indeksującej położenie tablicy relacji do elementu x.
— Tablice elementów ze zbioru Y, które są w relacji do poszczególnych elementów indeksowanych x_i.
— Tablice elementów ze zbioru X, które są w relacji do poszczególnych elementów indekstowanych y_j.

Ze względu na budowę macierzy, koszt znalezienia wartości relacji pomiędzy dowolnymi dwoma elementami w macierzy można wyrazić jako:

$$\overbrace{O(1)}^{\text{TOC}} + \overbrace{O(log_2 M)}^{\text{Przeszukanie}},$$

gdzie M - średnia liczba elementów, z którym dany element jest w relacji.

3. Algorytmy

Proces przetwarzania zapytania użytkownika jest interaktywny, może on być opisany następującym scenariuszem:

1. Do systemu wprowadzone zostaje zapytanie,
2. Zwrócone zostają: artykuły związane z zapytaniem oraz pytanie doprecyzowujące (w postaci nazwy kategorii) ,
3. Po udzieleniu odpowiedzi, algorytm wraca do kroku 2.

Opisany wyżej proces nazywany jest *sesją*. W pierwszym kroku każdej sesji używane są algorytmy wyszukujące treść z bazy danych na podstawie frazy zapytania. Następnie, etap iteracyjny, tj. wyznaczanie klastrów z odpowiedziami oraz procesu doprecyzowującego odbywa się za pomocą dwóch algorytmów: wyboru kierunku konceptualnego oraz algorytmu klasteryzującego DBScan [8].

Rolą algorytmu wyboru kierunku konceptualnego jest wyznaczenie najlepszego uogólnienia, jakie może przedstawić system użytkownikowi w celu zawężenia zbioru rezultatów. Algorytm DBScan służy do klasteryzacji wyników w powiązane tematycznie grupy (klastry), jego rola jest czysto prezentacyjna i nie wpływa na działanie innych algorytmów.

3.1. Użyteczność informacyjna kategorii

Jednym z założeń badanego systemu jest możliwość precyzowania zwracanych przez niego wyników na podstawie otrzymywanych od użytkowników zapytań oraz odpowiedzi. Podstawowym kryterium, które jest mierzone względem każdej kategorii c w zbiorze wszystkich kategorii C jest użyteczność informacyjna. Parametr ten pozwala określić, w jakim stopniu nazwa danej kategorii jest informacyjna dla użytkownika. W tym celu, mierzone są dwa parametry, zbierane na podstawie odpowiedzi użytkowników:

dk_i - określający liczbowo ile razy na dane pytanie dot. kategorii c_i użytkownik nie potrafił określić odpowiedzi.
yn_i - liczba odpowiedzi pozytywnych oraz negatywnych użytkowników, na pytanie dotyczące danej kategorii c_i.

Na podstawie tych dwóch parametrów, określa się użyteczność informacyjną u za pomocą Formuły 1.

$$u(dk, yn) = \begin{cases} 1.0 & \text{jeżeli } dk_i = 0 \vee yn_i = 0 \\ 0.1 & \text{jeżeli } \frac{dk_i}{yn_i + dk_i} < 0.1 \\ \dfrac{dk_i}{yn_i + dk_i} & \text{w innym przypadku} \end{cases} \tag{1}$$

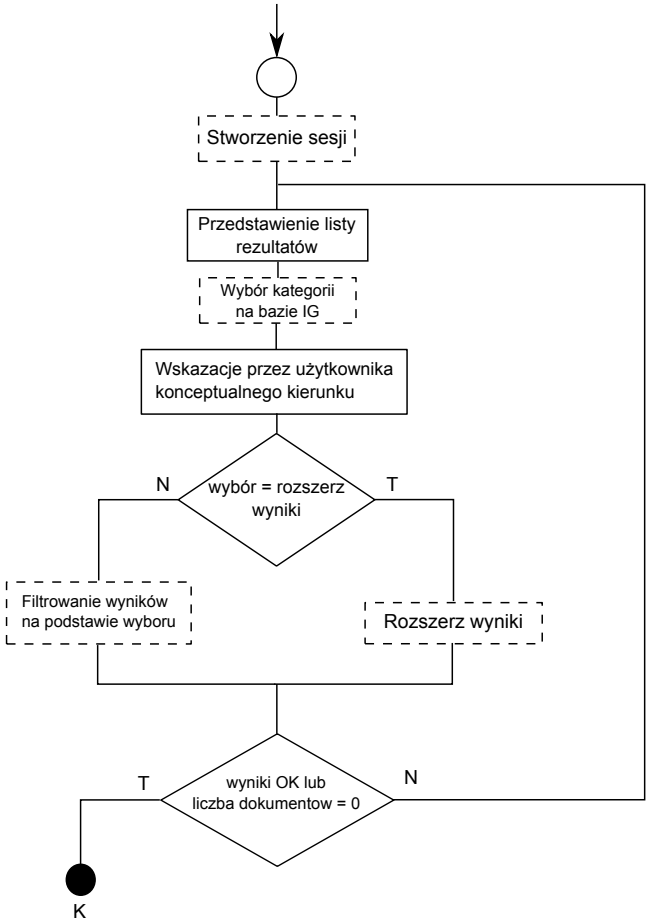
3.2. Precyzowanie rezultatów wyszukiwania

Schemat 1 ilustruje sposób interakcji użytkownika z systemem. Algorytm realizujący ten scenariusz na budowę modułową. Linią przerywaną zostały pokazane na schemacie te części algorytmu, które mogą zostać podmienione. Pojedyncze moduły, które realizują założoną funkcjonalność mogą być zaimplementowane na różne sposoby. Poniżej został opisany sposób ich implementacji, który został zaimplementowany w systemie.

3.2.1. Wybór kierunku konceptualnego

Wybór kierunku konceptualnego polega na wybraniu n najlepiej pasujących zapytań (kierunków konceptualnych) do użytkownika, pozwalających mu określenie tematyki która go interesuje. W implementowanym systemie kierunek konceptualny wyznaczony zostaje poprzez obliczenie zysku informacyjnego, jaki jest osiągnięty po otrzymaniu odpowiedzi od użytkownika. Pseudokod 1 obrazuje działanie omawianego algorytmu.

Algorytm dla każdej kategorii c, która jest związana z dokumentami biorącymi udział w procesie wyznaczania najlepszego kierunku określa parametr oznaczony jako IG_c. W trakcie obliczania zysku informacyjnego, dwukrotnie uwzględniania jest wartość użyteczności informacyjnej kategorii opisana w sekcji 3.1. Niska wartość tego parametru może albo całkowicie wykluczyć daną kategorii z propozycji (jednak szansa nigdy nie jest zerowa) albo zmniejszyć wartość zysku informacyjnego.



Rysunek 1. Schemat blokowy przedstawiający interakcję użytkownika z systemem

Algorytm 1 Pseudokod algorytmu wyboru kierunku konceptualnego

SET $categories \leftarrow AllCategoriesByDocuments(documents)$
for all $category$ in $categories$ **do**
  $r \leftarrow rand(0, 1.0)$
  if r >= $CategoryUsefullness(category)$ **then**
    continue
  end if
  $cc \leftarrow 0$
  for all doc in $documents$ **do**
    if $relation(doc, category) > 0.0$ **then**
      $cc \leftarrow cc + 1$
    end if
  end for
  $$IG_c \leftarrow 1.0 - \left| \frac{(\frac{count(documents)}{2.0} - cc)}{\frac{count(documents)}{2.0}} \right|$$
  $IG_c \leftarrow IG_c * CategoryUsefullness(category)$
end for
return $max_n(categories) ordered by IG_c$

3.2.2. Rozszerzenie zbioru wyników

Proces dający użytkownikowi możliwość rozszerzenia aktualnych wyników wyszukiwania o nowe, powiązane z nimi artykuły.

Zaimplementowany w projekcie algorytm bazuje na powiązaniach typu artykuł – artykuł (odnośnikach umieszczonych w artykułach). Pojedyncze rozszerzenie wyników dodaje do listy rezultatów wszystkie artykuły, które są w ten sposób powiązane z artykułami stanowiącymi zbiór wyjściowy.

3.2.3. Filtrowanie wyników na podstawie wyboru konceptualnego kierunku

Użytkownik inicjując proces wyszukiwania podaje zapytanie tekstowe. Odpowiedź, jaką może udzielić system na tej podstawie może być nie do końca zadowalająca, gdyż po 1. zapytanie może mieć wiele znaczeń i występować w wielu kontekstach, a po 2. użytkownik może chcieć zdobyć informacje nie związane ściśle z podanym hasłem, a jedynie z pojęciami pokrewnymi. Ustalenie konceptualnego kierunku pozwala na odfiltrowanie artykułów niezwiązanych z tym, czego szuka użytkownik, a także zaprezentowaniem nowych, które nie zostały skojarzone z początkowym zapytaniem.

Po każdym wyborze odpowiedzi użytkownika w trakcie sesji, następuje proces filtrowania (odrzucania) artykułów, które nie pasują do budowanego wzorca odpowiedzi. Wzorcem odpowiedzi nazywany jest wektor V składający się ze wszystkich par $kategoria - odpowiedz$. Wektor ten na początku sesji jest pusty i rozwijany jest on w trakcie zbierania kolejnych odpowiedzi od użytkownika. Algorytm filtracji zaimplementowany w systemie przedstawiony został za pomocą pseudokodu 2.

Algorytm 2 Filtrowanie rezultatów

for all doc in $IntialDocuments$ **do**
  $points = ComputeDocumentToAnswerSimilarity(doc, answerHistory)$
  $relPoints = \frac{points}{count(answerHistory)}$
  if $relPoints > 0.0$ **then**
    results $\leftarrow$ **results** $- doc$
  else if $relPoints < 0.0$ **then**
    results $\leftarrow$ **results** $+ doc$
  end if
end for

Punktacja artykułów jest niezależna od algorytmu określania podobieństwa wektora odpowiedzi i wektora opisującego dany artykuł. W prezentowanej pracy implementacja funkcji podobieństwa $ComputeDocumentToAnswerSimilarity$ została zdefiniowana w sposób opisany pseudokodem 3.

Zmienne np oraz yp są parametrami algorytmu, określającymi kary do współczynnika podobieństwa w przypadku odpowiednio braku relacji, gdy wyborem użytkownika była odpowiedź „nie” oraz „tak”. W testowanym systemie zostały one dobrane eksperymentalnie i wynoszą odpowiednio 0.05 dla parametry np oraz -0.4 dla parametru yp.

3.3. Algorytm klasteryzacji – DBScan

Drugim wykorzystywanym w systemie algorytmem jest algorytm klasteryzacji, który nie wpływa bezpośrednio na wyniki zwracane w trakcie sesji użytkownika (procesu interaktyw-

Algorytm 3 Podobieństwo dokumentu i wektora odpowiedzi

```
x ← 0.0
for all choice in answerHistory do
  relation = GetRelation(document, choice)
  if rel = 0.0 then
    if choice = No then
      y ← np
    else
      y ← yp
    end if
  else
    if choice = Yes then
      y ← rel
    else
      y ← −rel
    end if
  end if
  x ← x + y
end for
return x
```

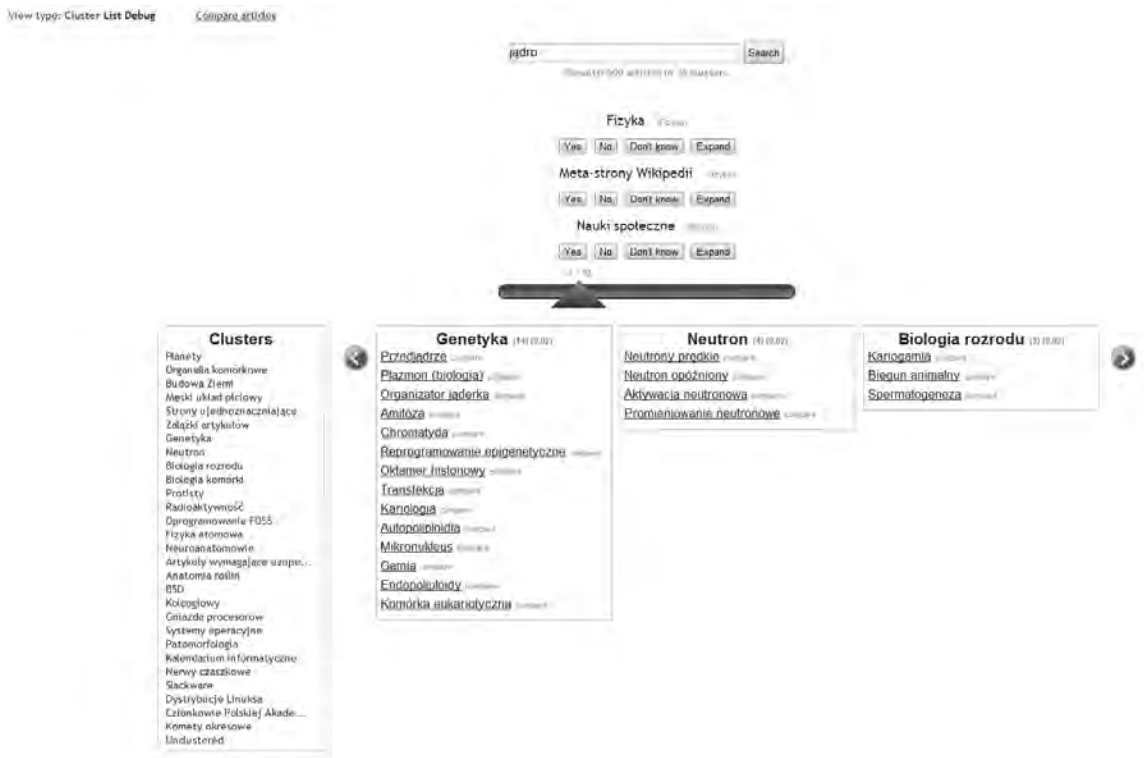
nego szukania). Wykorzystywany jest on w warstwie prezentacji, gdyż pozwala zebrać wyniki w logicznie powiązane zbiory, dzięki czemu są one dużo czytelniejsze dla użytkownika końcowego. W projekcie został użyty algorytm DBScan [8].

Algorytm został wybrany ze względu na kilka jego właściwości. Przede wszystkim, w przeciwieństwie do wielu innych algorytmów, jak np. *k-means* [9], nie wymaga on zdefiniowania liczby klastrów a priori. Jest to istotny czynnik, ze względu na duże różnice w liczności klastrów, które są często niezależne od liczby grupowanych elementów.

Kolejną zaletą algorytmu DBScan jest możliwość wykonania jego optymalizacji. Dzięki zapamiętywaniu części wyników pośrednich, algorytm ten typowo realizowany w złożoności $O(n^2)$ może zostać wykonany w złożoności $O(n * log(n))$, co jest wystarczające, by w czasie rzeczywistym przeliczyć klastry dla zbiorów artykułów o liczebności do tysiąca elementów.

Idea klasteryzacji dobrze wpasowuje się w charakterystykę powiązań między danymi, gdzie każdy artykuł można powiązać ze zbiorem kategorii, co w naturalny sposób je grupuje. Rysunek 2 pokazuje interfejs użytkownika w działaniu po procesie klasteryzacji dla przykładowego zapytania.

Zaimplementowana wersja algorytmu *DBScan* została oparta o kosinusową miarę podobieństwa, która dobrze sprawdza się w zbiorach o dużej liczbie wymiarów [10]. *DBScan* może być również parametryzowany poprzez parametr *epsilon* określający stopień podobieństwa między dwoma obiektami, aby zostały one zaklasyfikowane jako należące do jednego klastra. Cecha ta została wykorzystana w systemie dzięki wprowadzeniu procesu kilkukrotnej klasteryzacji jednego zbioru dokumentów. Dokumenty są klasteryzowane 3-krotnie, z różnymi wartościami parametru $epsilon$. Każdy kolejny przebieg łączy dokumenty, które są ze sobą słabiej skojarzone, dzięki temu najpierw znajdywane są bardziej szczegółowe, specjalistyczne grupy obiektów, następnie pozostałe dokumenty grupowane są w bardziej ogólne klastry. Parametry $epsilon$ zostały ustalone na drodze empirycznych doświadczeń, w aktualnie projekto-



Rysunek 2. Interfejs użytkownika systemu

Algorytm 4 Wielokrotna klasteryzacja

$clusters \leftarrow cluster(documents, t_0)$
for all t_i in T **do**
  for all c in $clusters$ **do**
    if $|c| > maxClusterSize$ **then**
      $R \leftarrow R + cluster$
      $delete(cluster)$
    end if
  end for
  $clusters \leftarrow clusters + cluster(R, t_i)$
end for
for all $cinclusters$ **do**
  if $|c| > maxClusterSize \vee |c| < minClusterSize$ **then**
    $U \leftarrow U + c$
  else
    $cat \leftarrow FindRepresentative(c)$
    $docsByCategory[cat] \leftarrow docsByCategory[cat] + c$
  end if
end for

wanym systemie wynoszą one odpowiednio 0.7, 0.6, 0.5 dla kolejnych przebiegów algorytmu. Algorytm 4 prezentuje w uproszczony sposób realizację powyższych założeń.

4. Uzyskane rezultaty

Z punktu widzenia użytkownika końcowego, najistotniejszym czynnikiem warunkującym odbieraną przez niego jakość działania systemu jest jakość klastrów mu prezentowanych oraz zbieżność procesu interakcyjnego szukania (pojedynczej sesji) z użyciem kierunków konceptualnych do zbioru istotnych wyników.

W celu oceny jakości otrzymywanych klastrów, dla kilku przykładowych zapytań została zmierzona ich koherentność, tj. spójność logiczna artykułów zawartych w pojedynczym klastrze. Parametr ten został zbadany w dwojaki sposób. Pierwszy określał koherentności subiektywnie, poprzez określenie przez człowieka dopasowania zwróconych rezultatów do poszczególnych grup. Drugi test polegał na określeniu współczynnika koherentności obiektywnej pozwalającej uniezależnić ten proces od oceny poprzez ludzi. Dokładny algorytm został opisany w dalszej części artykułu.

Zbieżność procesu szukania została zmierzona na podstawie subiektywnej spójności (koherentności) grupy artykułów istotnych w trakcie procesu doprecyzowywania z użyciem kierunków konceptualnych.

4.1. Koherentność subiektywna

Koherentność subiektywna K_c dla danego klastra c została zdefiniowana Formułą 2.

$$K_c = \frac{|X_c|}{|Y_c|}, \tag{2}$$

gdzie:

X_c - zbiór artykułów w klastrze c, które pasują do niego (na podstawie subiektywnej oceny osoby przeprowadzającej test)
Y_c - zbiór wszystkich artykułów w klastrze c

Współczynnik K_c zdefiniowany jest analogicznie do współczynnika precyzji p w popularnie stosowanej metryce badania jakości klastrów - *F-miarze*[11].

Koherentność dla całego zapytania zdefiniowana jest jako suma ważona koherentności dla wszystkich klastrów względem ich liczebności, zdefiniowane zostały również wartości koherentności minimalnej oraz maksymalnej. Określają one minimalną oraz maksymalną koherentność dla zbioru wszystkich klastrów, które zostały ujęte w zapytaniu. Wyniki koherentności dla trzech wybranych haseł wyszukiwania zostały przedstawione w Tabeli 1, w której podano dodatkowo trzy wartości koherentności zdefiniowane Formułami 3a-c.

$$K = \frac{\sum\limits_{c \in C} K_c * |Y_c|}{\sum\limits_{c \in C} |Y_c|} \tag{3a}$$

$$K_{min} = \min K_c \tag{3b}$$

$$K_{max} = \max K_c \tag{3c}$$

Tabela 1. Zestawienie wyników eksperymentu koherentności subiektywnej

Hasło	**K**	$\mathbf{K_{min}}$	$\mathbf{K_{max}}$	**l. klastrów**	**l. artykułów**	**Odch.std.K**
Widelec	1.000	1.000	1.000	11	124	0.000
Jądro	0.894	0.600	1.000	26	635	0.1071
Niemcy	0.993	0.857	1.000	11	601	0.0410

4.2. Koherentność obiektywna

Koherentności otrzymanych klastrów z wykluczeniem czynnika ludzkiego oceniona została z użyciem algorytmu bazującego na pomiarze minimalnej odległości między artykułami, a ich kategorią grupującą.

Podstawą wyznaczenia tej odległości jest miara relacji pomiędzy artykułami a kategoriami. Jeżeli artykuł a_i należy bezpośrednio do którejś z kategorii k_i, to ich relacja określona jest wartością: $R(a_i, k_i) = 1.0$. Jeżeli między artykułem a kategorią nie zachodzi żadna relacja przynależności to $R(a_i, k_i) = 0.0$. Wartości w przedziale $(0.0, 1.0)$ oznaczają, że artykuł należy do którejś kategorii pośrednio, tzn. że istnieje taki zbiór kategorii K_s, w którym:

1. Jedna z kategorii $k_j \in K_s$ jest bezpośrednio związana z artykułem: $R(a_i, k_j) = 1.0$.
2. Jedna z kategorii $k_j \in K_s$ jest bezpośrednio związana z kategorią k_i - $R(k_i, K_j) = 1.0$.
3. Każda z kategorii ze zbioru K_s jest związana przynajmniej z jedną inną kategorią z tego zbioru bezpośrednio, $R(k_{i1}, k_{i2}) = 1.0$.

gdzie:

Zbiór K_s nazywany jest łańcuchem przynależności artykułu i określany jest dla pary artykuł - kategoria jako $K_l(a, k)$.
Kategoria grupująca jest zdefiniowana jako kategoria k_r, która dla danego podzbioru $A_x \subset A$ spełnia warunek $\forall_{a \in A_x} R(a, k_r) > 0.0$
Kategoria grupująca jest minimalną kategorią grupującą, jeżeli w zbiorze K_{ss}, który jest sumą wszystkich łańcuchów przynależności pomiędzy artykułami ze zbioru A_x a k_r nie istnieje inna kategoria grupująca dla zbioru A_x.

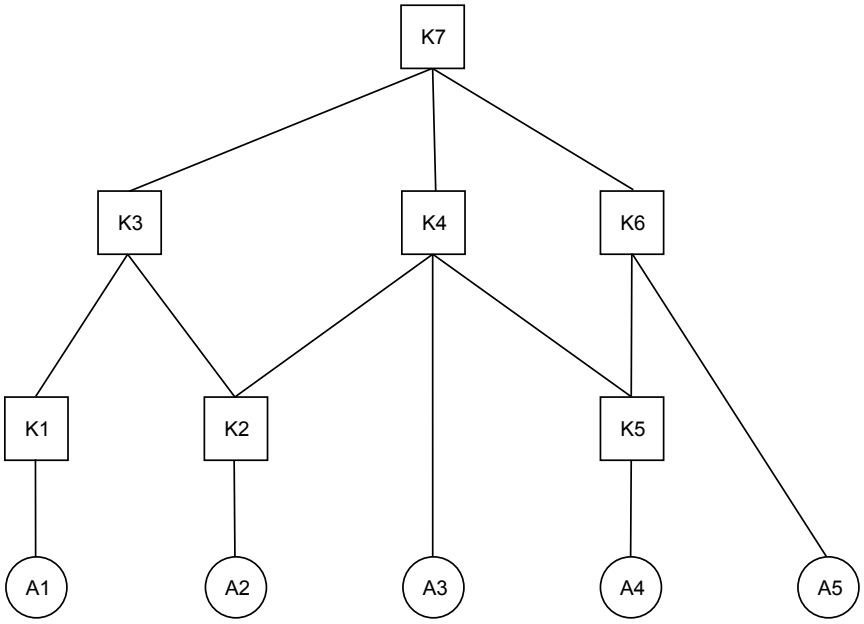
Rysunek 3 przedstawia przykładowy zbiór artykułów i kategorii ilustrujący przedstawione powyżej pojęcia. Artykuły zostały oznaczone symbolami $A1...A5$, kategorie - $K1...K7$. Krawędzie grafu oznaczają relacje o wartościach > 0.0. W podanym przykładzie, artykuł $A2$ jest silnie powiązany kategorią $K2$, a także poprzez łańcuchy przynależności z kategoriami $K3, K4, K7$. Artykuł $A3$ posiada 1 silną relację z kategorią $K4$, oraz relacje pośrednie z kategorią $K7$. Jeżeli zdefiniujemy podzbiór artykułów, dla których wyznaczone mają zostać kategorie grupujące w postaci zbioru $A2, A3$, to zgodnie z definicją będą to kategorie ze zbioru $K4, K7$, które stanowią część wspólną ich łańcuchów przynależności. Minimalną kategorią grupującą jest kategoria $K4$, ponieważ nie istnieje taki łańcuch przynależności, który byłby wspólny dla artykułów $A2, A3$ i zawierał w sobie inną kategorię grupującą. Warto nadmienić, że kategoria $K4$ jest właśnie taką kategorią dla kategorii $K7$, dlatego kategoria $K7$ nie jest minimalną kategorią grupującą.

Opisywany eksperyment polegał na zdefiniowaniu minimalnych kategorii grupujących dla każdego otrzymanego klastra (rozumianego jako zbiór artykułów), a następnie wyliczeniu ważonych odległości między artykułami a kategoriami grupującymi. Otrzymane wyniki zostały zebrane w Tabeli 2.

Zebrane wyniki pokazują, że dla wielu utworzonych klastrów, odległość artykułów od minimalnej kategorii grupującej wynosi 1.0000. Oznacza to, że wszystkie artykuły należą

Tabela 2. Zestawienie wyników eksperymentu koherentności obiektywnej

Hasło	Artykuły	Minimalna kategoria grupująca	Odległość
Widelec	-Brudny widelec -A teraz coś ... -Latający cyrk Monty Pythona	Monty Python	1.3333
	-Widelec -Łyżkowidelec -Widelczyk -Chochla -Sztućce -Łyżka	Sztućce	1.0000
	-Widelec rowerowy -Amortyzacja rowerowa -Mostek rowerowy -Kierownica rowerowa -Części rowerowe -Rama rowerowa	Części rowerowe	1.0000
	-Piotr Mocarski -Jacek Janowicz -Krzysztof Szubzda	Polscy artyści kabaretowi	1.0000
	-Kabaret Widelec -Gable -Niezbędnik -(63 innych)	Nauka	7.5693
	-Manewr Worka -Belladonna coup	Rozgrywka w brydżu	1.0000
	-Lech -FIS -WFM -MOJ 130 -Perkun -Podkowa -WSK -SHL -WSK M21W2 S2	Polskie motocykle	1.2222
	-BMW R 17 -BMW R-23 -(14 innych)	Motocykle BMW	1.0000
	-Honda CB 600F Hornet -Honda CBR 125R -Honda FMX 650	Motocykle Honda	1.0000
	-Romet 760 -Romet 210 -Komar	Motorowery Romet	1.0000
	-Lista odcinków serial Latający cyrk Montty Pythona -Lista odcinków serialu animowanego Yin Yang Yo! -Lista odcinków serialu Czarodzieje z Waverly Place	Listy odcinków	1.0000



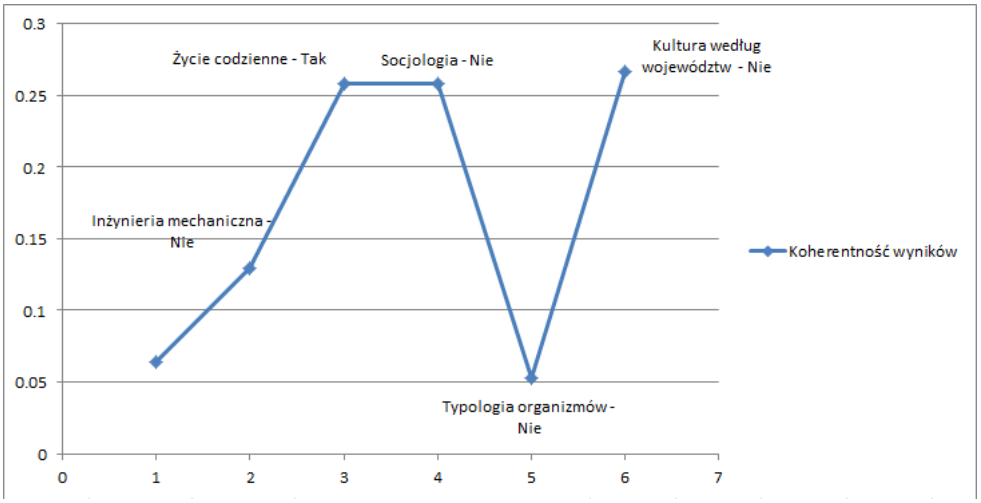
Rysunek 3. Przedstawienie przykładowego zbioru artykułów oraz kategorii w kontekście oceny koherentności obiektywnej

bezpośrednio do danej kategorii, czyli te klastry są bardzo silnie powiązane logicznie. Dla klastrów większych rozmiarów, które obejmują dużo szerszy kontekst (jak np. klaster „Nauka") odległość rośnie. Jest to efekt wytworzenia się wielopoziomowej struktury artykułów. Duża odległość oznacza, że pewne artykuły powinny były zostać zaklasyfikowane w oddzielne, bardziej specjalistyczne klastry. Badania pokazują, że ten problem pojawia się dla grup artykułów, które tworzą klastry o rozmiarze powyżej 60-ciu elementów.

4.3. Zbieżność koherentności algorytmu wyboru kierunku konceptualnego

Eksperyment ten polega na zmierzeniu jakości zbieżności otrzymywanych wyników (reprezentowanych przez klastry) w procesie interaktywnego szukania przez użytkownika. Badana jest w szczególności jakość algorytmu wyboru kierunku konceptualnego, który odpowiada za procedurę filtrowania wyników w procesie pozyskiwania odpowiedzi od użytkownika. Badany parametr jest parametrem subiektywnym, ponieważ wymaga określenia oczekiwanego docelowego zbioru wyników uzyskanych dla zadanego zapytania. Określenie zbieżności rezultatów z użyciem algorytmu wyboru kierunku konceptualnego, przeprowadzone zostało poprzez realizację następującego scenariusza:

— Systemowi zadawane jest zapytanie hasłowe, jednocześnie określana jest (nie na podstawie zwróconych wyników) grupa interesujących artykułów, które tematycznie pasują do danej grupy.
— Określana jest koherentność wyników, jako iloraz artykułów pasujących tematycznie do poszukiwanej grupy i wszystkich artykułów zwróconych przez system.
— Wybierane są najlepiej (zdaniem użytkownika) pasujące odpowiedzi do pytań zadawanych przez system, dla każdego kroku badana jest koherentność wyników.
— Proces jest zatrzymywany po osiągnięciu zadowalającego progu dokładności (może być inny dla każdego zapytania).



Rysunek 4. Wykres zbieżności rezultatów z użyciem wyboru kierunków konceptualnych dla hasła Widelec

— Zbieżność koherentności jest obliczona wzorem: $\frac{koh_e - koh_s}{steps}$
gdzie:
koh_e - koherentność otrzymana na końcu badania,
koh_s - koherentność początkowa,
$steps$ - liczba odpowiedzi udzielona systemowi.
Należy zauważyć, że wpływ na współczynnik zbieżności ma zarówno poprawa jakości otrzymanych wyników poprzez odfiltrowanie jak największej liczby nieskojarzonych artykułów oraz nie odrzucanie docelowych, jak i szybkość osiągania zbieżności.

Na Rysunku 4 przedstawiono wykres zbieżności wyników w procesie interaktywnego szukania dla testowej fraza zapytania *Widelec*. Początkowa liczba artykułów zwróconych przez zapytanie wynosiła 124, z czego wynika szybka zbieżność koherentności. Jak widać na wykresie przedstawionym na Rysunku 4, odpowiedź na pierwsze 2 zapytania zwiększyła koherentność ok 5-krotnie, głównie poprzez zmniejszenie liczby artykułów z 124 do 31. Kolejna odpowiedź nie zmieniła wyniku w znaczący sposób. Odpowiedź czwarta, dot. typologii organizmów pokazuje, że nieprawidłowe zaklasyfikowanie haseł, w tym przypadku, sztućce należały pośrednio do tej kategorii, może powodować znaczne obniżenie jakości otrzymanych wyników, piąta odpowiedź przywróciła jednak wszystkie artykuły odrzucone wcześniej, poprawiając wynik. Dla tych danych, zbieżność liczona dla trzech pierwszych odpowiedzi wynosi 0.0645, natomiast dla wszystkich pięciu 0.0404.

5. Dalszy rozwój systemu

Opisywany system w obecnej fazie stanowi kompletną implementację silnika wyszukiwarki z prostym interfejsem użytkowym. Prototyp systemu wdrożony został pod adresem http://bettersearch.eti.pg.gda.pl.

Głównym kierunkiem rozwoju systemu jest przede wszystkim zwiększenie jego wydajności, rozumianej nie tylko jako optymalizację istniejącego kodu, ale również rozwiązania systemowe, jak zmniejszenie złożoności obliczeniowej niektórych algorytmów, przede wszystkim algorytmu klasteryzacji. Aktualna implementacja osiąga limit swoich możliwości dla baz

danych wielkości polskiej Wikipedii, czyli ok. 0.5 miliona artykułów. Docelowa wydajność silnika to obsługa baz danych o rozmiarze co najmniej 5 miliona artykułów.

Pod względem badawczym, istotną cechą platformy jest w jak największym stopniu zautomatyzowany system pozwalający testować jakość użytych algorytmów, co ma szczególne znaczenie ze względu na wysoką modułowość oraz parametryzowalność systemu. Takie rozwiązanie zostało na tym etapie rozwoju pominięte.

System w aktualnej fazie działa wyłącznie na bazie artykułów pochodzących z Wikipedii. Kolejnym etapem jego rozwoju jest umożliwienie importowania dowolnych danych tekstowych. Umożliwi to zastosowanie go w praktycznie dowolnych dziedzinach, jak ogólna wyszukiwarka internetowa, czy specjalizowany silnik działający dla zadanej dziedziny.

Podziękowania. Niniejsza praca, zrealizowana została w ramach projektu Synat - Stworzenie uniwersalnej, otwartej, repozytoryjnej platformy hostingowej i komunikacyjnej dla sieciowych zasobów wiedzy dla nauki, edukacji i otwartego społeczeństwa wiedzy, finansowanej przez Narodowe Centrum Badań i Rozwoju nr umowy SP/I/1/77065/10.

Bibliografia

[1] C. Clarke, G. Cormack. *Dynamic inverted indexes for a distributed full-text retrieval system. Ws ec u R eport CS-95-0*, 10, 1995.

[2] S. Deerwester, S. Dumais, G. Furnas, T. Landauer, R. Harshman. *Indexing by latent semantic analysis. Journal of the American society for information science*, 41(6):391–407, 1990.

[3] L. Page, S. Brin, R. Motwani, T. Winograd. *The PageRank citation ranking: Bringing order to the web.* 1999.

[4] D. Harman. *Ranking algorithms. Information retrieval: data structures and algorithms*, pp. 363–392, 1992.

[5] J. Kleinberg. *Authoritative sources in a hyperlinked environment. Journal of the ACM (JACM)*, 46(5):604–632, 1999.

[6] P. Hansen, K. Järvelin. *Collaborative information retrieval in an information-intensive domain. Information Processing & Management*, 41(5):1101–1119, 2005.

[7] J. Szymanski. *Mining Relations between Wikipedia Categories*. F. Zavoral, J. Yaghob, P. Pichappan, E. El-Qawasmeh, editors, *Networked Digital Technologies - Second International Conference, NDT 2010, Prague, Czech Republic, July 7-9, 2010. Proceedings, Part II*, vol. 88 of *Communications in Computer and Information Science*, pp. 248–255. Springer, 2010.

[8] M. Ester, H. Kriegel, J. Sander, X. Xu. *A density-based algorithm for discovering clusters in large spatial databases with noise. Proceedings of the 2nd International Conference on Knowledge Discovery and Data mining*, vol. 1996, pp. 226–231. AAAI Press, 1996.

[9] J. Hartigan, M. Wong. *Algorithm AS 136: A k-means clustering algorithm. Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 28(1):100–108, 1979.

[10] P. Tan, M. Steinbach, V. Kumar, others. *Introduction to data mining*. Pearson Addison Wesley Boston, 2006.

[11] R. Baeza-Yates, B. Ribeiro-Neto, others. *Modern information retrieval*, vol. 82. Addison-Wesley New York, 1999.

A review of process metrics in defect prediction studies

Marian Jureczko[1], Lech Madeyski[2]

[1]*Wroclaw University of Technology, Poland*
[2]*Institute of Informatics, Wroclaw University of Technology, Poland*

Abstract:

Process metrics appear to be an effective addition to software defect prediction models usually built upon product metrics. We present a review of research studies that investigate process metrics in defect prediction. The following process metrics are discussed: Number of Revisions, Number of Distinct Committers, Number of Modified Lines, Is New and Number of Defects in Previous Revision. We not only introduce the definitions of the aforementioned process metrics but also present the most important results, recent advances and the summary regarding the use of these metrics in software defect prediction models, as well as the taxonomy of the analysed process metrics.

Keywords:

defect prediction, software metrics, process metrics

1. Introduction

Process metrics, although sometimes not easy to collect, are becoming the crucial ingredients of novel, more accurate defect prediction models and systems. This paper reviews research activities regarding the use of process metrics in modern software defect prediction models. We follow the definitions of process metrics that was given by Henderson-Sellers' [1], who distinguished between product and process metrics. According to Henderson-Sellers product metric refers to software "snapshot" at a particular point of time, while process metric reflects the changes over time, e.g. the number of code changes. Recently the term historical metrics is sometimes used instead of process metrics, e.g. [2]. Nevertheless, we decided to use the traditional nomenclature which draws on Henderson-Sellers' [1], as well as Kan's nomenclature [3] and is consistent with one we assumed before [4]. We provide both, the definitions of the process metrics (Section 2), as well as presentation and discussion (in subsequent sections) of the most important results obtained in a wide range of empirical research studies regarding defect prediction models and process metrics. The studies are summarized in Section 8 and the conclusions are discussed in Section 9.

2. Metrics definitions

Number of Revisions (NR). The NR metric represents the number of revisions of a given Java class during development of the investigated release of a software system.

Number of Distinct Commiters (NDC). The NDC metric counts the number of distinct authors, usually developers, who committed their changes in a given Java class during the development of the investigated release of a software system.

Number of Modified Lines (NML). The value of the NML metric is equal to the sum of all lines of source code which were added or removed in a given Java class. Each of the revisions which were committed during the development of the investigated release of a software system is considered. According to the CVS version control system, a modification in a given line of source code is equivalent to removing the old version and subsequently adding a new version of the line.

Is New (IN). It is a nominal metric. It shows whether the given class existed in the previous version of the investigated system or whether it is a new one.

Number of Defects in Previous Version (NDPV). The NDPV metric counts the number of defects which were repaired in a given class during the development of the previous release of a software system.

3. Number of Revisions

There are several studies which investigate the number of historical revisions as a defect indicator. Weyuker et al. [5, 6, 7, 8, 9] used *the number of changes in the prior release*, as well as several other metrics, and created a very efficient defect prediction model by means of negative binomial regression. The 20% of files selected by the model contained up to 92% of the defects. They found in [6] that the files which were changed in the previous release had about three times as many faults as detected in the unchanged files.

Ratzinger et al. [10] used the number of historical revisions in three different types of defect prediction models and evaluated them on five industrial projects.

Graves et al. [11] concluded that the change history contains more useful information than could be obtained from the product (size and structure) metrics. They found specifically that the numbers of lines of code of a module (a metric widely used in the defect prediction models) are not helpful in predicting faults when the number of times a module was changed is taken into account.

Schröter et al. [12] mined the Eclipse bug and version databases and calculated the correlations of process measurement with pre- and post-release failures. In the case of pre-release failures the *number of changes* had the highest correlation coefficient among all the investigated process and product metrics. It was .34–.47 in the case of the Pearson correlation and .44–.56 in the case of the Spearman correlation. The work was later extended by using the metric in commercial projects [13] and by defining a metric which represents series of changes [14].

Illes-Seifert and Paech [2] investigated a number of process metrics, among others *Frequency of Change*. The Spearman's correlation of the *Frequency of Change* metric with number of defects was high in all nine investigated projects (.43–.64). The metric was recommended as a very good defect indicator.

Shihab et al. [15] investigated the *Eclipse* project in order to identify the metric subset that will be as good as a full metric set. 34 different metrics were analysed. The authors reduced the number of metrics to a much smaller, statistically significant and minimally collinear subset. The subset contained four metrics including the NR metric. The small set of metrics was used to build logistic regression models. The obtained models were compared to the models that make use of the full set of metrics and a very little difference in prediction accuracy was found.

Moser et al. [16] conducted a comparative analysis of the efficiency of the process and the product metrics for defect prediction. The process metrics set contained metrics similar

to NR, NDC, NML, NDPV and IN. The authors built three types of models: process metric models, product metric models and combined models. The process metric models and the combined models were more efficient than the product metrics models. Therefore, the authors recommended using the process metrics in defect prediction.

4. Number of Distinct Commiters

The metric is considered in several studies. Weyuker et al. used the number of distinct developers in a defect prediction model in [5] and then investigated the relevance of the metric in [8] and [9]. The experiment included the addition of the developer's information to the defect prediction model, which resulted in a slight improvement of the prediction efficiency. Without the developer information, the model was able to identify correctly 20% of the files containing 76.4–93.8% of the defects. Inclusion of the developer information yielded 76.4–94.8%. Weyuker et al. investigated three large industrial systems with a succession of releases over years of development.

Ratzinger et al. [10] included the number of developers in their defect prediction models. The authors did not study the usefulness of the NDC metric per se. Nevertheless, they concluded that it is not the size and complexity measures that dominate defect-proneness but rather many people-related issues.

The *Number of developers* metric was also used by Zimmermann et al. [17]. Zimmermann et al. analysed twelve open-source and industrial projects in order to assess the possibility of a successful cross project defect prediction.

Different results were obtained by Graves et al. [11]. A study of the code from a 1.5 million line subsystem of a telephone switching system gave no evidence that a large number of developers working on a module caused it to be more faulty. The generalized linear models were used to assess the usefulness of the metric.

Schröter et al. [12] analysed the Eclipse bug and version database and found a high correlation coefficient of the number of authors metric with pre- and post-release failures. It was .15–.41 in the case of the Pearson correlation and .13–.49 in the case of the Spearman correlation. The findings were confirmed by Illes-Seifer and Paech [2]. The Spearman's correlation coefficients of the number of distinct authors with the number of defects vary from .16 to .74 in the nine software projects which were investigated by lles-Seifer and Paech.

The most sophisticated studies on developer-based defect predictions were described in [18] and [19]. Both papers present the analysis of the relationship between a given developer and the density of defects. The usefulness of that approach cannot be actually estimated, since the papers come up with conflicting results.

Nagappan et al. [20] considered the metrics which describe organizational structure, among others *Number of Engineers*. The authors used the metrics to make predictions for *Windows Vista*. The obtained models had a precision and recall of 86.2% and 84.0%, respectively.

5. Number of Modified Lines

Considerable research has been performed on the extent to which the number of modified lines of code impacts the defect counts. The metric was already investigated in 1996 by Khoshgoftaar et al. [21]. The authors investigated two consecutive releases of a large legacy software system for telecommunications and employed 16 static software product metrics and the and the number of added or modified code lines in the prediction.

Layman et al. [22] defined four metrics that depend on the number of modified blocks:

— *Churn = NewBlocks + ChangedBlocks*;
— *RelativeChurn = (NewBlocks + ChangedBlocks) / TotalBlocks*;
— *Deleted churn = Deleted blocks / Total blocks*;
— *NCD churn = (NewBlocks + ChangedBlocks) / DeletedBlocks*.

The models which identify fault-prone components were built using the four aforementioned, as well as several other, structure metrics. Principal Component Analysis and Stepwise Logistic Regression were used in the construction procedure. The prediction accuracy of the models was very high: 73.3–86.7%. The authors used them in *Windows Server 2003* [23] and *Windows Vista* [20, 14] projects.

Purushothaman et al. [24] investigated the number of lines that were changed and described the distribution of modification size across files in a large, written in C/C++, telephone switch which consisted of 50 subsystems. They found out that the probability of a one-line change introducing one error was less then 4 percent.

A NML related metric was also investigated and advocated by Nagappan and Ball [23]. They concluded that it was an excellent predictor of defect density in a large industrial software system.

The finding was later partly confirmed by Zimmermann et al. [17] through investigating twelve open-source and industrial software projects and by Ratzinger et al. [10], who examined five industrial projects.

Śliwerski et al. [25] investigated *Eclipse* and *Mozilla*, and found that the larger the modification, the greater is the defect introduction probability.

Hassan [26] investigated five large open-source projects. He used the size of modification to define module entropy and subsequently used the entropy to predict successfully defects in the modules.

Giger et al. [27] employed in defect prediction a metric derived from the NML, namely fine-grained source code changes (SCC). The SCC metric captures the semantics of changes and was suggested by Fluri et al. [28, 29]. The SCC metric was investigated and compared with the NML metric with regard to its correlation with the number of defect as well as the prediction power (in univariate model). It was an empirical study conducted on 15 Eclipse plug-ins. The results showed significant strong correlation between defects and SCC. Furthermore, the SCC metric had a stronger correlation with defects than the NML metric. The defect prediction was investigated with respect to two different prediction aims: classifying a file as defect-prone or not defect-prone and predicting the number of defects. Based on the obtained results the authors concluded that SCC is a good predictor for defect-prone and not defect-prone files. Moreover, the SCC based models outperformed the models build with the NML metric in both aforementioned cases. Nevertheless, the authors admitted that from external validity point of view their work could be biased by the sole focus on Eclipse projects. A further study with regard to the SCC metric was presented in [30]. The set of investigated projects was extended to 16 Eclipse plug-ins. The authors defined a Gini coefficient that reflects code ownership (i.e. developers were used as the ”population” and NR, NML or SCC as the ”wealth”), hence the obtained metrics are similar to the NDC metric to some extent. The Gini coefficients calculated for different metrics (i.e NR, NML and SCC) were investigated with regard to the correlation with defects and prediction power (in univariate model). The correlation coefficients for NR related metrics varies from -0.05 to -0.79 (median=-0.55); NML from -0.34 to -0,75 (median=-0.54) and SCC from -0.29 to -0.76 and according to the conduced tests the difference between correlations were not significant. Similar results (i.e. no significant difference) were obtained in the case of prediction.

6. Is New

The IN metric as well as the age of a class or file were considered in several studies [5, 31, 11, 2, 32, 33, 7, 9], but the definitions used in those works are not uniform and sometimes differ from ours.

Ostrand et al. [33, 31] defined two file categories: *New File* and *Old File*; Illes-Seifer and Paech [2] defined three categories: *Newborn*, *Young*, *Old*. In [5] the *File age* means the number of months a given file has existed, in [7] and [9] — the number of consecutive prior releases in which the file appeared. Graves et al. [11] measured the average age of the code. They calculated the weighted average of the dates of the changes to the module, weighted by the size of the change.

Khoshgoftaar et al. [32] defined two metrics which were connected with the age: *Is New* (1 - If the module did not exist in the "ending" version (accepted by the customer or released for operational testing) of prior build, 0 - Otherwise) and *Age* (0 - If the module is new, 1 - If the module was new in the prior build, 2 - Otherwise).

The age of a class or a file was usually reported as a good defect count indicator. However, the explanation of the nature of the relation between the age and the defect count differs among studies.

Ostrand et al. [33, 31] successfully used the *Is New* metric in their model and discovered that in every release the percentage of the faulty new files is larger than the percentage of the faulty pre-existing files. The finding was confirmed by Bell et al. [5] — an exploratory analysis indicated that fault-proneness decreases with the age of a file. Similar findings were also obtained by Weyuker et al. [7, 8].

Khoshgoftaar et al. [32] observed that *Is New* varied in their significance and *Age* was a variable with a negative coefficient with the number of defects.

The result supports the finding that older modules are more reliable.

Different results were obtained by Illes-Seifert and Peach [2]. They concluded that "a file's age is a good indicator for its defect count (...), but *Newborn* and *Young files* are not the most fault-prone files".

Graves et al. [11] found that the *Age* metric, when combined with the number of deltas, greatly improved the fit of a defect prediction model to a point where it was slightly better than the reference model.

7. Number of Defects in Previous Version

Another issue to which extensive research was devoted is the extent to which the number of defects from the previous version impacts the defect counts in the current version.

Wahyudin et al. [34] constructed a Reliability Growth Model using the defects from previous version and successfully applied the model to a new release.

Arisholm and Briand [35] scrutinized the number of faults corrected in release $i-1$ and the number of faults corrected in release $i-2$, as well as several other metrics in a large, object-oriented, evolving legacy software system. They build a multivariate prediction model using logistic regression on log-transformed variables. The obtained predictions were correct in more than 80% cases.

Ostrand et al. [31, 7, 9] used negative binomial regression to develop models which predict faults in software systems. One of the metrics employed in the project was the square root of the number of faults identified during the previous release. The authors obtained very good prediction efficiency, as it was reported above.

Graves et al. [11] described a study using the fault history of the modules of a large telecommunications system. The authors constructed several defect prediction models. One of the models predicted the number of faults to be a constant multiple of the number of faults that had been found in a period of time in the module. The model was not the best one but the authors found it challenging to improve it.

Kim et al. [36] analysed seven open-source software systems and developed a cache based algorithm to predict future faults. Two of the principles behind the algorithm were connected with the NDPV metric: *temporal locality* (If an entity introduces a fault recently, it will tend to introduce other faults soon) and *spatial locality* (If an entity introduced a fault recently, "nearby entities" (in the sense of logical coupling) will also tend to introduce faults soon). The algorithm was able to cover 73%–95% of faults by selecting 10% of the most fault prone source code files.

Khoshgoftaar et al. [32] analysed a real-time military system and concluded that modules with faults in the past are likely to have faults in the future.

Ostrand and Weyuker [33] investigated whether faultiness persists between the subsequent releases and, according to the authors, there is moderate evidence that files remain high-fault till later releases. 17% to 54% of the high-fault files of release i are still high-fault in release $i + 1$.

Gyimothy et al. [37] calculated the correlations between the numbers of defects associated with the different versions of classes (Mozilla versions 1.0 to 1.6 were analysed). The correlations were on a high level and varied from .69 to .9.

Contradictory results were obtained by Illes-Seifert and Peach [2]. The authors concluded that, according to the results of their correlation analysis, "the number of defects found in the previous release of file does not correlate with its current defect count."

One might also doubt whether the NDPV metric is useful in defect prediction once the results obtained by Schröter et al. [12] are taken into consideration. Schröter et al. [12] calculated the correlation between pre- and post-release failures, but the obtained coefficients were smaller than the correlation coefficients calculated from the two metrics mentioned before (NR and NDC). Only three process metrics were considered in [12].

The analysis by Shihab et al. [15] showed that the NDPV metric is relevant in defect prediction.

8. Summary

The summary of the studies on the use of process metrics in defect prediction is presented in Table 1. The goals of each study are denoted as prediction (represents defect prediction improvement) or correlation (represents the investigation of the correlation with number of defects), while recommendations are denoted as: "↗" (represents positive recommendation), "↘" (represents negative recommendation) or "–" (no recommendation).

There is a number of works that recommend process metrics. However, it could be risky to arrive at a very optimistic conclusion, since the studies employed many different techniques and investigated a limited number of software projects, which influences their external validity. Furthermore, the positive recommendations describe only the fact that the authors find the metric useful in defect prediction. Specifically, a metric can be recommended in 2 studies but in the first one — as positively correlated, and in the second one — as negatively correlated with the number of defects. Nevertheless, that is a promising direction for further research,

and the level of prediction accuracy of the obtained results makes them potentially useful when employed in industry.

Table 1: Using process metrics in defect prediction: a summary of studies

Metric	Paper	Data sets	Goal of study	Recommendation
NR	Bel et al. [5]	proprietary systems over a 2.5 year period	prediction; correlation	↗
	Ostrand et al. [6]	13 releases of a proprietary system	prediction	–
	Weyuker et al. [7]	35 release of a proprietary system	prediction	↗
	Weyuker et al. [8]	35 release of a proprietary system	prediction using developer information	–
	Weyuker et al. [9]	3 proprietary systems with 17, 9 and 35 respectively and 1 system without formal releases	prediction	–
	Ratzinger et al. [10]	a proprietary system	Q-Mine: Predicting Short-Term Defects for Software Evolution	–
	Graves et al. [11]	subsystem of a telephone switching system	prediction	↗
	Schröter et al. [12]	Eclipse	correlation	–
	Illes-Seifert and Paech [2]	9 open-source projects	correlation	↗
	Shihab et al. [15]	Eclipse	prediction	↗
	Moser et al. [16]	Eclipse	prediction	↗
	Kim et al. [36]	7 open–source projects	prediction	–
	Gao et al. [38]	a proprietary system	methods of limiting the metrics set	↗
	Jureczko [39]	48 releases of 15 open-source and 38 releases of 7 proprietary projects	prediction; correlation	↗
NDC	Weyuker et al. [8]	35 release of a proprietary	prediction using developer information	↗
	Weyuker et al. [9]	3 proprietary systems with 17, 9 and 35 respectively and 1 system without formal releases	prediction	↗
	Ratzinger et al. [10]	a proprietary system	Q-Mine: Predicting Short-Term Defects for Software Evolution	–
	Zimmermann et al. [17]	4 open-source and 7 proprietary projects	cross-project defect prediction	–
	Graves et al. [11]	subsystem of a telephone switching system	prediction	↘
	Schröter et al. [12]	Eclipse	correlation	–
	Matsumoto et al. [18]	Eclipse	the effect of developer features	↗
	Weyuker et al. [19]	16 releases of a proprietary system	improving defect prediction using developer information	–

Continued on next page

Table 1 – *Continued from previous page*

Metric	Paper	Data sets	Goal of study	Recommendation
	Nagappan et al. [20]	Windows Vista	improving defect prediction using organizational complexity	↗
	Illes-Seifert and Paech [2]	9 open-source projects	correlation	↗
	Moser et al. [16]	Eclipse	prediction	↗
	Bell et al. [40]	3 proprietary systems, each containing 18 releases	improving defect prediction using developer information	–
	Jureczko [39]	48 releases of 15 open-source and 38 releases of 7 proprietary projects	prediction; correlation	↗
NML	Khoshgoftaar et al. [21]	2 releases of proprietary system	prediction	↗
	Layman et al. [22]	proprietary system	prediction	↗
	Nagappan and Ball [23]	Windows Server 2003	prediction	↗
	Nagappan et al. [20]	Windows Vista	improving defect prediction using organizational complexity	–
	Nagappan et al. [14]	Windows Vista	prediction	↗
	Purushothaman et al. [24]		distribution of defects with regard to change size	↗
	Zimmermann et al. [17]	4 open-source and 7 proprietary projects	cross-project defect prediction	–
	Ratzinger et al. [10]	a proprietary system	Q-Mine: Predicting Short-Term Defects for Software Evolution	–
	Śliwerski et al. [25]	Mozilla, Eclipse	prediction	↗
	Hassan [26]	6 open–source projects	prediction	↗
	Giger et al. [27]	15 Eclipse plug–ins	prediction	↗
	Giger et al. [30]	16 Eclipse plug–ins	prediction	–
	Graves et al. [11]	subsystem of a telephone switching system	prediction	↗
	Moser et al. [16]	Eclipse	prediction	↗
	Bell et al. [41]	18 releases of a proprietary system	prediction	↗
	Shin et al. [42]	2 open–source projects	predicting locations of vulnerable code	↗
	Sisman et al. [43]	AspectJ	prediction	↗
	Krishnan et al. [44]	Eclipse	prediction	↗
	Khoshgoftaar et al. [32]	subsystem a real time military system	prediction	↗
	Gao et al. [38]	a proprietary system	methods of limiting the metrics set	↗
	Jureczko [39]	48 releases of 15 open-source and 38 releases of 7 proprietary projects	prediction; correlation	↗
IN	Ostrand et al. [33]	13 releases of a proprietary system	distribution of faults with regard to files age	↗

Continued on next page

Table 1 – *Continued from previous page*

Metric	Paper	Data sets	Goal of study	Recommendation
	Ostrand et al. [31]	26 releases of 2 proprietary system	prediction	–
	Illes-Seifert and Paech [2]	9 open-source projects	correlation	–
	Bel et al. [5]	proprietary systems over a 2.5 year period	prediction; correlation	–
	Weyuker et al. [7]	35 release of a proprietary	prediction	↗
	Weyuker et al. [8]	35 release of a proprietary	prediction using developer information	–
	Weyuker et al. [9]	3 proprietary systems with 17, 9 and 35 respectively and 1 system without formal releases	prediction	–
	Graves et al. [11]	subsystem of a telephone switching system	prediction	↗
	Khoshgoftaar et al. [32]	subsystem a real time military system	prediction	–
	Ostrand et al. [6]	13 releases of a proprietary system	prediction	–
	Moser et al. [16]	Eclipse	prediction	↗
	Kim et al. [36]	7 open–source projects	prediction	–
NDPV	Wahyudin et al. [34]	Apache MyFaces	prediction; correlation	↗
	Arisholm and Briand [35]	proprietary system	prediction	↗
	Ostrand et al. [33]	13 releases of a proprietary system	persistence of high–fault files	↗
	Ostrand et al. [31]	26 releases of 2 proprietary system	prediction	–
	Weyuker et al. [7]	35 release of a proprietary	prediction	↗
	Weyuker et al. [9]	3 proprietary systems with 17, 9 and 35 respectively and 1 system without formal releases	prediction	–
	Graves et al. [11]	subsystem of a telephone switching system	prediction	–
	Kim et al. [36]	7 open–source projects	prediction	↗
	Khoshgoftaar et al. [32]	subsystem a real time military system	prediction	↗
	Gyimothy et al. [37]			
	Illes-Seifert and Paech [2]	9 open-source projects	correlation	–
	Schröter et al. [12]	Eclipse	correlation	–
	Shihab et al. [15]	Eclipse	prediction	↗
	Ostrand et al. [6]	13 releases of a proprietary system	prediction	–
	Moser et al. [16]	Eclipse	prediction	↗
	Sisman et al. [43]	AspectJ	prediction	↗
	Gao et al. [38]	a proprietary system	methods of limiting the metrics set	↗
	Jureczko [39]	48 releases of 15 open-source and 38 releases of 7 proprietary projects	prediction; correlation	↗

9. Conclusions and Further Work

On the basis of the review of process metrics presented in this paper we may set forth a taxonomy of the metrics according to the information sources required to calculate them. It is presented in Table 2.

Table 2. Taxonomy of process metrics

Metric	Source code	Developers	Defects
Number of Revisions (NR)	X		
Number of Modified Lines (NML)	X		
Is New (IN)	X		
Number of Distinct Commiters (NDC)	X	X	
Number of Defects in Previous Version (NDPV)	X		X

We may see that all of the process metrics rely on the information concerning the source code artefact changes, but only some of them combine it with the number of developers responsible for the aforementioned changes or the defects in earlier version. It is easy to imagine that new process metrics may include other sources of information than just source code artefacts. Changes in UML models or other artefacts, as well as a wide range of fluctuations in development teams (with respect to their size or structure), serve as good examples how to extend further the proposed taxonomy. It is also worth mentioning that some of the changes may be not directly related to the software process used.

This paper shows, in our opinion, that process metrics can be an effective addition to software defect prediction models usually built upon product metrics; however, the issue might be elaborated on and developed further. Our next aim is to evaluate the added predictive ability due to process metrics. Demonstration of a statistically significant association of a new process metric with defect risk is not enough. Hence, the next aim is to propose the evaluation framework in which it would be possible to assess the incremental value of new process metrics.

References

[1] B. Henderson-Sellers. *Object-oriented metrics: measures of complexity*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1996.

[2] T. Illes-Seifert, B. Paech. *Exploring the relationship of a file's history and its fault-proneness: An empirical method and its application to open source programs*. *Information and Software Technology*, 52(5):539–558, 2010.

[3] S. H. Kan. *Metrics and Models in Software Quality Engineering*. Addison-Wesley, Boston, MA, USA, 2002.

[4] L. Madeyski. *Test-Driven Development: An Empirical Evaluation of Agile Practice*. Springer, (Heidelberg, Dordrecht, London, New York), 2010.

[5] R. M. Bell, T. J. Ostrand, E. J. Weyuker. *Looking for bugs in all the right places*. *ISSTA '06: Proceedings of the 2006 international symposium on Software testing and analysis*, pp. 61–72, New York, NY, USA, 2006. ACM.

[6] T. J. Ostrand, E. J. Weyuker, R. M. Bell. *Where the bugs are*. *ISSTA '04: Proceedings of the 2004 ACM SIGSOFT International Symposium on Software Testing and Analysis*, pp. 86–96, New York, NY, USA, 2004. ACM.

[7] E. J. Weyuker, T. J. Ostrand, R. M. Bell. *Adapting a fault prediction model to allow widespread usage*. *PROMISE'06: Proceedings of the 4th International Workshop on Predictor Models in Software Engineering*, pp. 1–5, New York, NY, USA, 2006. ACM.

[8] E. J. Weyuker, T. J. Ostrand, R. M. Bell. *Using Developer Information as a Factor for Fault Prediction*. *PROMISE '07: Proceedings of the Third International Workshop on Predictor Models in Software Engineering*, p. 8, Washington, DC, USA, 2007. IEEE Computer Society.

[9] E. J. Weyuker, T. J. Ostrand, R. M. Bell. *Do too many cooks spoil the broth? Using the number of developers to enhance defect prediction models*. *Empirical Software Engineering*, 13(5):539–559, 2008.

[10] J. Ratzinger, M. Pinzger, H. Gall. *EQ-Mine: Predicting Short-Term Defects for Software Evolution*. M. Dwyer, A. Lopes, editors, *Fundamental Approaches to Software Engineering*, vol. 4422 of *Lecture Notes in Computer Science*, pp. 12–26. Springer Berlin / Heidelberg, 2007.

[11] T. L. Graves, A. F. Karr, J. S. Marron, H. Siy. *Predicting Fault Incidence Using Software Change History*. *IEEE Transactions on Software Engineering*, 26(7):653–661, 2000.

[12] A. Schröter, T. Zimmermann, R. Premraj, A. Zeller. *If your bug database could talk. In Proceedings of the 5th International Symposium on Empirical Software Engineering, Volume II: Short Papers and Posters*, pp. 18–20, 2006.

[13] N. Nagappan, T. Ball. *Using Software Dependencies and Churn Metrics to Predict Field Failures: An Empirical Case Study*. *ESEM '07: Proceedings of the First International Symposium on Empirical Software Engineering and Measurement*, pp. 364–373, Washington, DC, USA, 2007. IEEE Computer Society.

[14] N. Nagappan, A. Zeller, T. Zimmermann, K. Herzig, B. Murphy. *Change Bursts as Defect Predictors*. *Proceedings of the 2010 IEEE 21st International Symposium on Software Reliability Engineering*, ISSRE '10, pp. 309–318, Washington, DC, USA, 2010. IEEE Computer Society.

[15] E. Shihab, Z. M. Jiang, W. M. Ibrahim, B. Adams, A. E. Hassan. *Understanding the impact of code and process metrics on post-release defects: a case study on the Eclipse project*. *ESEM '10: Proceedings of the 2010 ACM-IEEE International Symposium on Empirical Software Engineering and Measurement*, pp. 1–10, New York, NY, USA, 2010. ACM.

[16] R. Moser, W. Pedrycz, G. Succi. *A comparative analysis of the efficiency of change metrics and static code attributes for defect prediction*. *ICSE '08: Proceedings of the 30th International Conference on Software Engineering*, pp. 181–190, New York, NY, USA, 2008. ACM.

[17] T. Zimmermann, N. Nagappan, H. Gall, E. Giger, B. Murphy. *Cross-project defect prediction: a large scale experiment on data vs. domain vs. process*. *ESEC/FSE '09: Proceedings of the 7th joint meeting of the European Software Engineering Conference and the ACM SIGSOFT Symposium on the Foundations of Software Engineering*, pp. 91–100, New York, NY, USA, 2009. ACM.

[18] S. Matsumoto, Y. Kamei, A. Monden, K. ichi Matsumoto, M. Nakamura. *An Analysis of Developer Metrics for Fault Prediction*. *PROMISE '10: Proceedings of the Sixth International Conference on Predictor Models in Software Engineering*, pp. 18:1–18:9. ACM, 2010.

[19] E. J. Weyuker, T. J. Ostrand, R. M. Bell. *Programmer–based Fault Prediction*. *PROMISE '10: Proceedings of the Sixth International Conference on Predictor Models in Software Engineering*, pp. 19:1–19:10. ACM, 2010.

[20] N. Nagappan, B. Murphy, V. Basili. *The influence of organizational structure on software quality: an empirical case study*. *ICSE '08: Proceedings of the 30th International Conference on Software Engineering*, pp. 521–530, New York, NY, USA, 2008. ACM.

[21] T. M. Khoshgoftaar, E. B. Allen, N. Goel, A. Nandi, J. McMullan. *Detection of software modules with high debug code churn in a very large legacy system*. *Proceedings of the The Seventh International Symposium on Software Reliability Engineering*, ISSRE '96, pp. 364–, Washington, DC, USA, 1996. IEEE Computer Society.

[22] L. Layman, G. Kudrjavets, N. Nagappan. *Iterative identification of fault-prone binaries using in-process metrics*. *ESEM '08: Proceedings of the Second ACM-IEEE International Symposium on Empirical Software Engineering and Measurement*, pp. 206–212, New York, NY, USA, 2008. ACM.

[23] N. Nagappan, T. Ball. *Use of relative code churn measures to predict system defect density*. *ICSE '05: Proceedings of the 27th International Conference on Software Engineering*, pp. 284–292, New York, NY, USA, 2005. ACM.

[24] R. Purushothaman, D. E. Perry. *Toward Understanding the Rhetoric of Small Source Code Changes*. *IEEE Transactions on Software Engineering*, 31:511–526, 2005.

[25] J. Śliwerski, T. Zimmermann, A. Zeller. *When do changes induce fixes? MSR '05: Proceedings of the 2005 International Workshop on Mining Software Repositories*, pp. 1–5, New York, NY, USA, 2005. ACM.

[26] A. E. Hassan. *Predicting faults using the complexity of code changes*. *ICSE '09: Proceedings of the 31st International Conference on Software Engineering*, pp. 78–88, Washington, DC, USA, 2009. IEEE Computer Society.

[27] E. Giger, M. Pinzger, H. C. Gall. *Comparing fine-grained source code changes and code churn for bug prediction*. *Proceedings of the 8th Working Conference on Mining Software Repositories*, MSR '11, pp. 83–92, New York, NY, USA, 2011. ACM.

[28] B. Fluri, H. C. Gall. *Classifying Change Types for Qualifying Change Couplings*. *Proceedings of the 14th IEEE International Conference on Program Comprehension*, ICPC '06, pp. 35–45, Washington, DC, USA, 2006. IEEE Computer Society.

[29] B. Fluri, M. Wuersch, M. PInzger, H. Gall. *Change Distilling: Tree Differencing for Fine-Grained Source Code Change Extraction*. *IEEE Trans. Softw. Eng.*, 33(11):725–743, November 2007.

[30] E. Giger, M. Pinzger, H. Gall. *Using the gini coefficient for bug prediction in eclipse*. *Proceedings of the 12th International Workshop on Principles of Software Evolution and the 7th annual ERCIM Workshop on Software Evolution*, IWPSE-EVOL '11, pp. 51–55, New York, NY, USA, 2011. ACM.

[31] T. J. Ostrand, E. J. Weyuker, R. M. Bell. *Predicting the Location and Number of Faults in Large Software Systems*. *IEEE Transactions on Software Engineering*, 31(4):340–355, 2005.

[32] T. M. Khoshgoftaar, E. B. Allen, R. Halstead, G. P. Trio, R. M. Flass. *Using Process History to Predict Software Quality*. *Computer*, 31(4):66–72, 1998.

[33] T. J. Ostrand, E. J. Weyuker. *The distribution of faults in a large industrial software system*. *ISSTA '02: Proceedings of the 2002 ACM SIGSOFT International Symposium on Software Testing and Analysis*, pp. 55–64, New York, NY, USA, 2002. ACM.

[34] D. Wahyudin, A. Schatten, D. Winkler, A. M. Tjoa, S. Biffl. *Defect Prediction using Combined Product and Project Metrics - A Case Study from the Open Source "Apache" MyFaces Project Family*. *SEAA '08: Proceedings of the 2008 34th Euromicro Conference Software Engineering and Advanced Applications*, pp. 207–215, Washington, DC, USA, 2008. IEEE Computer Society.

[35] E. Arisholm, L. C. Briand. *Predicting fault-prone components in a java legacy system*. *ISESE '06: Proceedings of the 2006 ACM/IEEE International Symposium on Empirical Software Engineering*, pp. 8–17, New York, NY, USA, 2006. ACM.

[36] S. Kim, T. Zimmermann, E. J. Whitehead Jr., A. Zeller. *Predicting Faults from Cached History*. *ICSE '07: Proceedings of the 29th International Conference on Software Engineering*, pp. 489–498, Washington, DC, USA, 2007. IEEE Computer Society.

[37] T. Gyimothy, R. Ferenc, I. Siket. *Empirical Validation of Object-Oriented Metrics on Open Source Software for Fault Prediction*. *IEEE Transactions on Software Engineering*, 31(10):897–910, 2005.

[38] K. Gao, T. M. Khoshgoftaar, H. Wang, N. Seliya. *Choosing software metrics for defect prediction: an investigation on feature selection techniques*. *Software: Practice and Experience*, 41(5):579–606, 2011.

[39] M. Jureczko. *Significance of Different Software Metrics in Defect Prediction*. *Software Engineering: An International Journal*, 1(1):86–95, 2011.

[40] R. Bell, T. Ostrand, E. Weyuker. *The limited impact of individual developer data on software defect prediction*. *Empirical Software Engineering*, pp. 1–28, 2011. 10.1007/s10664-011-9178-4.

[41] R. M. Bell, T. J. Ostrand, E. J. Weyuker. *Does measuring code change improve fault prediction? Proceedings of the 7th International Conference on Predictive Models in Software Engineering*, Promise '11, pp. 2:1–2:8, New York, NY, USA, 2011. ACM.

[42] Y. Shin, A. Meneely, L. Williams, J. Osborne. *Evaluating Complexity, Code Churn, and Developer Activity Metrics as Indicators of Software Vulnerabilities*. *Software Engineering, IEEE Transactions on*, 37(6):772 –787, nov.-dec. 2011.

[43] B. Sisman, A. C. Kak. *Incorporating version histories in Information Retrieval based bug localization*. *Mining Software Repositories (MSR), 2012 9th IEEE Working Conference on*, pp. 50 –59, june 2012.

[44] S. Krishnan, C. Strasburg, R. R. Lutz, K. Goševa-Popstojanova. *Are change metrics good predictors for an evolving software product line? Proceedings of the 7th International Conference on Predictive Models in Software Engineering*, Promise '11, pp. 7:1–7:10, New York, NY, USA, 2011. ACM.

Poszukiwanie twarzy i rozpoznawanie emocji w scenach zmiennych

Sylwester Wosiak, Katarzyna Buda, Antoni Wiliński

Zachodniopomorski Uniwersytet Technologiczny w Szczecinie, Wydział Informatyki

Abstract:

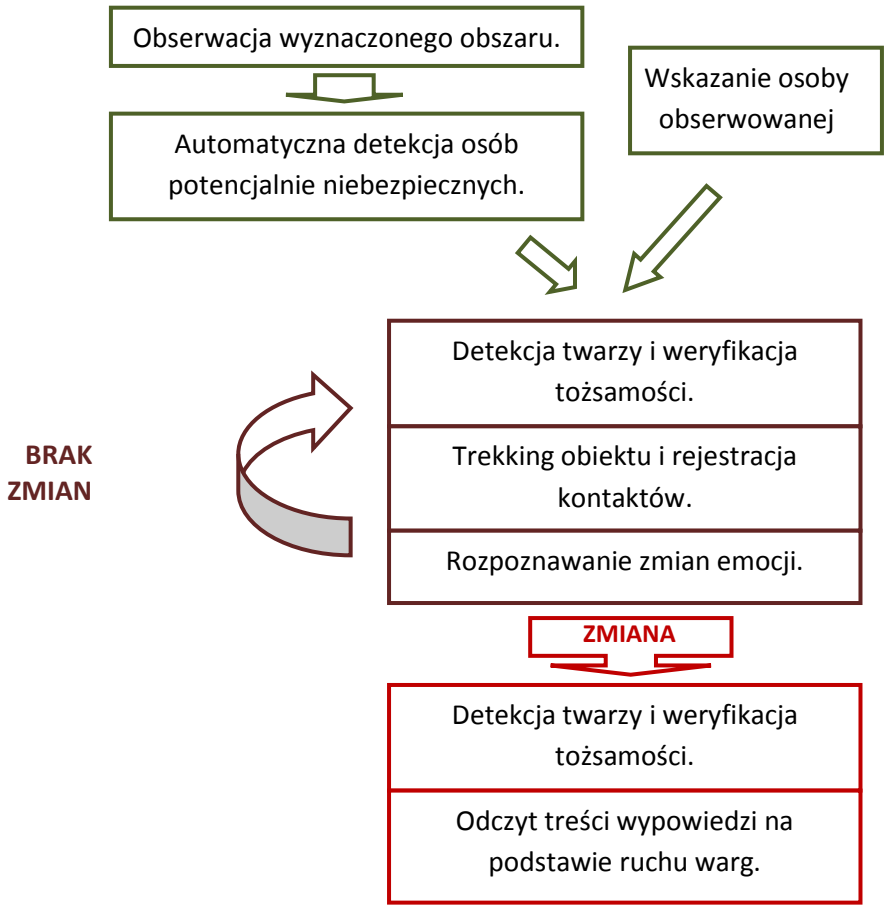
This article focuses on the problems encountered when using face observation and emotion recognition for the purposes of identification and also classification of specific emotions. The identification of emotions are particularly difficult, especially within varying scenes. In this paper we review the main methods of the identification of certain emotions. The authors present the results of their research analysis in tracking changes of the emotions in the selected software packages.

Keywords:

motion analysis, detection of emotions, recognition of emotional changes, identification methods, facial recognition, biometrics, people monitoring.

1. Wprowadzenie

Systemy nadzoru i kontroli dostępu wykorzystujące do identyfikacji tożsamości twarz, jako element obrazu poddawany rozpoznawaniu są, pod względem zachowania poufności prowadzonych działań, jednymi z najbezpieczniejszych i najwygodniejszych sposobów obserwacji osoby [1,2,4]. Bardzo istotnym czynnikiem przemawiającym za takim rozwiązaniem jest fakt, że pobranie próbek wizerunku twarzy dokonywane jest w sposób bezinwazyjny, oraz co ważniejsze może odbywać się przy braku interakcji z osobą identyfikowaną (bez jej zgody i świadomości). Kolejną niewątpliwie ważną cechą przemawiającą za użyciem twarzy do celów rozpoznawania jest to, że zawiera ona dużą ilość informacji zgromadzonych na stosunkowo małym obszarze. Umożliwia to nie tylko identyfikację osoby ale także jednocześnie ocenę emocji [7,8,11,12,13,17,19], a nawet automatyczny odczyt treści wypowiedzi na podstawie ruchu warg [17,18]. Często rozwiązanie oparte na elementach sprawdzonych i wielokrotnie już wykorzystywanych w innych aplikacjach daje przy podejściu kompleksowym zupełnie nowe, niewykorzystywane dotąd możliwości. Autorzy, uwzględniając liczne źródła literaturowe, proponują następujący uogólniony model systemu monitoringu z algorytmem uruchamiającym poszczególne funkcjonalności systemu (rys. 1).



Rysunek 1. Algorytm działania systemu

Zespolenie aplikacji w jeden system nadzoru i monitorowania osób na podstawie obserwacji twarzy ma na celu wprowadzenie nowych funkcjonalności, między innymi:

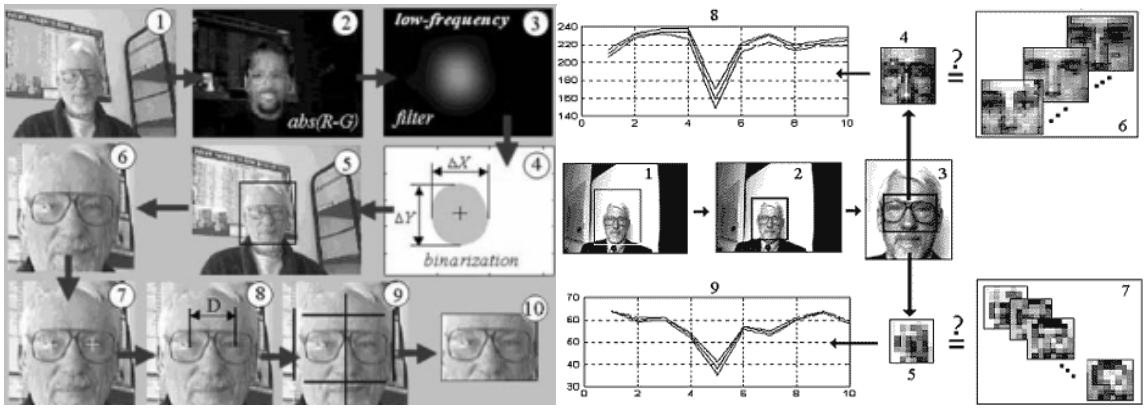
- detekcji twarzy i weryfikacji tożsamości;
- analizy trasy przemieszczania osoby z rejestracją kontaktów, częstotliwości i długości konwersacji;
- poznania nastroju i predyspozycji psychofizycznych, a w powiązaniu z okolicznościami i specyfiką miejsca obserwacji intencji monitorowanej osoby, w celu zapewnienia bezpieczeństwa chronionego obiektu i wczesnego podjęcia odpowiednich działań prewencyjnych;
- tworzenia sieci powiązań osób i wykrywania wymiany informacji wrażliwych oparte na detekcji zmian emocji.

1.1. Detekcja twarzy i weryfikacja tożsamości

Pierwszym w aspekcie kolejności i podstawowym zadaniem systemów rozpoznawania twarzy jest wykrycie i zlokalizowanie twarzy na obrazie [1,2]. Do osiągnięcia tego celu wykorzystują one różne podejścia i rodzaje informacji. Należą do nich aplikacje oparte na algorytmach analizy barw i kształtu, metody map i wykrywania krawędzi, a także elastyczne wzorce służące do precyzyjnej lokalizacji cech.

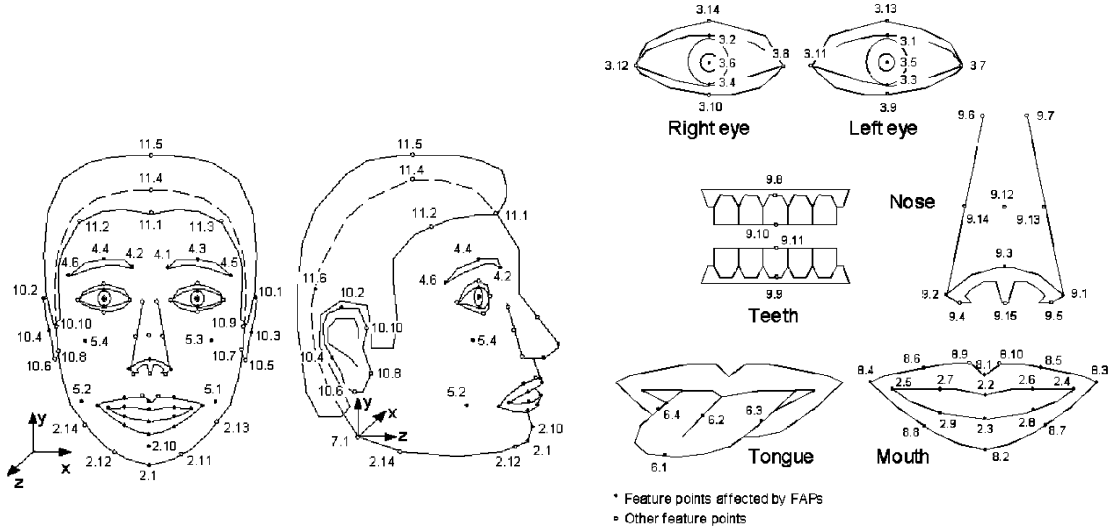
Rysunek 2. Podział detektorów twarzy zaproponowany przez Yanga

Ten podstawowy podział najważniejszych metod detekcji twarzy zaproponował m.in. Yang [1]. Klasyfikację tę przedstawiono na rys. 2. Na kolejnym rys. 3 przedstawiono etapy dalsze, następujące po detekcji twarzy, związane z jej weryfikacją np. w celu identyfikacji, stwierdzenia tożsamości lub dodania do bazy.



Rysunek 3. Etapy lokalizacji i weryfikacji twarzy [2]

Twarz po wykryciu zostaje znormalizowana i porównana ze wzorcem w bazie, w celu identyfikacji. Korzysta się z wielu metod np. takich jak KLT (Karhunen-Loeve Transform), LDA (Linear Discriminant Analysis), PCA (Principal Component Analysis) i ich kompilacji [2] oraz metod opartych na pomiarze punktów charakterystycznych twarzy [3].



Rysunek 4. Punkty charakterystyczne twarzy w standardzie MPEG-4 FAP [3]

Rys. 4 przedstawia ogólny pogląd na złożoność i precyzję tej ostatniej metody. Czynnikiem charakterystycznym wszystkich omówionych krótko podejść lub metod detekcji i weryfikacji twarzy było założenie niezmienności sceny, na tle której twarz była ekstrahowana z obrazu. Typowym przykładem praktycznego stosowania takich metod jest użycie zdjęcia dla identyfikacji osoby lub obrazu z kamery, gdy twarz jest w zasadzie jedynym obiektem obrazu np. przy identyfikacji gościa obserwowanego przy wejściu do domu, w wąskim przejściu itp. Z odmienną praktycznie ważną sytuacją ma się do czynienia wówczas, gdy scena, na tle której twarz jest obserwowana jest silnie zmienna w czasie.

1.2. Lokalizacja twarzy i weryfikacja tożsamości w zmiennych scenach

Detekcja twarzy, nadzór i identyfikacja osoby nastręcza wiele problemów w momencie, gdy zaczyna się ona poruszać, zwłaszcza w otoczeniu innych ludzi. Następuje wtedy ciągła zmiana tła, zmienia się również skala i kąt obserwacji, a twarz lub nawet cała postać może zostać chwilami przesłonięta przez meble, elementy infrastruktury stałej lub inne osoby (następuje permanentna zmiana sceny). Z tym problemem zdecydowanie lepiej radzą sobie aplikacje identyfikujące, oprócz twarzy, również ubiór rozpoznawanych osób. Opisana przez Josefa Sivica [4] metoda lokalizacji osób w powtarzających się ujęciach tej samej sceny wykorzystuje dwie podstawowe metody.

Pierwsza (rys. 5) - rozpoznawanie osób opiera się na detekcji twarzy oraz informacji o kolorze ubioru.

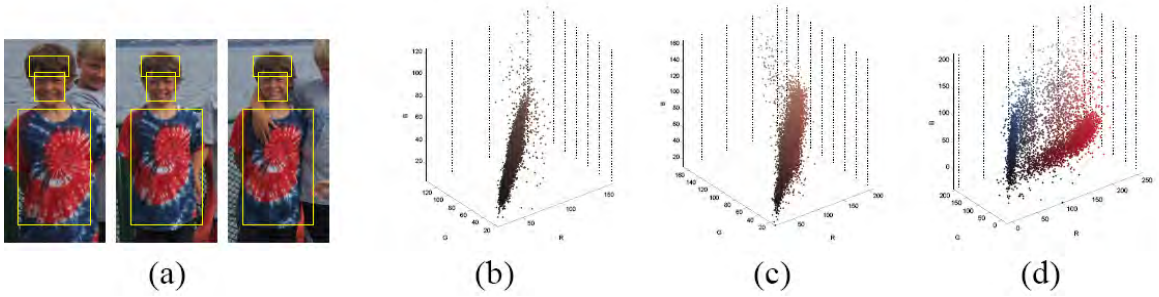
Druga (rys. 6) – model bazowy struktury ubioru oraz informacja o włosach są wykorzystywane do identyfikacji w wypadku braku widoczności twarzy rozpoznawanej osoby.

Na rys. 5 przedstawiono serię obrazów, na których wykorzystuje się jednocześnie informację o ubiorze i o twarzy. Na kolejnych fotografiach pokazano: (a) – Przestrzenna maska wykorzystywana do opisu odzieży każdego człowieka. Jasne pola wskazują miejsca o największej wadze, (b) – wycięta osoba z nałożonym obszarem detekcji twarzy, (c) – obraz uzyskany po nałożeniu maski, (d) i dalej – zidentyfikowane osoby [4].

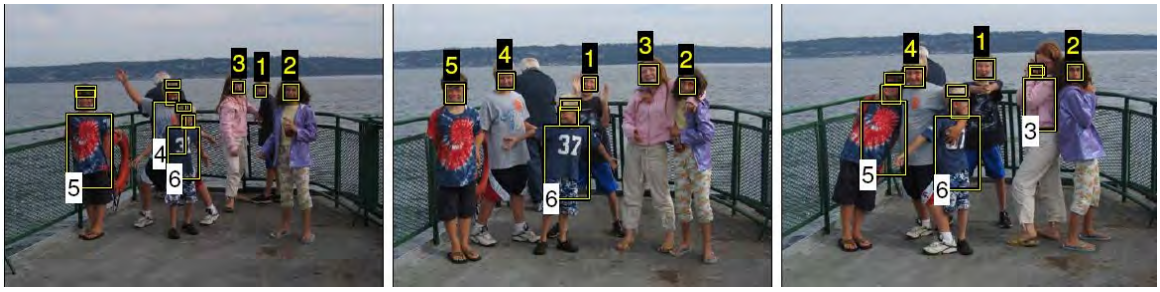
(a) (b) (c) (d)

Rysunek 5. Przykład zastosowania pierwszej z metod Sivica uwzględniającej ubiór i twarz [4]

Na kolejnym zdjeciu przedstawiono drugą z metod Sivica uwzględniajacą detekcję wybranych elementów osoby (twarz, włosy, ubranie, skóra) na podstawie kolorów podstawowych. Na rysunku tym kolejno prezentuje się: (a) – prostokąty wyznaczające obszary włosów, skóry i tułowia dla 3 różnych obrazów, (b, c, d) - próbki rozkładu Gaussa kolorów RGB dla włosów, skóry twarzy i tułowia. W tym przypadku widać dominację koloru niebieskiego i czerwonego. Będzie to niestety widoczne podobnie jak elementy rys. 6 w wydaniu kolorowym tej pracy.



Rysunek 6. Przykład zastosowania metody opisanej przez Sivica uwzględniającej kolory elementów postaci [4]



Rysunek 7. Zdjęcia ilustrujące jednoczesne użycie dwu metod [4]

Zdjęcia na rys. 7 przedstawia porównanie skuteczności rozpoznawania jednocześnie obydwiema metodami. Osoby rozpoznane na postawie detekcji twarzy i informacji o ubiorze oznakowane są żółtymi cyframi na czarnym tle (w wydaniu czarno-białym będzie to jasna cyfra na ciemnym tle), natomiast osoby, do których identyfikacji wykorzystano model bazowy struktury ubioru oraz włosów oznaczono ciemną cyfrą na białym tle [4]. Metody te niejako podsumowują rozdział o detekcji twarzy i identyfikacji osób także w zmiennych scenach. Nie rozwiązują one innego ważnego wyzwania stawianego biometrii – rozpoznawania emocji i zmian emocji u rozpoznanych w scenach osób.

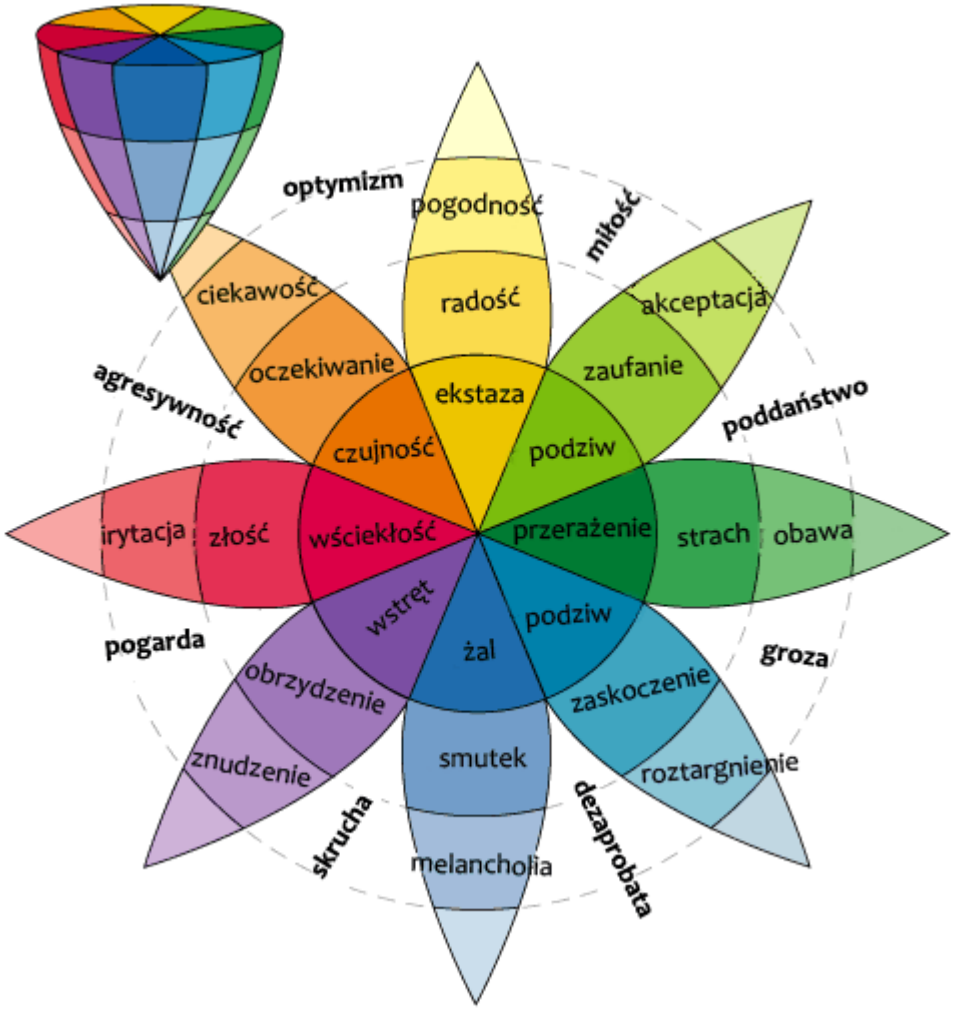
1.3. Rozpoznawanie emocji u obserwowanych osób

Od zarania dziejów ludzie spotykając na swej drodze obcego w obawie o swoje bezpieczeństwo uważnie obserwowali zachowanie i mimikę twarzy próbując odgadnąć jego intencję. Poznanie emocji przeciwnika czy rozmówcy jest równie ważne podczas negocjacji, by osiągać jak najkorzystniejszy wynik i nie zakończyć ich przedwcześnie. Ludzie nauczyli się ukrywać swoje emocje, kontrolując zachowania pozawerbalne, jednak istnieją liczne reakcje organizmu, zmieniające się w zależności od ogólnego pobudzenia współczulnego układu nerwowego, których kontrola jest praktycznie niemożliwa. Na tej podstawie opiera się np. działanie wariografu, wymagającego jednak bezpośredniego kontaktu z badanym.

Najkorzystniej byłoby poddać jednoczesnemu testowi wiele osób zgromadzonych w jednym miejscu w celu rozpoznania łączących ich emocji czy wykrycia jednostek potencjalnie niebezpiecznych, np. terrorystów na terminalu lotniska. Optymalnym rozwiązaniem byłoby połączenie w jednym systemie procesów weryfikacji tożsamości i rozpoznawania emocji, wykorzystując do tego celu np. kamery przemysłowe, które są dzisiaj tak powszechne, że ich obecność nie wzbudza niczyich podejrzeń.

Według teorii emocji psychologa Roberta Plutchika opracowanej w latach 1960-80 istnieje osiem podstawowych emocji, ewolucyjnie naturalnie rozwiniętych [5]. Są to wrodzone emocje odnoszące się do zachowań adaptacyjnych, mające na celu pomoc w przetrwaniu.

Pozostałe emocje są kombinacją emocji pierwotnych odpowiedzialnych za konkretne przeżycia. Relacje między emocjami opisuje trójwymiarowy diagram przedstawiony na rys. 7. Zgodnie z teorią Plutchika, emocje sąsiadujące na diagramie, gdy są odczuwane jednocześnie, mogą być połączone w nowe odczuwane jednocześnie zwane diadami. Emocje przeciwstawne nie mogą być ze sobą łączone, ponieważ zachodzi między nimi konflikt. Emocje są skutkiem przeżywanych zdarzeń, ale też same wywołują konkretne działania adekwatne do wydarzeń.



Rysunek 7. Trójwymiarowy diagram emocji Plutchika wg Ivana Akira [5]

Badając komunikację społeczną Albert Mehrabian stwierdził, że aż 55 % informacji przekazywanych jest za pomocą komunikatów niewerbalnych, 38 % stanowi artykulacja, a zaledwie 7 % przekazu to słowa [6]. Nie dziwi, więc fakt, że tak wielu naukowców zajęło się problematyką automatycznego rozpoznawania emocji i choć poczyniono już wielkie postępy, wyniki nadal nie są w pełni zadawalające. Zdecydowana większość algorytmów wykorzystuje punkty charakterystyczne twarzy do wyznaczenia newralgicznych obszarów: ust, oczu brwi, okolic nosa. Podstawowym problemem koniecznym do rozwiązania dla właściwej pracy tych aplikacji jest więc prawidłowe wyznaczenia wszystkich punktów. Zadanie to staje się bardzo trudne, gdy osoba poddana rozpoznawaniu znajduje się w ruchu, uniemożliwiając nieprzerwaną obserwację jej twarzy. W tab. 1 przedstawiono podsumowanie

wielu badań dotyczących ekstrakcji emocji obserwowanych osób na podstawie obrazów twarzy i niektórych cech behawioralnych. Problematyka ta ma ciągle rosnące znaczenie, jest niezwykle aktualna w kontekście bieżącej sytuacji politycznej. Stanowi też wielkie wyzwanie dla uczonych zajmujących się różnymi dziedzinami nauki – informatykami, matematykami, psychologami, socjologami, lekarzami i biologami. Tab. 1. wskazuje, poprzez różnorodność metod, na ogromną kreatywność i determinację takich zespołów.

Tabela 1. Metody automatycznego rozpoznawania ekspresji.

Źródło	**Przetwarzanie wstępne**	**Wyodrębnienie cech**	**Klasyfikacja**	**Wyniki badań**
Bourel [14]	Śledzenie punktów orientacyjnych twarzy przy użyciu specjalizowanej aplikacji	Skalarna kwantyzacja dynamiki twarzy, reprezentowana przez tymczasową zmianę punktów charakterystycznych.	Klasyfikator: ranking wag k najbliższych sąsiadów *Klasy:* prototypy 6-ciu emocji	Sekwencja 300 obrazów. Stosunkowo niskie błędy dla zasłoniętych dolnych części twarzy, oraz zaszumionych zdjęć.
Pantic, Rothkrantz [15]	Lokalizacja / śledzenie twarzy i jej części. Korzysta z wielu detektorów	Wyodrębnienie statycznych geometrycznych wartości dla krawędzi obszarów.	Klasyfikator: dwu etapowy, opierający się na dekodowaniu zmian AU i ich klasyfikacji wg prototypów ekspresji. *Klasy:* AU i prototypy 6-ciu emocji. Przetwarzanie końcowe: dwuetapowa architektura klasyfikacji.	256 obrazów, 2 widoki, 8 modeli. Skuteczności 91%
Essa, Pentland [16]	Lokalizacja twarzy przez dopasowanie obrazu 2D do siatki 3D	1. Aktywacja mięśni na podst. modelu deformacji, 2. Przepływ energii 2D	Klasyfikator: 1. maksymalna korelacja między aktywnością mięśni, 2. minimalna odległość energii od wzorca. *Klasy:* prototypy 5-ciu emocji	Sekwencje 52 obrazów na wprost, 8 modeli. Skuteczności 98%.
Tian i inni [17]	Segmentacja i śledzenie cech bazując na mieszanym modelu Gaussa, algorytmie Lucasa-Kanade, i detektorze krawędzi Canna	Reprezentacja zmian dynamiki twarzy przedstawiona na danych statycznych za pomocą wektorów, normalizacja w płaszczyźnie ruchów głowy.	Klasyfikator: po jednej sieci wielowarstwowej na górną i dolą część twarzy. Analizowane AU. *Klas*y: 6 prototypów dla górnej i 10 dla dolnej części twarzy, neutralny wyraz.	Sekwencja 122 obrazów, 21 modeli. Skuteczność 93,3%
Donato i inni [18]	Geometryczna normalizacja, rotacja i skalowanie. Segmentacja: górna i dolna. Normalizacja jasności i histogramu. Estymacja ruchu	Całościowa analiza przestrzenna: PCA, LFA, ICA, FLD. Lokalna analiza przestrzenna: PCA, Gabor.	Klasyfikator: dopasowanie do najbliższego sąsiada, bądź wzorca. *Klasy:* 6 prototypów dla górnej i 6 dla dolnej części twarzy.	111 sekwencji obrazów, 20 modeli. Skuteczność 95, 5% (dla filtrów Gabora i AU)
Lien i nni [19]	Segmentacja lub estymacja ruchu. Geometryczna normalizacja, rotacja i skalowanie głowy. Segmentacja: górna i dolna połowa.	Alternatywna reprezentacja: Przemieszczenie 6 punktów powyżej granicy brwi, PCA, wariancja pikseli.	Klasyfikator: kwantowany wektor poprzedzony ukrytym modelem Markova, *Klasy:* 3 prototypu dla górnej części twarzy i wyraz neutralny.	Sekwencje 260 obrazów, 60 modeli. Skuteczności 93% (PCA), 85% AU.

Na tle tych metod rozpoznawania emocji Autorzy przedstawiają własne próby przeprowadzone za pomocą oprogramowania nabytego u jego twórcy Wiktora Minkina [7,8].

2. Vibraimage jako metoda rozpoznawania emocji

Jedną z wielu ciekawych metod rozpoznawania emocji niewymagających ciągłej obserwacji twarzy jest *vibraimage*. Autorzy nie tłumaczą tej nazwy na język polski, na razie nie jest ona stosowana. Idea metody opiera się na wykrywaniu ludzkich emocji przez kontrolę zaburzeń pracy układu przedsionkowego (ulokowanego w głowie w okolicy ucha), odpowiedzialnego między innymi za zachowanie równowagi oraz pozycji pionowej głowy. Przeżywane emocje mają wpływ na pracę całego układu nerwowego człowieka i indukują zakłócenia również w pracy układu przedsionkowego, co może zostać odczytane przy odpowiedniej obserwacji drgań powierzchni twarzy oraz ruchów głowy w trzech wymiarach, zarejestrowanych na kilku następujących po sobie ramkach obrazu [7]. Przez analizę amplitudy i częstotliwość drgań poszczególnych pikseli otrzymujemy obraz wynikowy w postaci vibraimage. To trójwymiarowy obraz twarzy, w którym trzeci wymiar – badanej zmiennej związanej z emocją reprezentowany jest za pomocą koloru. Wartości macierzy początkowych dla parametrów reprezentujących amplitudę i częstotliwość obliczono według wzoru [8]:

$$A_{x,y} = \frac{1}{N}\sum_{i=1}^{N}\left|U_{x,y,i} - U_{x,y,(i+1)}\right|, \tag{1}$$

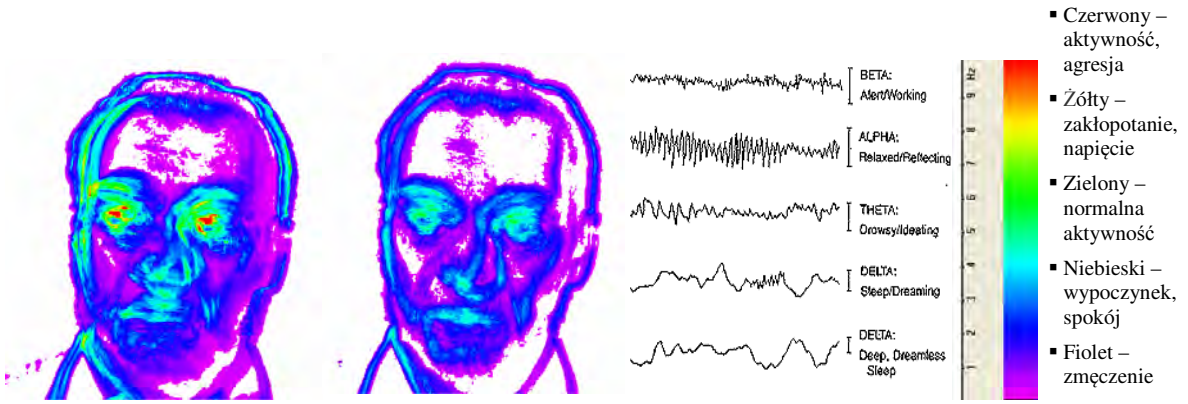
a komponent częstotliwości każdego punktu vibraimage jest obliczana według wzoru [8]:

$$F_{x,y} = \frac{F_{in}}{N}\sum_{i=1}^{N}\begin{cases}\left|U_{x,y,i} - U_{x,y,(i+1)}\right| > 0 : 1\\ \text{otherwise: } 0\end{cases}, \tag{2}$$

gdzie:

x, y to współrzędne punktu,
$U_{x,y,i}$ to amplituda sygnału w punktach x, y w i-tej ramce,
$U_{x,y(i+1)}$ jest amplituda sygnału w punktach x, y w (i+1) ramce,
N jest liczbą ramek uśredniania amplitudy składowej vibraimage,
F_{in} jest częstotliwością przetwarzanego obrazu.

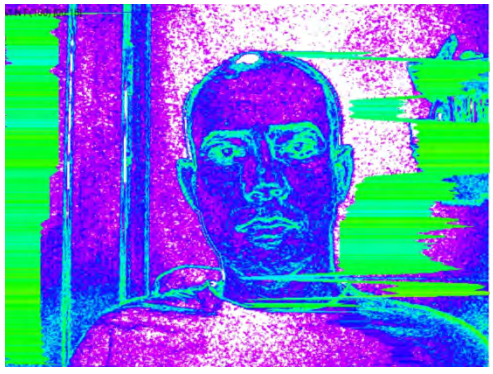
Zakres fluktuacji badanego obiektu został skorelowany na podstawie parametrów drgań z różnymi stanami emocjonalnymi takimi jak gniew, napięcie, agresja itp., a także nastrojem oraz stanami podświadomych napięć. W celu wizualizacji dobrano zakres barw vibraimage w sposób intuicyjny, odpowiadający zdekodowanym stanom emocjonalnym – rys. 8.



Rysunek 8. Amplitudowy i częstotliwościowy vibraimage ludzkiej głowy [7]

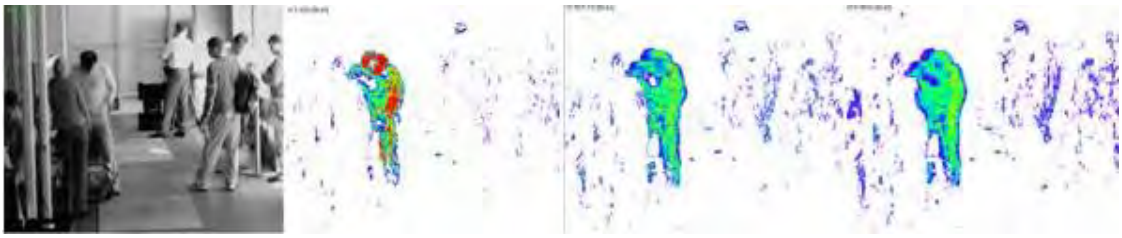
Na rys. 8 przedstawiono zakresy fal mózgowych i przypisane im kolory. Na rysunku czarno-białym skala z prawej strony ma charakter poglądowy. Poszczególne kolory odpowiadają odpowiednim kolorowym obszarom na obrazie twarzy [7,8].

Rysunek 9. Aura – zewnętrzny vibraimage ruchów głowy [7]



Rysunek 10. Zewnętrzny amplitudowy vibraimage wokół wewnętrznego częstotliwościowego vibraimage [7]

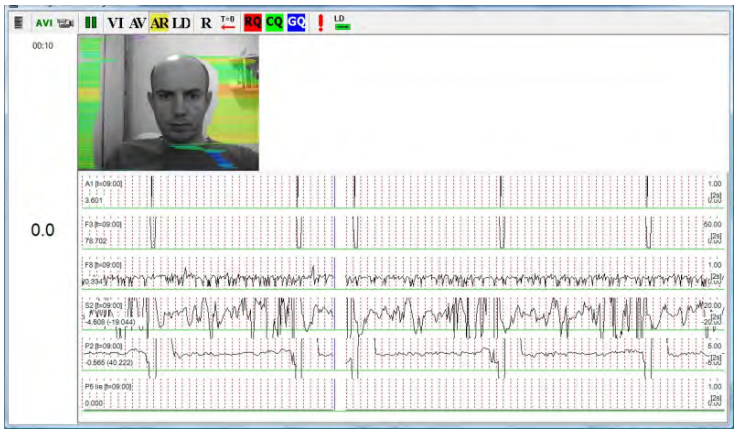
Na rys. 9 i 10 przedstawiono rezultaty obserwacji obrazu vibraimage wykonane przez jednego z autorów pracy. Na rys. 9 przedstawiono tzw. amplitudowy (zewnętrzny) vibraimage. W przypadku zdjęć czarno-białych są to poziome kreski z prawej strony obrazu (na kolorowym zdjęciu są to zielone pasma). Na rys. 10 oba obrazy vibraimage twarzy tego samego autora. Kolorowe pola związane bezpośrednio z twarzą stanowią tzw. vibraimage wewnętrzny (częstotliwościowy) a pasma z boku obrazu – jak poprzednio na rys. 9 vibraimage amplitudowy.



Rysunek 11. Ocena poziomu aktywności emocjonalnej vibraimage grupy osób [7]

Autorzy nie odnoszą się w niniejszej pracy szczegółowo do możliwości analizy obrazów emocji zwanych vibraimage. Jest to jedynie raport z podejmowanych prób, jednych z wielu

prowadzonych na całym świecie. O potencjalnych zastosowaniach można sądzić np. na podstawie zdjęcia (rys. 11) cytowanego przez V. Minkina [7]. Na zdjęciu widać grupę osób, może to być np. obraz z lotniska, wśród których część posiada zmienny vibraimage. Jest zupełnie oczywiste, że na zdjęciu występują wspomniane wcześniej sceny zmienne. Logiczne jest także, że w tym przypadku zmienność scen nie rzutuje na rozpoznawanie emocji dla już zainicjowanych obrazów vibraimage poszczególnych osób tak negatywnie jak w przypadku poszukiwania twarzy podobnych na obrazach w celu identyfikacji osób.



Rysunek 12. Detektor kłamstwa - analizujący mikrodrgania twarzy i ruchy głowy [7].

Na kolejnym rys. 12 widać inne zastosowanie metody vibraimage. Rysunek przedstawia interfejs oprogramowania [7], którego jedną z fukcjonalności jest detekcja takich emocji, które dadzą się wykorzystać do detekcji kłamstwa. Na iterfejsie w obrebie zdjecia widoczny jest zewnetrzny amplitudowy vibraimage a ponizej rejestrowana zmienność kilku obserwowanych parametrów biometrycznych. Regularne eksperymenty i rejestracja takich parametrów w funkcji zmiennych emocji pozwalaja na znalezienie korelacji pomiedzy obrazem vibraimage a rozmaitymi odmianami emocji obserwowanego człowieka.

3. Podsumowanie

Przez ostatnie lata pojawiło się wiele komercyjnych systemów rozpoznawania twarzy i weryfikacji tożsamości. Jednak pomimo gwałtownego rozwoju aplikacji bazujących na obserwacji twarzy ludzkiej zauważa się wyraźną dysproporcję i istotną wyraźną lukę w dziedzinie badań nad metodami rozpoznawania emocji. Rozwijanie metod automatycznego rozpoznawania emocji i zmian aktywności ruchowej obserwowanych osób jest nieuniknione, gdy chcemy poważnie myśleć o ich zastosowaniu w systemach biometrycznych przeznaczonych do powszechnego użytku. Bez najmniejszych wątpliwości przynajmniej niektóre z metod opisanych w tej pracy będą dalej rozwijane. Przede wszystkim jednak powinniśmy zdać sobie sprawę, że na obecnym etapie rozwoju stwarzają one liczne problemy, a ich zastosowanie wiąże się z wieloma ograniczeniami. Prawdopodobnie przyszłość należeć będzie do rozwiązań bardziej zaawansowanych wykorzystujących wiedzę zdobytą w trakcie uczenia. Tego typy algorytmy wymagają jednak zdecydowanie większej mocy obliczeniowej, a problemy praktyczne związane z ich zastosowaniem wymagają odrębnych badań.

Bibliografia:

[1] Yang M.-H., Kriegman D.J., Ahuja N. *Detecting Faces in Images: A Survey.* IEEE Transactions On Pattern Analysis And Machine Intelligence, vol. 24, no. 1, pp. 34- 58, 2002.

[2] Kukharev G., Kuźmiński A. *Techniki Biometryczne Część 1 – Metody Rozpoznawania Twarzy.* Politechnika Szczecińska Wydział Informatyki, 2003.

[3] Norma ISO/IEC 14496-2, *Generic coding of audio-visual objects, part 2: Visual* (MPEG-4 Visual), grudzień 1998.

[4] Sivic J., Zitnick C. L, and Szeliski R., *Finding people in repeated shots of the same scene*, University of Oxford, 2006.

[5] Plutchik R. *What is an emotion?* The Journal of Psychology, 61, University of California 1965.

[6] Pease A. *Mowa ciała. Jak odczytywać myśli innych ludzi z ich gestów.* Jedność, 2005.

[7] *Sistema analiza psikhofiziologicheskogo i emotsional'nogo sostoyaniya cheloveka*, Sankt – Peterburg 2010, http://www.elsys.ru/downloads/VI7ManualRusLite7_3.pdf

[8] Minkin V. *Vibroizobrazhenie*, Sankt – Peterburg, SPb.: Renome, 2007.

[9] N. A. Bernshtein, *Motion Physiology and Activity [in Russian]*, Nauka, Moscow 1990, pp. 373392.

[10] V. A. Minkin and N. N. Nikolaenko, *Application of Vibraimage Technology and System for Analysis of Motor Activity and Study of Functional State of the Human Body*, Biomedical Engineering, Vol. 42, No. 4, 2008.

[11] Kamenskaya E., Kukharev G., *Recognition of Psychological Characteristics from Face, Metody Informatyki Stosowanej, nr 1/2008, s 59.*

[12] Chiang P.-Y., Liao W.-H., Li T.-Y., *Automatic caricature generation by analyzing facial features,* In: Asian Conference on Computer Vision, Korea, January 27-30 2004.

[13] Chibelushi C. C., Bourel F., *Facial Expression Recognition: A Brief Tutorial Overview*, In: CVonline: On-Line Compedium of Computer Vision 2003.

[14] Chibelushi C. C., Bourel F., Low A.A. , *Robust Facial Expression Recognition Using a State-Based Model of Spatially-Localised Facial Dynamics*, Proc. Fifth IEEE Int. Conf. Automatic Face and Gesture Recognition, pp. 106-111, 2002

[15] Pantic M., Rothkrantz L.J.M., *An Expert System for Multiple Emotional Classification of Facial Expressions,* Proc. 11th IEEE Int. Conf. on Tools with Artificial Intelligence, pp. 113-120, 1999.

[16] Essa I. A., Pentland A.P., *Coding, Analysis, Interpretation, and Recognition of Facial Expressions,* IEEE Trans. Pattern Analysis and Machine Intelligence, Vol. 19, No. 7, pp. 757-763, 1997.

[17] Tian Y-L, Kanade T., Cohn J.F., *Recognizing Action Units for Facial Expression Analysis*, IEEE Trans. Pattern Analysis and Machine Intelligence, Vol. 23, No. 2, pp. 97-115, 2001.

[18] Donato G., Bartlett M.S., Hager J.C., Ekman P. , Sejnowski T.J., *Classifying Facial Actions,* IEEE Trans. Pattern Analysis and Machine Intelligence, Vol. 21, No. 10, pp. 974-989, 1999.

[19] Lien J.J., Kanade T., Cohn J.F. Li, C-C., *Automated Facial Expression Recognition Based on FACS Action Units*, Proc. Third IEEE Int. Conf. Automatic Face and Gesture Recognition, pp. 390-395, 1998.

Wykorzystanie formalizacji języka BPMN w badaniu ewolucji systemów SOA

Andrzej Ratkowski

Politechnika Warszawska, Instytut Automatyki i Informatyki Stosowanej

Abstract:

The paper presents formalization of BPMN notation by means of Language of Temporal Ordering of Observational Behaviour LOTOS. The paper consists mapping of elements of BPMN notation into LOTOS behavior expressions. The LOTOS semantic is used subsequently to analyze transformations of BPMN processes during evolution of Service-Oriented Architecture system that is represented in BPMN notation.

Keywords:

SOA, BPMN, LOTOS, SOA Evolution

1. Wprowadzenie

W artykule zaproponowano definicję semantyki notacji BPMN przy użyciu języka modelowania systemów współbieżnych LOTOS [2]. W tym celu przedstawiono odwzorowanie elementów notacji BPMN do języka LOTOS. Przedstawiono także sposób wykorzystania formalnych narzędzi LOTOS do badania przekształceń, którym podlegają procesy BPMN w trakcie ewolucji systemów o architekturze usługowej.

Język BPMN [3] stopniowo staje się standardem *de facto* w dziedzinie modelowania procesów biznesowych. Jest wykorzystywany przede wszystkim w powyższym celu, ale umożliwia rownież wytwarzanie procesów wykonywalnych. Procesy wykonywalne sterujące współpracą systemów informatycznych udostępniają swoje funkcje za pomocą usług w rozumieniu architektury usługowej (SOA). W konsekwencji, procesy BPMN podlegają rozwojowi w drodze ewolucji, która dotyczy całych systemów informatycznych opartych na architekturze usługowej [8].

1.1. Ewolucja systemów SOA

Systemy SOA, w najbardziej ogólnym ujęciu [1], opierają się na atomowych usługach oraz bardziej złożonych agregatach, które poprzez orkiestrację lub choreografię łączą usługi atomowe w usługi złożone. Przy czym zarówno usługi atomowe jak i usługi złożone mogą podlegać przekształceniom i rozwojowi dostosowując się do zmieniających się technologii, czy nowych wymagań biznesowych. Systemy oparte na architekturze usługowej, ze swojej natury, podlegają rozwojowi ewolucyjnemu. W [8] uzasadniono, że brakuje właściwej metodyki śledzenia i nadzoru nad takim rozwojem. W związku z powyższym zaproponowano w [6] i [7] własną koncepcję metodyki zarządzania ewolucją systemów, opartych o paradygmaty SOA.

Proponowana metodyka zakłada podział systemu SOA na trzy warstwy:

1. warstwę procesów biznesowych, które modelują rzeczywiste procesy realizowane przez organizację,

2. warstwę procesów orkiestracji usług, wspomagających procesy biznesowe warstwy 1,
3. warstwę usług atomowych z których są zbudowane procesy warstwy 2.

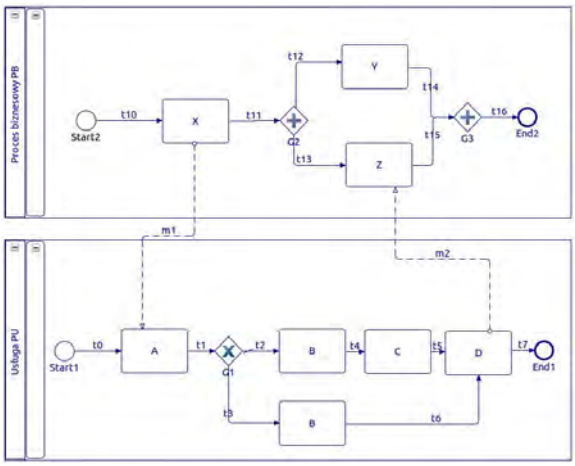
Omawiana powyżej metodyka wspiera śledzenie spójności zmian pomiędzy poszczególnymi warstwami. Podczas takiej ewolucji pomiedzy wartwami, można wyróżnić następujące specyficzne zagadnenia / problemy:

- czy zmiana w warstwie procesów biznesowych powinna modyfikować procesy orkiestracji usług,
- czy różne realizacje procesów orkiestracji są równoważne a jeżeli nie to na czym polegają różnice,
- czy zmiana w warstwie usług (np. zmiana technologiczna, usunięcie, dodanie albo modyfikacja usługi) ma wpływ na wyższe warstwy.

Niniejszy artykuł odpowiada na powyższe problemy / zagadnienia / pytania.

1.2. Język BPMN w architekturze usługowej

Na rys. 1 zaprezentowano diagram BPMN, który przedstawia system o architekturze usługowej. Basen *Proces Biznesowy PB* opisuje przykładowy proces biznesowy, którego czynności *(X,Y,Z)* są wykonywane przez pracowników organizacji. W ramach niektórych z tych czynności następuje wywołanie usługi *PU* systemu informatycznego, której realizacja jest opisana także w postaci diagramu BPMN, Czynności *A,B,C,D* są użyciami usług niższej warstwy. Jak z powyższego wynika usługa *PU* stanowi orkiestrację usług *A,B,C,D.*



Rysunek 1. Diagram BPMN system usługowego

1.3. Język LOTOS

Język LOTOS (*Language Of Temporal Ordering Specification*) [2] jest formalnie zdefiniowanym zapisem zachowania i komunikacji procesów współbieżnych. Używany jest do definiowania protokołów komunikacyjnych oraz do formalnej weryfikacji własności modelowanych systemów komputerowych (np. równoważności zachowań, zawierania się śladów wykonania). Język LOTOS, w wersji podstawowej (*basic LOTOS*), składa się z następujących elementów:

μ - *akcja*, reprezentuje aktywność jaką wykazuje modelowany system (wystąpienie zdarzenia, wysłanie komunikatu, itp.), akcja jest określona identyfikatorem. Występują dwa rodzaje akcji:

g – akcja obserwowalna, widoczna na zewnątrz systemu,

i - akcja nieobserwowalna, niewidoczna na zewnątrz.

B – wyrażenie zachowania czyli zbiór najważniejszych wyrażeń zachowania, do których należą:

***stop* -** najprostsze wyrażenie zachowania oznacza brak aktywności systemu,

μ;B – wyrażenie z operatorem poprzedzania (;) oznacza, że akcja μ poprzedza wyrażenie zachowania *B*, po wystąpieniu akcji μ może wystąpić zachowanie określone wyrażeniem *B,*

B1[]B2 – wyrażenie z operatorem wyboru niedeterministycznego ([]) oznacza że może wystąpić zachowanie *B1* albo *B2*,

B1|[μ_1,.., μ_n]\B2 – wykonanie równoległe z synchronizacją na akcjach, μ_1,.., μ_n oznacza, że wyrażenia *B1* i *B2* wykonują się współbieżnie i niezależnie aż do momentu wystąpienia w każdym z nich akcji ze zbioru $\{\mu_1,..,\mu_n\}$ (akcji synchronizacyjnych). W momencie wystąpienia akcji synchronizacyjnej, w wyrażeniu *B1* lub *B2*, wyrażenie to zatrzymuje wykonanie i oczekuje do momentu aż drugie wyrażenie dojdzie do miejsca w którym ma wystąpić ta sama akcja synchronizacyjna,

B1|||B2 – wykonanie równoległe całkowicie niezależne, wyrażenia *B1* i *B2* wykonują się niezależnie od sibie aż do momentu ich zakończenia,

exit – zakończenie sukcesem wykonywania wyrażenia zachowania,

B1>>B2 – kompozycja szeregowa, zakończenie sukcesem *B1* umożliwia rozpoczęcie B2,

***B1[>B2* -** blokowanie, wykonanie pierwszej akcji w wyrażeniu *B2* powoduje, że nie wykona się wyrażenie *B1*

hide μ_1,.., μ_n in B – ukrywanie akcji μ_1,..,μ_n w wyrażeniu *B*, takie wyrażenie oznacza, że jeżeli w trakcie wykonania wyrażenia zachowania *B,* wystąpi akcja należąca do zbioru $\{\mu_1,..,\mu_n\}$, to będzie akcją nieobserwowalną, widoczną pod identyfikatorem *i.*

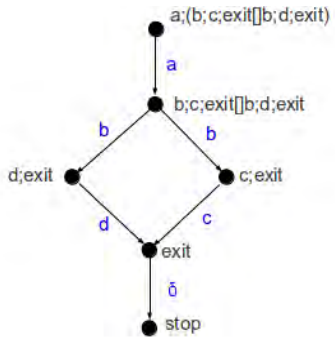
process identyfikator_procesu[a_1,..a_n] := B1 endproc – deklaracja i definicja procesu. W miejscu, w którym zostanie użyty *identyfikator_procesu,* zostanie wykonane wyrażenie *B1*, przy czym $a_1,..a_n$ są *bramkami* procesu. Proces ten można porównać do definicji procedury w języku strukturalnym. Wyrażenie *B1* może zawierać wywołanie innych procesów, a także procesu, którego stanowi definicję (rekurencja).

Semantyka poszczególnych wyrażeń zachowania języka LOTOS jest mozliwa do precyzyjnego okreslenia przy użyciu reguł wnioskowania. Reguły te umożliwiają skonstruowanie *etykietowanego systemu przejść (labeled transition system LTS)*, dla wyrażenia zachowania. Etykietowany system przejść określa, jakie stany może przyjąć wyrażenie zachowania, które połączone są etykietami akcji występującymi pomiędzy poszczególnymi stanami. Stany etykietowanego sytemu przejść stanowią wyrażenia zachowania, do jakich przechodzi kolejno wyrażenie zachowania, pod wpływem zachodzenia poszczególnych akcji.

W celu zilustrowania powyższemo, można rozwazyć następujące przykładowe wyrażenie:

$$a;(b;c;exit[]b;d;exit) \tag{1}$$

Zgodnie z opisem semantyki języka LOTOS, wyrażenie zachowania definiuje następująca kolejność zachodzenia akcji: jako pierwsza występuje akcja *a*, następnie może wystąpić wyrażenie *b;c;exit* albo *b;d;exit* (połączone operatorem wyboru niedeterministycznego). Alternatywne wyrażenia mogą wygenerować ciągi akcji, które porzedstawiają się odpowiednio: b,c,δ lub b,d,δ. Akcja δ jest specjalną akcją, którą generuje wyrażenie *exit* i oznacza zakończenie wyrażenia sukcesem. W związku z tym, etykietowany system przejść wyrażenia (1) ma postać grafu, takiego jak przedstawiono na rys. 1. Wierzchołki grafu stanowią stany wyrażenia, krawędzie grafu są oznaczone etykietami, odpowiadającymi akcjom, jakie zachodzą w czasie przejścia pomiędzy stanami.



Rysunek 2. Etykietowany system przejść wyrażenia(1)

2. Formalizacja języka BPMN przy użyciu języka LOTOS

Do formalizacji języka BPMN wykorzystano język LOTOS. Podobnie jak w [4], gdzie zaprezentowane jest odwzorowanie języka BPEL do języka LOTOS. Odwzorowanie notacji BPMN do algebry procesowej w wersji CSP przedstawiono w [5].

W niniejszym artykule zostało zaprezentowane odmienne podejście do odwzorowania, które charakteryzuje się prostszym wyrażeniem algebry procesowej (w języku LOTOS), z mniejszą ilością stanów i przejść etykietowanego systemu przejść. Użyte zostały dwa odwzorowania, które przyporządkowują wyrażenie zachowania języka LOTOS fragmentowi języka BPMN. Odwzorowania opierają się na poniższej koncepcji ogólnej:

- każdej czynności języka BPMN odpowiada proces języka LOTOS, którego wykonanie powoduje wygenerowanie akcji obserwowalnej (o identyfikatorze równym etykiecie czynności języka BPMN), a następnie wywołanie procesu języka LOTOS, który odpowiada przepływowi do następnych czynności, zdarzeń lub bramek,
- każdemu przepływowi (zarówno przepływowi sekwencyjnemu jak i przepływowi komunikatu) odpowiada proces języka LOTOS, którego wykonania powoduje wywołanie procesu czynności, zdarzenia lub bramki, do którego prowadzi dany przepływ.

W ten sposób rekurencyjnie określany jest cały proces, rozpoczynający się od zdarzenia początkowego. Takie odwzorowanie umożliwia wyrażenie dowolnego połączenia pomiędzy czynnościami, w tym zapętleń, na które pozwala notacja BPMN. Ponadto, dzięki takiemu podejściu, etykietowany system przejść, jaki powstaje z wyrażenia języka LOTOS, jest możliwie najmniejszy, co pozytywnie wpływa na czas analizy i wnioskowania na temat procesu BPMN.

2.1. Odwzorowanie notacji BPMN do języka LOTOS

W tabeli 1 i 2 przedstawiono w formalny sposób odwzorowanie kluczowych elementów notacji BPMN .

Zbiory: **A** – czynności i zdrzenia BPMN, każda czynność lub zdarzenie to dwójka *(id, typ)*, gdzie $id \in N$, $typ \in$ *{activity, eventStart, eventIntermediate, eventEnd}*.

G – bramki, każda bramka to dwójka (id, typ), gdzie id ∈ N, typ ∈ {xor, event-based, parallel, …}.

T – przepływy sekwencyjne, każdy przepływ to trójka *(from, to, id)*, gdzie $from \in A$, $to \in A$, $id \in N$.

C – komunikaty, każdy komunikat to czwórka *(from, to, id, visibility)*, gdzie $from \in A$, $to \in A$, $id \in N$, $visibility \in$ *{internal, external}*.

P – procesy, każdy proces to podzbiór iloczynu kartezjańskiego (A x G x T x C).

Odwzorowania:

DL(fragment BPMN) – odwzorowanie fragmentu języka BPMN na definicję procesów języka LOTOS.

BL(fragment BPMN) – odwzorowanie fragmentu języka BPMN na wyrażenie zachowania, które może być używane wewnątrz procesów języka LOTOS.

Tabela 1. Odwzorowanie notacji BPMN do języka LOTOS

Element **BPMN**	**Odwzorowanie DL**	**Odwzorowanie BL**

Czynność
 	
AID – identyfikator czynności
 	
LABEL – etykieta czynności
	DL({a ∈ A: a.id=*AID*})->`/* bez przepływów komunikatu*/`
	`process act_AID[null] :=`
	`  LABEL;`**BL({t ∈ T: t.from=*AID*})**
	`enproc`
	`/* albo */`
	`/* z przepływem komunikatu*/`
	`process act_AID[null] :=`
	`  LABEL;`**BL({c ∈ C: c.from=*AID*});**
	BL({t ∈ T:t.from=*AID*})
	`endproc`
	+
	DL({t ∈ T: t.from=*AID*})
	+
	DL({c ∈ C: c.from=*AID*});
		BL({a ∈ A: a.id=*AID*})->*act_AID*[null]

 Zdarzenie	Tak jak czynność	Tak jak czynność

Przepływ Sekwencyjny
 	
TID – identyfikator przepływu
 	
TOID – identyfikator elementu docelowego
 	
TID1..TIDn – kolejne identyfikatory przepływów wychodzące z elementu
	DL({t ∈ T: t.id=*TID*})->`process trans_TID[null] :=`
	BL({a ∈ A: a.id=*TOID*})
	`endproc`
	+
	DL({a ∈ A: a.id=*TOID*})
		`/*jeden przepływ*/`
		BL({t ∈ T: t.id=*TID*}) ->`trans_TID[null]`
		
		`/*wiele przepływów, wychodzących lub wychodzących z czynności AID*/`
		BL({T1 ⊆ T: T1.from/T1.to=*AID*}) ->`(trans_TID1[null]||[s_AID]||`
		`trans_TID2[null]||[s_AID]||`
		`...`
		`trans_TIDn[null])`

Tabela 2. Odwzorowanie notacji BPMN do języka LOTOS

Element **BPMN**	**Odwzorowanie DL**	**Odwzorowanie BL**

Przepływ Komunikatu
 	
CID – identyfikator przepływu
 	
TOID – identyfikator elementu docelowego
 	
LABEL – etykieta komunikatu
	DL({c ∈ C: c.id=*CID*})->`/* cel przepływu znajduje się wewnątrz procesu */`
	process comm_*CID*[null] :=
	LABEL; **BL({a ∈ A: a.id=*TOID*})**
	endproc
	`/* albo */`
	`/* cel przepływu znajduje się na zewnątrz procesu */`
	process comm_*CID*[null] :=
	LABEL; stop
	endproc
	+
	DL({a ∈ A: a.id=*TOID*})
		`/*jeden przepływ*/`
		BL({c ∈ C: c.id=*CID*}) ->`comm_CID[null]`
		
		`/*wiele przepływów, wychodzących lub wchodzących do czynności AID*/`
		BL({C1 ⊆ C: c.from/c.to=*AID*}) ->`(comm_CID1[null]|||`
		`comm_CID2[null]|||`
		`...`
		`comm_CIDn[null])`

Bramka
 	
GID – identyfikator bramki
	DL({*g* ∈ *G* : g.id=*GID*})->`/*bramka XOR */`
	process gate_*GID*[null] :=
	BL({t_1 ∈ T: t_1.from=*GID*})
	[]
	BL({t_2 ∈ T: t_2.from=*GID* })
	[]
	...
	BL({t_n ∈ T: t_n.from=*GID* })
	endproc
	
	`/*albo bramka parallel początkowa*/`
	process gate_*GID*[null] := c
	BL({t_1 ∈ T: t_1.from=*GID*})
	|[s_GID]|
	BL({t_2 ∈ T: t_2.from=*GID* })
	|[s_GID]|
	...
	BL({t_n ∈ T: t_n.from=*GID* })
	Endproc
	`/*bramka parallel końcowa*/`
	process gate_*GID*[null] :=
	BL({t ∈ T: t.from=*GID*})
	Endproc
	+
	DL({t ∈ T: t.from=*GID*})
		BL({*g* ∈ *G* : g.id=*GID*})->`gate_GID[null]`

Proces *PID* – etykieta procesu	**DL({p ∈ P: id=*PID*})**->`process proc_PID[{`C_{ext}`.id ⊆ C: visibilty=external}] :=` `hide [{`C_{int}` ⊆ C: visibilty=internal} in` **BL({a ∈ A: type=eventStart})** Endproc + **DL({a ∈ A: type=eventStart})**	**BL({p ∈ P: id=*PID*})**-> `proc_PID[[{`C_{ext}` ⊆ C: visibilty=external}]`

Uwagi do notacji odwzorowania:

- Symbol „+” oznacza połączenie wyrażeń zachowania (konkatenację).
- W kolumnie Element BPMN, oprócz symbolu elementu, znajdują się oznaczenia dodatkowych własności instancji elementu, używane w odwzorowaniu.
  Odwzorowanie DL procesu *BP* z rys. 1 znajduje się w listingu 1.

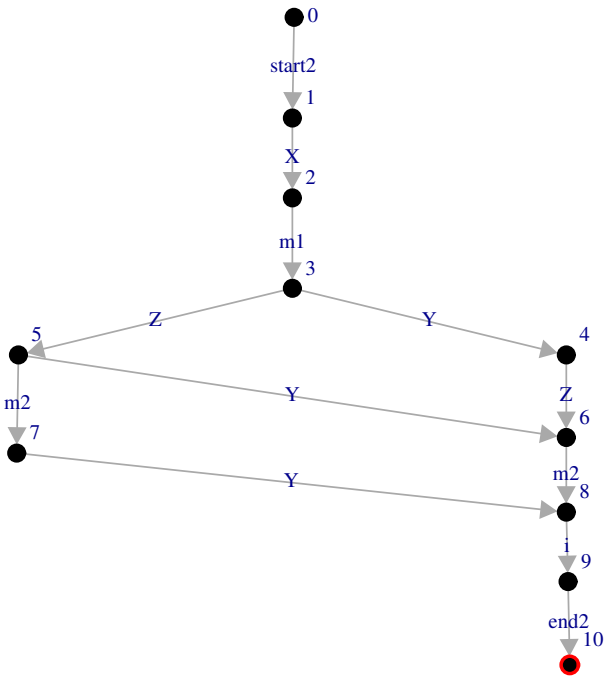
```
process proc_PB[m1,m2] :=
  hide s_G3 in
  act_start2[null]
endproc

process act_start2[null] :=  start2;trans_t10[null]endproc
process trans_t10[null]  :=  act_X[null]            endproc
process act_X[null]      :=  X;m1;trans_t11[null]   endproc
process trans_t11[null]  :=  gate_G2[null]          endproc

process gate_G2[null]    :=
  trans_t12[null]
    |[s_G3]|
  trans_t13[null]
endproc

process trans_t12[null] :=   act_Y[null]           endproc
process trans_t13[null] :=   act_Z[null]           endproc
process act_Y[null]     :=   Y;trans_t14[null]     endproc
process act_Z[null]     :=   Z;m2;trans_t15[null]  endproc
process trans_t14[null] :=   s_G3;gate_G3[null]    endproc
process trans_t15[null] :=   s_G3;stop             endproc
process gate_G3[null]   :=   trans_t16[null]       endproc
process trans_t16[null] :=   proc_end2[null]       endproc
process proc_end2[null] :=   end2;stop             endproc
```

Listing 1. Odwzorowanie DL procesu BP



Rysunek 3. Etykietowany system przejść odwzorowania DL procesu PB

Wyrażeniu zachowania, które powstało w odwzorowaniu procesu PB (*proc_PB[m1,m2]*) odpowiada etykietowany system przejść, który jest przedstawiony na rys. 3. Przejścia pomiędzy stanami odpowiadają etykietom czynności procesu w języku BPMN. Z diagramu przedstawionego na rys. 3, z perspektywy procesu BPMN *PB* wynika, że najpierw następuje zdarzenie *start2*, następnie czynność *X* oraz jest przesyłany komunikat *m1*, po czym wykonywane są czynności *Y* i *Z*. Trzeba jednak zaznaczyć, że w procesie *PB* czynności te są wykonywane równolegle, i z tego względu w etykietowanym systemie przejść odwzorowania *DL* czynności Y i Z mogą zajść w dowolnej kolejności. Po wykonaniu czynności *Y* i *Z* występuje nieobserowowalna akcja *i*, która odpowiada synchronizacji na

bramce *G3*. Ostatnią akcją, która ma miejsce jest zdarzenie *end2*. Jak wynika z powyższej analizy taka sekwencja czynności jest zgodna z definicją diagramu BPMN z rys. 1

2.2 Wykorzystanie odwzorowania w badaniu ewolucji

Dzięki wykorzystaniu odwzorowania *DL* można badać ewolucję systemów w architekturze usługowej, analizując etykietowane systemy przejść w następujący sposób:

- weryfikacja równoważności dwóch różnych realizacji procesu usługi. Dwa procesy orkiestracji są równoważne, jeżeli jest spełniona relacja bisymulacji obserwowalnej pomiędzy wyrażenia języka LOTOS, które powstają z odwzorowania *DL*. Z równoważności wynika brak potrzeby zmiany w procesie biznesowym, który korzysta z procesu usługi,
- zgodność procesu biznesowego z procesem usługi, który wspiera proces usługowy – poprzez porównanie identyfikatorów i kolejności komunikatów przepływających pomiędzy procesem biznesowym a procesem usługi.
- identyfikacja zmiany w procesie usługi, która jest wymagana na skutek zmiany w procesie biznesowym.
- identyfikacja zmiany w procesie biznesowym, która jest wymagana na skutek technologicznej zmiany w procesie usługi.

3. Podsumowanie

Zaprezentowana w przedmiotowym artykule formalizacja notacji BPMN w języku LOTOS umożliwia wnioskowanie o właściwościach zachowania procesów zapisanych w notacji BPMN. Przy użyciu odpowiedniego sposobu modelowania systemów usługowych oraz interpretacji wniosków wynikających z przedstawionego odwzorowania, można zastosować przedstawioną formalizację w procesie nadzoru i badania ewolucji systemów usługowych.

Praca jest finansowana ze środków Ministerstwa Nauki i Szkolnictwa Wyższego w ramach grantu nr 5321/B/T02/2010/39.

Bibliografia

[1] Erl T. *Service-Oriented Architecture: Concepts, Technology and Design.* Perentice Hall PTR, August 2005
[2] International Organization for Standards *Information Processing Systems: Open Systems Interconnection: LOTOS: A Formal Description Technique Based on the Temporal Ordering of Observational Behaviour.* ISO 8807:1989,1989
[3] Object Management Group. *Business process model and notation (BPMN), version 2.0.* Technical report, Needham, MA, August 2009
[4] Ratkowski A. *Projektowanie transformacyjne procesów w architekturze usługowej*. Rozprawa doktorska 2011 r.
[5] Wong P. Y. H., Gibbons J. *A Process Semantics for BPMN*. ICFEM 2008
[6] Zalewski A., Kijas S. *From Principles to Details - Integrated Framework for Architecture Modelling of Large Scale Software Systems*, CEE-SET 2011
[7] Zalewski A., Kijas S., Sokołowska D. *Architecture Evolution with Maps of Architectural Decisions 2.0,* ECSA 2011
[8] Zalewski A. i inni. *W kierunku metodyki ewolucji systemow o architekturze uslugowej* Krajowa Konferencja Inżynierii Oprogramowania KKIO 2011

QEMU CPU Tracer – an exact profiling tool

Sławomir Chyłek

Warsaw University of Technology, Institute of Computer Science

Abstract:

This paper presents a novel approach to collect and analyse information regarding code executed by the computer's processor. The motivation for this work is driven by the need to precisely answer the question which instructions are actually executed by the processor. Profiling tools and debugging tools can provide the answer for user space programs, although there is no tool that supports profiling both the user and the kernel space code without affecting the software under test. The basis of this work is the QEMU computer system emulator. It was utilized to develop a tool called QEMU CPU Tracer (QCT). The tool fulfils a requirement of exact code profiling, has acceptable performance and can also be utilized as a debugging utility thanks to the ability of tracking the execution of user space applications. In the paper a detailed description of QCT architecture is followed by a presentation of results of conducted experiments, a comparison to other similar solutions and a discussion over further research possibilities.

Keywords:

system emulation, profiling, fault injection

1. Introduction

Dependable computing is a constantly growing branch of information technology industry. Computer systems of all types are expected to assure the highest reliability and availability – from servers, through personal computers to mission critical embedded systems like the Anti-Lock Braking System (ABS). Apart from hardware solutions researchers also focus on software techniques that increase fault tolerance. To evaluate such solutions in the laboratory environment fault injection tools have been developed to alter the programs' execution by injecting faults of many kinds (e.g. bitflips in memory, stuck at 1's and 0's, failed memory read/store) [1]. Experience gained with our in-house implemented fault injection framework QEMU Fault Injector (QEFI) [2, 3] showed a need to point out most the frequently executed kernel code. This step is required to narrow down components that have to be evaluated against susceptibility to faults, since every piece of software has an innate immunity to faults at some level [4] making fully random testing inefficient. Moreover the most frequently executed code is most likely to be altered by faults, thus pointing out such code in complex projects like monolithic Linux kernel is valuable information.

To address the stated problem a QEMU CPU Tracer (QCT) has been developed. It is a novel profiling tool capable of tracing all jumps taken by CPU during the code execution. It is an exact profiler since a full trace of code paths taken by CPU is collected. QCT is based on the QEMU computer system emulator [5], it supports profiling both single process and a whole operating system depending on the QEMU's invocation mode. QCT's mode of operation is similar to code coverage tools, although a binary code is examined while no instrumentation is introduced. In this paper QCT is compared with other well-known profiling tools gprof and oprofile to show the differences in profiling sample applications. Results of

profiling the emulated operating system are also presented. Research conducted with QCT and QEFI are aimed to determine a direction of future works towards a more robust operating system's code.

The remainder of the paper is organized as follows. In Section 2 a detailed architecture of QCT is presented. Section 3 describes performed experiments with a discussion. Finally, we describe future work in Section 4.

2. QCT architecture

QCT has been designed in a multi-component fashion and is focused on performance. The approach of collecting every CPU's jump is a performance demanding task, since a large amount of data is being processed. To achieve data throughput at high level QCT has been divided into three components: a modified QEMU, a trace compressing service and post processing tools. The profiling experiment involves running a modified QEMU which sends profiling data over inter-process communication to a compressing service. The service stores an execution trace in a file for further analysis with post processing tools. A detailed description of each component is given below.

2.1. QEMU

QEMU is a sophisticated emulator that supports multiple architectures. It has two strategies for performing emulation: virtualizing with Kernel-based Virtual Machine (KVM) when a host's architecture matches the emulated one or runtime binary translation where blocks of target system instructions are decoded and translated into native instructions which manipulate a data structure representing a target's processor state. While virtualizing is faster, the binary translation is more flexible and enables inserting arbitrary code which is executed during emulation without altering the emulation target. Modifications introduced by QCT are an extension of our previous work described in [6]. These modifications include disabling virtualization feature (so in every case the binary translation path is taken) and altering the decoding procedure of binary translation, so before every jump instruction an extra code is executed to log the jump's source and the destination addresses. With these modifications it is possible to log a full trace of the processor's activity. Data produced by the specified method is stored in a buffer and periodically sent to compressing service over an inter-process pipe.

QEMU can operate in two different modes: a full system emulation mode and a single process emulation mode. The former emulates a whole computer system (i.e. CPU, buses, storage devices, etc.). The latter executes a specified binary by emulated CPU while system calls are forwarded to the underlying operating system or mocked by QEMU itself. QCT supports operating in both modes, although during the operating system emulation only the kernel-space code is being traced. It was accomplished by collecting data only when the emulated CPU has I/O privilege flag set to 0 (i.e. privileged mode). This decision was motivated by the inability to distinguish user-space processes without monitoring emulated operating systems internal data structures and decreased performance compared to profiling the application directly in the single process emulation mode.

2.2. Compressing service

Execution traces are stored in a compressed form, since a large amount of data can be produced by profiling. During experiments it was observed that a trace of booting emulated operating systems can produce a 500 MB GZip compressed trace file with compression ratio

100:1. The compressing service has been separated from QEMU process, since it would slow down emulation and thanks to this architecture multi-processor computer systems can be utilized.

The service additionally provides an interface to control the process of storing data. The user can interactively stop/resume flushing data to a file or change destination file, so during long-term experiments separate trace files can be utilized to store different data from different stages of an experiment.

2.3. Post processing tools

Post processing tools are responsible for presenting collected data in a user-friendly form. To achieve this all addresses are decorated with symbol names. However, to complete this task some extra steps have to be performed. The issue is caused by loadable symbols (shared libraries and kernel modules), which are loaded into determined at runtime memory areas. The solutions implemented by QCT are different for a single process profiling mode and for an operating system profiling mode. In case of the single process emulation the QEMU's feature to log invocation of system calls is utilized to intercept offsets in memory under which shared libraries are loaded (shared library is loaded by linker by calling mmap system call). These offsets combined with data from the static analysis of binaries (i.e. with ldd and objdump utilities) allow computing symbol names from runtime addresses. This process is fully automated, although in case of profiling an operating system the user's interaction is required. During the session with emulated system a user has to copy a /proc/kallsyms file somewhere into virtual hard drive, so it can be accessed later. The file contains a memory map of system symbols including symbols loaded by kernel modules and can be utilized by post processing tools for address translation. The solution has a drawback that the kallsyms file contains a memory snapshot of the point in time when it was created, so if some kernel modules were loaded or unloaded the file contents gets obsoleted.

Post processing tools can operate in two modes: the trace mode and the summary mode. This distinction was made for performance reasons. In the trace mode jumps taken by the emulated CPU are printed sequentially, so the user can examine the exact execution path of the program (e.g. it is possible to follow the process of binding a shared symbol during the first call and taking the short-path during next calls). During processing the trace file statistics of occurrences of the very same jumps (i.e. source and destination addresses are the same) are also computed. This mode can be useful not only for profiling, but also for debugging. Nevertheless applying the trace mode on 500 MB compressed trace file would be time inefficient and would create an illegible result, so a summary mode was introduced. In this mode during the trace file decompression data packages are sent to multiple processes which just generate occurrence statistics. Parallel processing breaks the ability to examine exact execution paths, although it allowed to process 500MB compressed trace file on a 16-core machine in 30 minutes. Occurrence statistics are stored as Coma Separated Value (CSV) files - a portable format accepted by many programs that can be used for visualization (e.g. Excel or R Project).

2.3. Limitations

The presented architecture allows collecting exact information regarding the binary code execution. However, the proposed architecture has some limitations. There is no way to automatically mark anyhow runtime generated code (e.g. by Just-in-time compilers) – it would be always marked as unknown symbol. Unfortunately not every application can be run in a single process emulation mode under QEMU – some syscalls are just mocked or not

supported (e.g. an attempt to run gedit text editor failed). Another drawback is that the process of collecting kallsyms file depends on having a virtual hard drive and forces to run an operating system emulation long enough to issue the appropriate commands, so profiling cannot be stopped at an early bootup stage.

3. Experiments

Experiments conducted with QCT are grouped into two sets. In the first set QCT is compared with other well-known profiling utilities oprofile and gprof to point out the differences between the tools. In the second set Linux kernel is profiled under different scenarios to prove QCT's applicability.

To compare QCT with other profiling tools it is needed to state that all of selected profilers produce different kinds of profiling data. While QCT's output is a set of entries with jump information that can be analysed in different ways (e.g. how many times a function was called, how many times a loop in a function was run or how many times a specific conditional branch was executed) oprofile provides just a number of times a symbol was hit when a triggering event occurred and gprof produces a combination of symbol hits (time triggered) and function call counts (collected thanks to instrumentation). In the tables below all QCT results are counted as a number of jumps taken with destination addresses to a specific symbol (i.e. it is a sum of the symbol calls, loop jumps and conditional jumps inside the symbol).

Table 1. wget profiling results

QCT [jump hits]		**oprofile [hits]**		**gprof [% hits, calls]**		
Logputs	4375500	dot_update	353	url_enqueue	33.33	32540
dot_update	2228807	logputs	257	retrieve_tree	22.22	569301
fd_read_body	1684413	fd_read_body	146	url_dequeue	11.11	519570
check_redirect_output	940342	select_fd	74	url_parse	11.11	148861
log_set_flush	445639	check_redirect_output	57	_fini	11.11	21846
Fputs	437551	ptimer_measure	55	url_escape_1	11.11	N/A
poll_internal	397496	print_row_stats	53	main	0.00	149349
fd_read	346948	fd_read	45	retrieve_from_file	0.00	49725
Logprintf	325802	poll_internal	31	__do_global_dtors_aux	0.00	49564
__divdi3	204054	log_vprintf_internal.isra.0	28	unescape_single_char	0.00	49482

Table 2. thttpd profiling results

QCT [jump hits]		**Oprofile [hits]**		**Gprof [% hits, calls]**		
tmr_mstimeout	247240	tmr_run	71	send_response	0.00	1189
tmr_run	233972	main	38	send_mime	0.00	662
httpd_got_request	69421	tmr_mstimeout	36	add_response	0.00	572
Main	57585	fdwatch	11	httpd_parse_request	0.00	244
Bufgets	39835	fdwatch_get_next_client_data	7	tdate_parse	0.00	170
fdwatch_check_fd	28576	httpd_parse_request	5	mmc_logstats	0.00	165
httpd_parse_request	13759	make_log_entry	5	check_referer	0.00	164
Fdwatch	13736	fdwatch_check_fd	4	httpd_got_request	0.00	164
fdwatch_get_next_client_data	13220	httpd_got_request	3	mmc_destroy	0.00	164
figure_mime	3886	auth_check	2	tmr_cancel	0.00	162

The test procedure for all profilers was to start a thttpd daemon instance and pull 300 MB of JPEG files using wget application. Both applications were profiled with QCT, oprofile and gprof. Profiling results are shown in Table 1 and Table 2. It is easy to spot that all utilities produce completely different results, although QCT's and oprofile's outputs are comparable, since they are much more precise than gprof. Inaccuracy of gprof is caused by following factors: samples are collected too rarely, only a part of application is instrumented (i.e. only code compiled with extra flags) and a number of symbol calls does not sufficiently reflect the effort of executing that symbol.

In Table 3 a summary of times needed to collect profiling data is given. Profiling application time in QCT is strictly related to the number of computations the target application performs. It is easy to spot that wget application is performing many operations, so the run under QCT is 4.34 times slower than under oprofile which does not introduce any overhead at all. In case of thttpd the application is mostly the invoking operating system, so profiling times do not vary significantly. This conclusion is also confirmed by results from Table 2 where gprof was unable to collect any symbol hits.

Table 3. Profiling times

Profiler	**wget program run [s]**	**thttpd program run [s]**
QCT	20.35	5.406
Oprofile	4.693	5.951
Gprof	5.552	4.915

Table 4. Linux kernel profiling results

Bootup [jump hits]		**Idle [jump hits]**	
inflate_fast	16895045	run_timer_softirq	89616
format_decode	10311134	get_next_timer_interrupt	87176
vsnprintf	4925058	sched_clock_local	78124
number	4922215	tick_nohz_stop_sched_tick	69082
unmap_vmas	4047305	ktime_get	57062
each_symbol_in_section	3255050	read_tsc	50615
find_symbol_in_section	3228306	delay_tsc	38452
strcmp	3195004	update_wall_time	34996
kallsyms_expand_symbol	2504431	tick_do_update_jiffies64	30985
find_next_bit	2359406	_spin_lock	22516
thttp [jump hits]		**wget [jump hits]**	
sk_run_filter	659920	sk_run_filter	993877
tcp_ack	497547	copy_to_user	369503
copy_from_user	347407	ext3_mark_iloc_dirty [ext3]	313769
tcp_write_xmit	251599	__find_get_block	284242
e1000_xmit_frame [e1000]	250805	netif_receive_skb	279191
netif_receive_skb	241343	do_get_write_access [jbd]	241034
tcp_v4_rcv	239837	packet_rcv_spkt	240635
skb_release_head_state	210342	kmem_cache_alloc	233149
__kfree_skb	182703	tcp_v4_rcv	223269
tcp_sendmsg	173379	skb_release_head_state	205389

Profiling Linux kernel was carried out in four scenarios: bootup sequence, idle and wget/thttpd programs execution like in the comparison of profilers, but with payload reduced to 3 MB to minimize size of a trace file. Profiling results are shown in Table 4. Data collected with QCT make possible to create a detailed profile of kernel operations under different load. Thanks to QCT it is possible to tell that during bootup sequence functions responsible for decompression and symbol management are quite frequently used, during idle operation scheduling, time and interrupts code is involved in comparable shares and during network operations sk_run_filter (run a filter on a socket) function is the most commonly invoked piece of code.

3. Conclusion and further research

Presented QCT solution is a novel approach to collect code execution statistics. It was designed to cooperate with QEFI fault injection framework and it fully satisfies the requirement of pointing out the most frequently executed code unlike tools based on statistical code sampling. Since it is based on QEMU it is possible to easily adopt QCT to other computer platforms like ARM (e.g. QEMU is a basis for Android Device Emulator). Future plans for development include integration of QCT with QEFI, so execution after fault injection may be recorded for further analysis and enriching post processing tools with the ability to create profiles not only based on jumps but also on the rest of instructions (e.g. ability to distinguish loops executing large portions of code and collect other statistics [7]).

References

[1] Arlat J., Crouzet Y., Karlsson J., Folkesson P., Fuchs E., Leber G.H.: *Comparison of physical and software-implemented fault injection techniques*. IEEE Transactions on Computers, t. 52, nr 9, pages 1115-1133, 2003

[2] Chyłek S., Goliszewski M.: *Wstrzykiwanie błędów oparte na modelach - Zastosowania QEMU w analizie niezawodności urządzeń mobilnych*. Zeszyty Naukowe Wydziału Elektroniki, Telekomunikacji i Informatyki Politechniki Gdańskiej, 2011

[3] Chyłek S., Goliszewski M.: *QEMU-based fault injection framework*. Zeszyty Naukowe Studia Informatica, Instytut Informatyki Politechniki Śląskiej, Vol. 33, No. 3B (108), 2012 (publication pending)

[4] Gawkowski P., Sosnowski J.: *Experiences with software implemented fault injection*. Proceedings of the 20th International Conference on Architecture of Computing Systems 2007

[5] Bellard F.: *QEMU, a Fast and Portable Dynamic Translator*. Proceedings on USENIX Annual Technical Conference, pages 41-46, 2005

[6] Chyłek S.: *Collecting program execution statistics with QEMU processor emulator*. Proceedings on IMCSIT '09. International Multiconference on Computer Science and Information Technology, pages 555-558, 2009

[7] Sosnowski J., Tupaj Ł.: *CPU testability in embedded systems*. Proceedings of the 5th Int. Symposium DELTA, IEEE Comp. Soc., pp. 108-112, 2010

Shortest path problem in static navigation situations

Mariusz Dramski

Maritime University of Szczecin

Abstract:

The main task of each navigator is to conduct safely the ship from the point of departure to destination. Although there are many different solutions of this problem, it's still necessary to carry out further research. This is dictated by the specific requirements that are specified, for example by the dynamics of the ship's own or reservoir characteristics.
This article presents a short review of different methods such Dijkstra, Bellman-Ford, Floyd or A algorithms applied to navigation problems. Besides some alternative methods based on artificial intelligence are mentioned. At the end a comparison of these solutions showed the advantages and disadvantages of each approach.*

Keywords:

shortest path, navigation, restricted area

1. Introduction

Every area has some specific restrictions which fact causes navigation problems. The navigator is equipped with maps which are the projection of the area. Hence, the problem boils down to build a grid of points and find a path in graph which was in this way created.

There is a lot algorithms to find a shortest path e.g. Dijkstra, Bellman-Ford, Floyd or A*. There are also some alternative approaches such artificial intelligence methods.

Listed algorithms were taken into research. As en example of alternative methods, SA-CO algorithm was chosen (simplified ant colony optimization). Few experiments were carried out. The aim of consideration was to check the average time of execution of each algorithm and the efficiency in finding a shortest (optimal) path in graph.

2. Some chosen approaches to solve the shortest path problem

2.1. Floyd algorithm

This algorithm is very useful if all the distances between each pair of graph's vertices are looked for. Let n be the number of vertices. Input data is a matrix of weights $A[i,j], 1 \leq i,j \leq n$. As a result a matrix of distances $D[i,j] = d(v_i, v_j)$ is obtained. The computational complexity of this algorithm is $O(n^3)$. Information given in the output data can be useful to find a shortest path between the point of departure of our object and it's destination. It has to be noticed that more data then needed is calculated. The point of departure is well known and constant in navigation problems, so Floyd algorithm gives more information which is not necessary and should be treated as excess data. More precise description of this algorithm can be found in [4].

2.2. Bellman-Ford algorithm

Bellman-Ford algorithm [4] gives as a result the shortest distances between the source s and the rest fo vertices in a oriented graph without minus cycles. Input data is an oriented graph $\langle V, E\rangle$ with n vertices, the source $s \in V$, and a matrix of edges' weights $A[u, v], u, v \in V$. Output data is a vector $D[v] = d(s, v), v \in V$ which contains the distances from the source to all vertices. The proceeding can be described in few steps:

- for each $v \in V$, $D[v] = A[s, v]; D[s] = 0$,
- for $k = 1$ to $n - 2$ do
    - for $v \in V \backslash \{s\}$ do
        - for $u \in V$ do $D[v] = \min(D[v], D[u] + A[u, v])$.

The computational complexity of this algorithm is $O(n^3)$ but if the execution of last loop doesn't cause any changes in $D[v]$, the calculations can be stopped.

2.3. Dijkstra algorithm

Dijkstra algorithm proposed in [1] is one of the most popular solutions for the shortest path problem. It finds the path with the lowest cost between a start vertex and every other vertex in the graph. This algorithm is often used in network routing protocols and many other problems where graphs can be applied.

The main steps of Dijkstra algorithm:

- Let s be the start node, $w(i, j)$ is the weight of edge i, j.,
- Create a distance matrix d for all the vertices of graph, assuming $d(s) = 0$ and $d(v) = \infty$ if there is no edge,
- Create a priority queue Q, where the priority is a distance from the start node s,
- Repeat until Q is not empty:
    - Remove from the queue the vertex u with the lowest priority,
    - For each neighbour v of the vertex u, $d(v) = \min\big(d(u) + w(u, v), d(v)\big)$,
- The last row of matrix d is a vector containing the shortest distance values from s to all the vertices of graph.

The computational complexity of Dijkstra algorithm is $O(n^2)$, although the use of appropriate data structure can reduce it [4].

2.4. A* algorithm

A* uses a best-first search and finds a least-cost path from a given source node to destination. A simple heuristic function $f(x) = g(x) + h(x)$ is applied, where $g(x)$ is the cost from the source node to the current node and $h(x)$ is an estimated distance from current node to destination. The aim is to find a path with $f(x) \rightarrow min$. The computational complexity of this approach is $O(n)$, but it is necessary to assert that A* search algorithm is able to find only the shortest path between the source and destination node (not all nodes of the graph).

2.5. SACO algorithm

SACO (Simplified Ant Colony Optimization) [2,3] is an alternative method based on artificial intelligence. This is the simpliest solution with the use of artificial ants. The insects are in one of two modes:

- ‘ahead’,
- ‘return’.

In the ahead mode ants move from the nest (original point) to the point where food is located (destination). When an ant achieves its goal its mode changes into the 'return' and the insect goes back along the same trail. Ants in the ahead mode randomly choose the next node in the graph $G = (N, A)$, where: N – set of all points in the graph, A – set of all branches in the graph, on condition that there exists a suitable branch. This random selection is determined by the amount of pheromone left on the path between nodes. In the SACO algorithm, apart from the actual path, its length is remembered, so that one can estimate how much pheromone an ant lays on a given path. In this way ants relatively quickly begin to travel along the shortest route. In this context a few principles should be applied:

- pheromone 'updating' takes place only on the way back,
- created loops should be eliminated,
- quality of solution is analyzed on the basis of the pheromone amount left.

Each branch (i, j) in the graph $G(N, A)$ is related to the variable:

$$\tau_{ij} = 1, \forall (i, j) \in A$$

When an ant k is found in the node i, the above variable can be used to calculate the probability of choosing the route to the node j:

$$p_{ij}^{k} = \begin{cases} \dfrac{\tau_{ij}^{\alpha}}{\sum_{i \in N_i^k} \tau_{ij}^{\alpha}} \text{ for } j \in N_i^k \\ 0 \text{ for } j \notin N_i^k \end{cases}$$

where:
N_k^i – neighbourhood of the ant k when it is in the node i. This neighbourhood contains all other nodes directed to i, except the preceding node (this does not refer to a situation where the node i is an extreme node – there is no other way than the way back.
α – parameter determined experimentally (value $\alpha = 2$) [2].
τ_{ij} – will be called the artificial pheromone trace. This trace is read out and recorded by ants. The pheromone update can be done according to this formula:

$$\tau_{ij}(t+1) = \tau_{ij}(t) + \Delta\tau_{ij} \quad \text{where } \Delta\tau_{ij} = 1$$

The phenomenon of pheromone evaporation is not taken account of in the SACO algorithm.

For the algorithm to work we have to define the graph of all possible paths that ants may follow. The correct creation of the graph is extremely important as it will make up a basis for the proper representation of the examined area in the form of points of a grid. How fast the ant algorithm will operate depends on the number of these points and their actual distribution.

The computational complexity of this algorithm is $O(n^3)$, which fact tells that this approach may not be the best solution, but it is still worth to include it into our area of consideration.

3. Experimental verification

3.1. Infinite graphs

For the purpose of experimental verification of given algorithm few fragments of infinite graphs were created. The kind of this graph is shown at Fig. 1 Every vertex of these fragments has 8 edges (sides of the squares and it's diagonals – except external vertices). The bottom left node (number 1) of every fragment is a start node for shortest path routing

algorithm and the top right one (number n) is the stop node. So it's easy to observe that the shortest path will always be the diagonal of the square. Two facts were taken into a consideration:

- the average time of finding a solution,
- the length of found path.

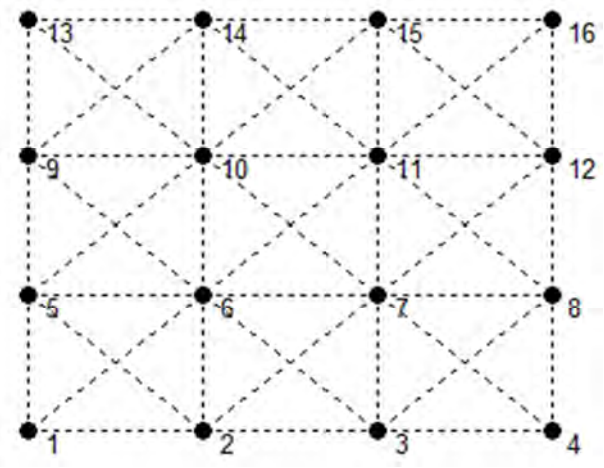


Figure 1. Infinite graph example

Table 1 illustrates the average times for the execution[1] of each algorithm for different fragments of infinite graphs.

Table 1. Average times of execution

Number of nodes	Floyd	B.-Ford	Dijkstra	A*	SACO
4	0.0043	0.0068	0.0157	0.0130	1.0595
9	0.0033	0.0050	0.0166	0.0105	1.7452
16	0.0163	0.0170	0.0076	0.0159	1.2820
25	0.0444	0.0466	0.0144	0.0179	1.8591
36	0.0931	0.0872	0.0240	0.0169	16.8955
49	0.1519	0.1551	0.0349	0.0189	2.6338
64	0.3013	0.3030	0.0435	0.0232	24.5784

Fig. 2 shows the times of execution of each algorithm in the logarithmic scale (due to the fact that the times for SACO algorithm are significantly longer). It illustrates all the results from Table 1.

[1] Intel® iCore™ i7-2760 QM CPU @ 2.20 GHz, 4 GB RAM

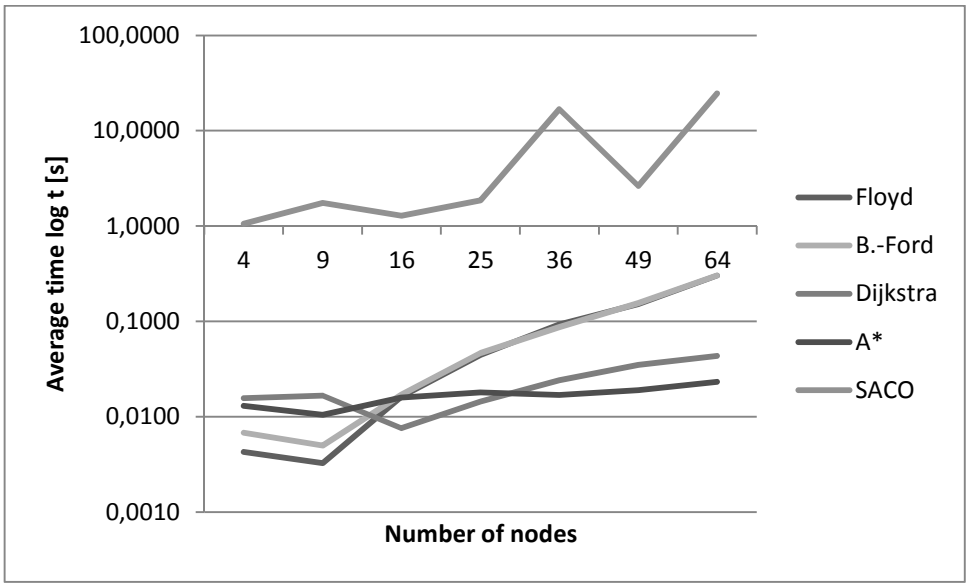


Figure 2. The times of execution

3.2. Example area

An example restricted area was created. It's illustrated at Fig. 3. The graph of possible paths is a set of simple square graphs which also can be treated as fragments of an infinite graph. The purpose of this choice was simple – easy to see the shortest path even without doing research.

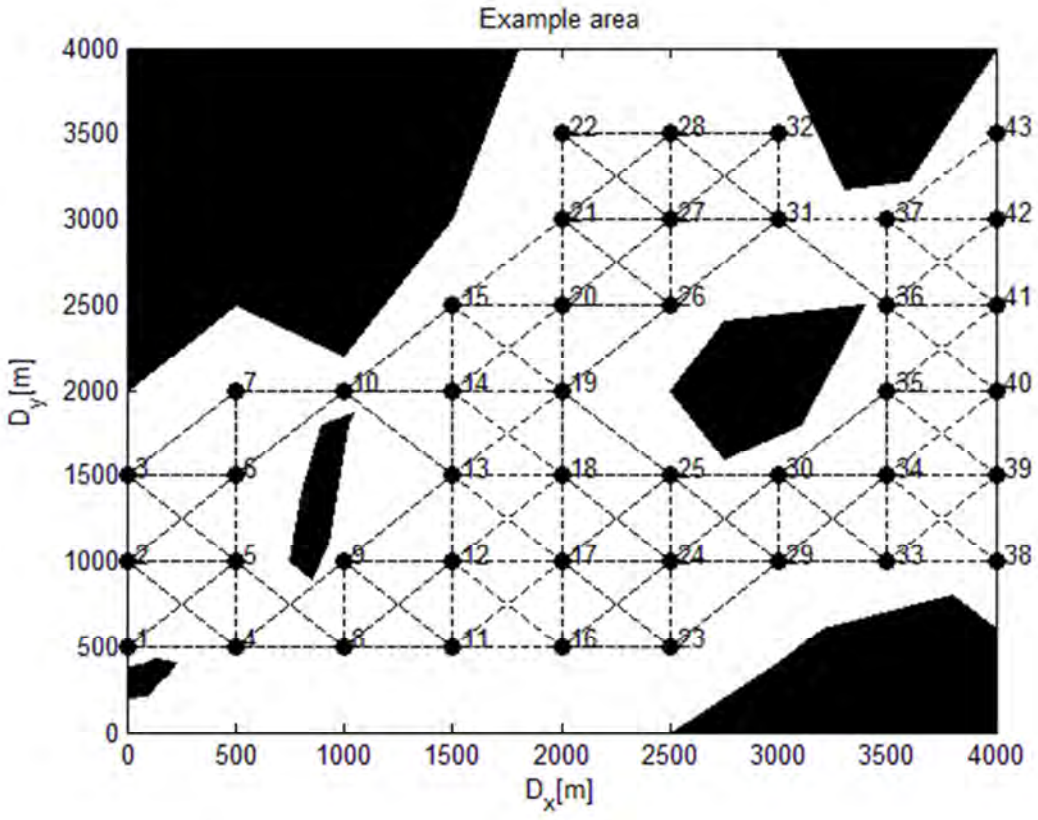


Figure 3. An example area with the graph of possible paths

Table 2. Comparison between Dijkstra and SACO algorithms – experiments' results

Algorithm	Route [nodes]	Route's length [nodes]	Route's length [m]	Time [s]
Floyd	1-4-9-13-19-26-31-37-43	9	5242.6	0.04331
Bellman-Ford	1-4-9-13-19-26-31-37-43	9	5242.6	0.06213
Dijkstra	1-4-9-13-19-26-31-37-43	9	5242.6	0.02483
A*	1-4-9-13-19-26-31-37-43	9	5242.6	0.01204
SACO	1-4-9-13-14-20-27-31-37-43	10	5535.5	12.49039

In Table 2 the results of experiment were illstrated. This data conducts to the conclusion that well known method of shortest path routing are better then proposed alternative method. Besides it has to be told, that in the case of SACO algorithm only the best obtained route is presented. This method does not guuarantee any optimality, so there is a problem with the repeatibility too. Every single use of SACO algorithm can result other solution.

Obtained paths can also be compared by analysing Fig. 7 and Fig. 8. As it can be seen both routes are safe and possible, but Dijkstra algorithm let it be quite shorter in this case.

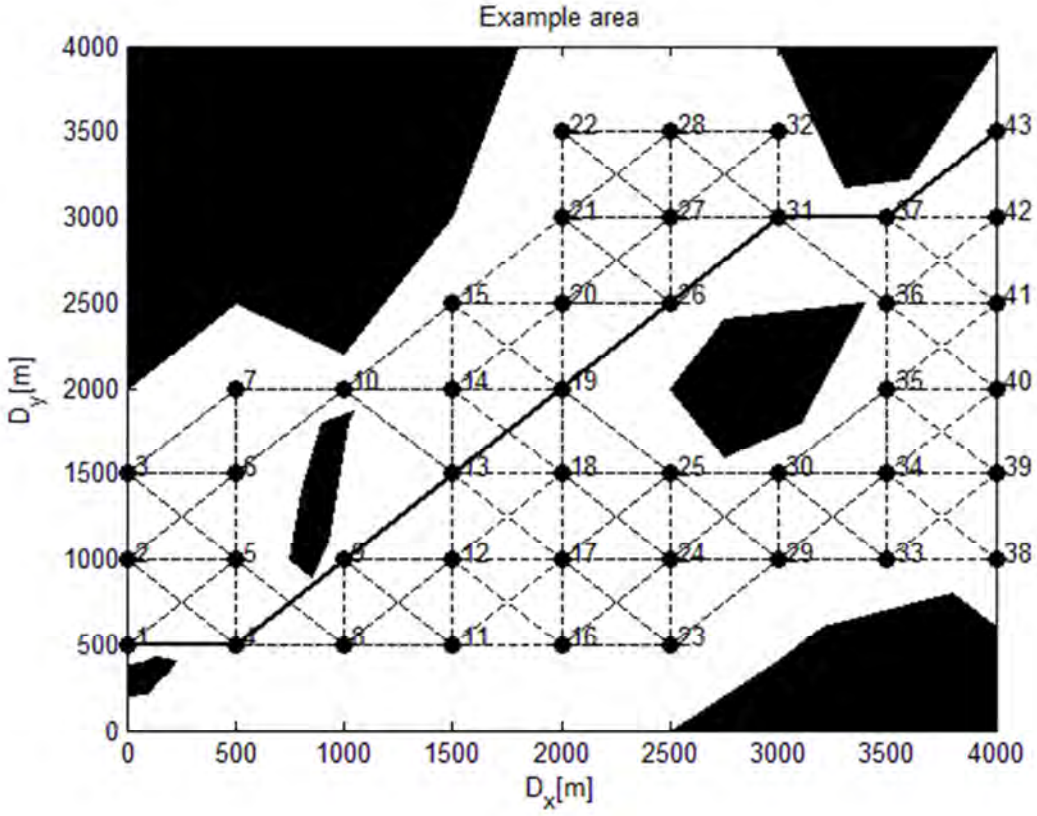


Figure 4. Classic methods' result

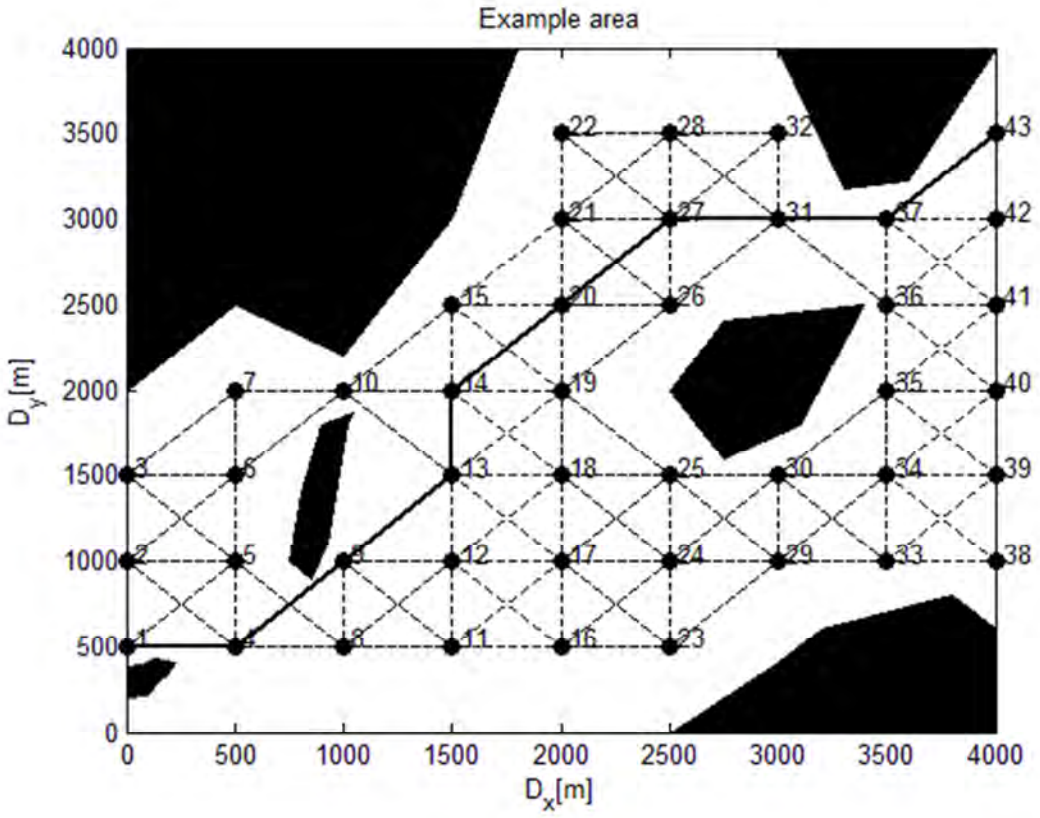


Figure 5. SACO algorithm result

4. Discussion

To solve the problem of navigate in a restricted area, few well known algorithms were investigated. The results lead to the following discussion:

- The fastest algorithm was A*. This fact was naturally expected because of the computational complexity of this method. So, this solution could be applied e.g. in a decision support system for navigators which requires real-time work. The A* algorithm has, however, one significant disadvantage. If the restrictions change dynamically, there is a need to repeat the whole process of finding a shortest path. Depending on the total number of nodes in the graph, the time of execution can raise significantly. Besides the definition of A* algorithm requires to estimate the geometric distance between current point and the point of destination. So it makes A* the solution which can't be applied in every area of consideration.
- Dijkstra algorithm was the second because of the average time of execution. This fact was also expected due to the computational complexity. Besides, this approach as output data is able to find a path to all remaining points of graph from the destination. So, Dijkstra algorithm is the most universal solution. Different examples of applies in the world literature only confirm this fact.
- Bellman-Ford algorithm is very similar to Dijkstra (except the fact that weigths of graph's edges can be negative) and gives the same result. The disadvantage is higher computational complexity which causes a longer time of execution.
- Floyd algorithm's execution time is practically the same as in Bellman-Ford. Besides this approach gives as as result the shortest distances between all the pairs of nodes in the graph. The output data structure is a matrix (not a vector like in Bellman-Ford or Dijkstra).

- SACO algorithm has the same computational complexity as Bellman-Ford or Floyd algorithm but due to the probabilistic character, the average time of execution is very long. Besides it was the unique solution which doesn't ensure the guarantee of optimality. Obtained route is only suboptimal which fact can even disqualify this approach. However alternative methods seem to be very interesting in further research. One of the advantages of SACO alforithm is that it can be stopped in every moment of execution – this lets to change the restrictions in dynamic mode.

This discussion can be also reduced to the informations summarized in Table 3 below.

Table 3. All algorithms' comparison

	Floyd	**Bellman-Ford**	**Dijkstra**	**A***	**SACO**
Optimality' guarantee	Yes	Yes	Yes	Yes	No
Computational complexity	$O(n^3)$	$O(n^3)$	$O(n^2)$	$O(n)$	$O(n^3)$
Time[2]	long	long	average	short	very long
Distances	between every node	from the source to all others	from the source to all others	from the source to destination	from the source to destination
Numbering of nodes	Free	Free	Free	Free	Start node must be 1, and the destination n
Route repeatibility	Yes	Yes	Yes	Yes	No

5. Conclusions

Obtained results and the discussion above, lead to the following conclusions:

- The fastest algorithm is A*.
- Every algorithm except SACO has a gurantee of optimality.
- Dijkstra algorithm is considered as the most universal one.
- Every algorithm can be used depending on the expectations and conditions.

References

[1] Dijkstra, E. W. *A note on two problems in connexion with graphs*. *Numerische Mathematik* 1, 1959, pp. 269–271

[2] Dorigo M., Stutzle T. *Ant Colony Optimization*. MIT Press, 2004.

[3] Dramski M., Mąka M., *The choice of ship's safe route in a restricted area with the use of quadtrees for a simplified ant algorithm*. Marine Traffic Engineering Conference MTE 2011, October 12-14, Świnoujście 2011

[4] Lipski W., *Kombinatoryka dla programistów*, WNT 2007

[5] Śmierzchalski R., Michalewicz Z. *Modeling of ship trajectory in collision situations by an evolutionary algorithm*. IEEE Transactions on Evolutionary Computation, Vol. 4, 2000, pp. 227-241.

[2] Simple linguistic evaluation based on the results obtained

Ontology based approach to Antarctic knowledge portal designing

Larisa Globa, Rina Novogrudskay

National Technical University of Ukraine "Kiev Polytechnic Institute"

Abstract:

The article reviews the main approaches to designing of the National Antarctic Data Centre Internet knowledge-portal. Basic modern technical solutions, underlying software and hardware complex of Internet portal are discussed. In particular, emphasis is made that development of a united information environment Antarctic research data will provide significant economic benefits across the country through the commercialization of their unique data.

Keywords:

Internet-portal of Antarctic data, knowledge, ontology model, Antarctic research.

1. Introduction

According to the requirements of Resolution 4.1 XXII ATCM in 1998 all Antarctic Treaty Parties engaged in Antarctic research are required to establish National Antarctic Data Centre (NADC). As the result of requirements it is the urgent task to develop the Centre in Ukraine. Effective functioning of NADC could help to solve some problems related to collecting, preserving, processing and transmission, of primary and processed results of Antarctic research to a wide range of scientists and scientific institutions in Ukraine, as well as interested foreign research organizations. Moreover, development of such an information environment will allow Ukraine to fully enter the world's Antarctic scientific community that increase the efficiency of our scientists and raise the international prestige of our state.

Functionally, the National Antarctic Data Centre will:

- develop a unified information environment for presenting the results in fields of Antarctic research of Ukrainian scientists in the Internet environment;
- provide the single access point to all kinds of scientific resources;
- use of these basic approaches, such as:
  - simultaneity of system development, implementation and use of data while providing simultaneous operation of all scientists;
  - standardization of common components of information processes;
  - integration as a method of interaction of individual components;
  - intellectualization as a method of necessary analytical and statistical data computing.

At present, Ukraine has successfully carried out extensive research in the Antarctic in various scientific fields. This process involved a large group of Ukrainian scientists and a number of prominent educational and research institutions. Scientists have accumulated large tracts of primary and processed scientific information stored in various formats. However, there is a problem of access to the results obtained and further processed. Important results, both theoretical and practical, which are obtained the researchers, are available only

on the level of individual research institutions. Unfortunately, there is a process of knowledge concentration in certain fields of science, and they are available only to a narrow circle of scholars and specialists. It is very important negative factor in the process of access to knowledge such as poor and weak ordering and structuring of large amounts of information in the currently existing storage systems [1].

In connection with all mentioned above the urgent task is to hold the generalization, classification, reduction to a single format of the data, and on this basis, enabling them to use a wide circle of knowledge for domestic and foreign scientific community, for whom they are intended.

To solve these problems it is proposed to construct specialized knowledge Internet-portal for work with big amounts of various informational and computational resources in a definite technical sphere. Such kind of portal can not only give the possibility to search and systematize the information, but can also help to realize definite computational user's tasks.

The aim of this research is to improve access to Antarctic scientific knowledge by means of designing specialized Internet portal of Antarctic research knowledge. The following tasks are to be solved:

- qualitative knowledge presentation on the portal;
- to systematize and structure information;
- to formalize the Antarctic scientific knowledge;
- to organize an affective and purposeful search.

It is necessary to design the model of knowledge representation at the portal of Antarctic scientific knowledge. It is clear that model of high quality designing to represent knowledge at the portal may allow realizing all the above mentioned requirements. Ontology was used as a model knowledge representation at the portal.

2. Portal ontology

The knowledge portal ontology in the field of Antarctic research was constructed based on the above descriptions. Formally the ontology may be specified as O = {C, A, R, T, F,D}, where *C* is the set of classes that describes the notions of a subject field; *A* is the set of attributes that describes the features of notions and relations; R is the set of relations specified for classes: $R=\{R_{AS}, R_{IA}, R_n, R_{CD}\}$,where R_{AS} is associative relation, R_{IA} is relation "is-are", R_n is relation of "heredity", R_{CD} is relation "class-data"; *T* is the set of standard types of attributes values; F is the set of limits for values of attributes notions and relations; D is the set of class exemplar [2].

The ontology of the Antarctic portal includes 2 ontologies: External ontology and internal ontology [3].

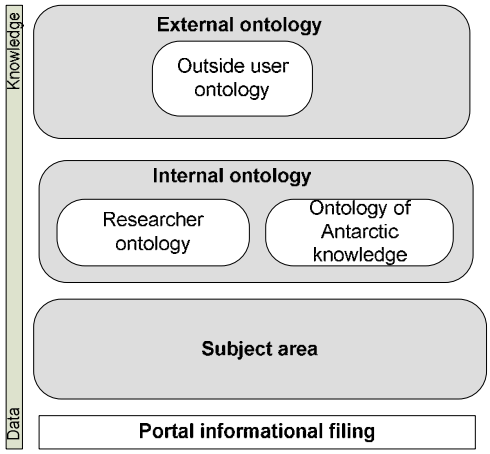


Figure 1. The portal ontology system

2.1. Internal ontology. Antarctic knowledge ontology.

Internal ontology includes Researcher ontology and Ontology of Antarctic knowledge. $O_2 \supset \{O_3, O_4\}$.

Antarctic knowledge ontology

Antarctic knowledge ontology (AKO) consists of six classes of concepts, these classes are metadescriptions that define the structure for the classes of the subject area. There are links between classes defined by relationship of types. Based on this AKO can be described as follows: $O_3 = \{C_{o_3}, A_{o_3}, R_{o_3}, T_{o_3}, F_{o_3}, D_{o_3}\}$, $C_{o_3} = \{C_1, C_2, C_3, C_4, C_5, C_6\}$, $A_{o_3} = (A_{C_1}, A_{C_2}, A_{C_3}, A_{C_4}, A_{C_5}, A_{C_6})$ $D_{o_3} = (D_{C_1}, D_{C_2}, D_{C_3}, D_{C_4}, D_{C_5}, D_{C_6})$.

$R_{o_2} = (R_{AS_1}(O_3),..., R_{AS_6}(O_3), R_{IA_1}(O_3), R_{n_1}(O_3), R_{CD_1}(O_3),..., R_{CD_5}(O_3))$, .

The method of investigation – $C_1(O_3)$. This class is used to describe the methods used in Antarctic research. Attributes include the class name of the method, its author. $C_1(O_3) = (A_{C_1}, D_{C_1})$.

The object of study – $C_2(O_3)$. The concepts of this class define typed objects of study and structure to describe them. Attributes include the name of the class of the object, its parameters (characteristics). $C_2(O_3) = (A_{C_2}, D_{C_2})$

The result of the study – $C_3(O_3)$. The concepts in this class are used to describe the results of scientific activities and their typing. The class contains the result of the results already obtained by scientists. Research results are usually reflected in publications. Attributes include the name of the class, by whom it was received, the required characteristics. $C_3(O_3) = (A_{C_3}, D_{C_3})$.

The aim of the study $-C_4(O_3)$. The concepts in this class reflect the purpose of the study and the challenges that must be addressed. Attributes class is the direction of research. $C_4(O_3) = (A_{C_4}, D_{C_4})$.

Research equipment – $C_5(O_3)$.The concepts in this class describe the equipment, which conducted the study. Attributes include the class type of equipment, name, year, size, some required for the experiment parameters and characteristics. $C_5(O_3) = (A_{C_5}, D_{C_5})$.

Branch of science – $C_6(O_3)$. This class allows you to structure science, to allocate it important sections and subsections. The concepts in this class describe areas of research undertaken at the Antarctic stations, as well as outside on the basis of data collected. Attributes include a science class, science section, the direction of research. $C_6(O_3) = (\mathrm{A}_{C_6}, D_{C_6})$.

The mapping object – $C_7(O_3)$. This class describes the format in which data will be presented on the portal. By mapping the attributes of the class attribute, text, graphic document, audio, video: $C_7(O_3) = (\mathrm{A}_{C_7}, D_{C_7})$.

Associative relations – $R_{AS}(O_1) = \{C_i(O_1) \times C_j(O_1)\}$:

"Applies to" – relates the method and type of research facilities, to which it applies. $R_{AS_3} = \{C_1(O_3) \times C_2(O_3)\}$.

"Carried on" – links the research method with research equipment. $R_{AS_4} = \{(C_1(O_3) \times C_5(O_3))\}$.

"Aimed at" – the appointment of research associates research method. $R_{AS_5} = \{C_4(O_3) \times C_1(O_3)\}$.

"Received on" – the result of research links with research equipment, which it was received. $R_{AS_6} = \{C_3(O_3) \times C_5(O_3)\}$.

"Conducted with" – the result of the study relates to the object of research. $R_{AS_7} = \{C_3(O_3) \times C_2(O_3)\}$.

"Uses" – section relates to the method of science research. $R_{AS_8} = \{C_6(O_3) \times C_1(O_3)\}$, $R_{AS_9} = \{C_6(O_3) \times C_3(O_3)\}$, $R_{AS_8} = \{C_6(O_3) \times C_5(O_3)\}$.

"Investigates" – a branch of science relates to the object and purpose of the research study. $R_{AS_{12}} = \{C_6(O_3) \times C_2(O_3)\}$, $R_{AS_{11}} = \{C_6(O_3) \times C_4(O_3)\}$.

"Visible" – the result of the study relates to the display object. $R_{AS_{13}} = \{C_7(O_3) \times C_3(O_3)\}$

Relation is-are – $R_{IA}(O_1) = C_k(O_1) \subset C_m(O_1)$

The ratio of "is-are" is the relationship between class research equipment and facilities for carrying out different types of experiments, the measuring equipment. $C_5 \subset C_{5_1} \wedge C_{5_2} \wedge C_{5_3} \wedge C_{5_4}$.

Inheritance relationship – $\mathrm{R_n}(\boldsymbol{O_1}) = \mathrm{a_i}, \mathrm{r_i} \mid \mathrm{A_{C_m}}(\boldsymbol{O_1}) \rightarrow \mathrm{a_i}, \mathrm{r_i} \mid \mathrm{A_{C_k}}(\boldsymbol{O_1})$

Relation of inheritance is used for class research equipment of its subclasses are set for different types of experiments, the measuring equipment. $\mathrm{A(C_5), R(C_5) \rightarrow A(C_{5_1}), R(C_{5_1})}$, $\mathrm{A(C_5), R(C_5) \rightarrow A(C_{5_2}), R(C_{5_2})}$, $\mathrm{A(C_5), R(C_5) \rightarrow A(C_{5_3}), R(C_{5_3})}$, $\mathrm{A(C_5), R(C_5) \rightarrow A(C_{5_4}), R(C_{5_4})}$.

Relation class-data – $R_{CD}(O_1) = C_j(O_1) \subseteq D_i(O_1)$. Such type of relation is used for all classes of this ontology.

The picture 3 shows classes and relations of AKO.

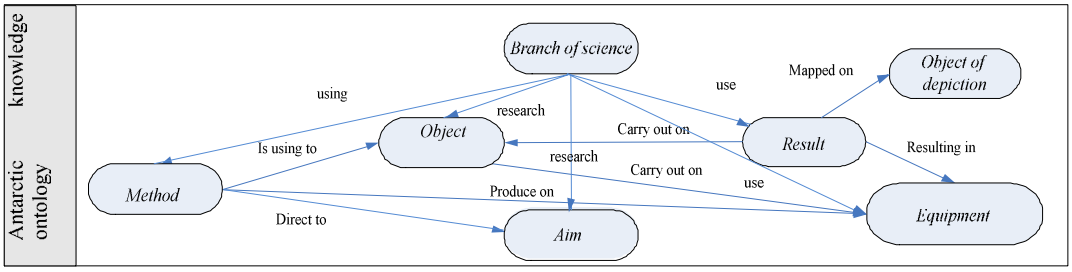


Figure 2. Antarctic knowledge ontology classes and connections

2.2 Internal ontology. Researcher ontology.

Research ontology (RO) – combines four classes of concepts that describe the opportunities provided by the researchers on the portal. The properties and characteristics of each concept are described using the attributes of concepts and restrictions on the range. Based on this OR can be described as: $O_4 = \{ C_{o_4}, A_{o_4}, R_{o_4}, T_{o_4}, F_{o_4}, D_{o_4} \}$, $C_{o_4} = \{C_1, C_2, C_3, C_4\}$,

$A_{o_4} = (A_{C_1}, A_{C_2}, A_{C_3}, A_{C_4})$, $D_{o_4} = (D_{C_1}, D_{C_2}, D_{C_3}, D_{C_4})$

$R_{o_2} = (R_{AS_1}(O_4),...,R_{AS_7}(O_4), R_{IA_1}(O_4), R_{n_1}(O_4), R_{CD_1}(O_4),...,R_{CD_5}(O_4))$

Researcher ontology includes the following classes of concepts:

Researcher – $C_1(O_4)$ This class includes scientists, researchers, explorers, who have access to data unavailable to the average user, and also have the opportunity to upload their own information and modify an existing one. Attributes of this class include personal data, degree, rank, scientific interests, rights of access. $C_1(O_4) = (A_{C_1}, D_{C_1})$

Scientific documentation – $C_2(O_4)$. This class is used to describe the scientific information available to authorized users only. Attributes include the name, description, author, type of access (read-only / read, and formatting). $C_2(O_4) = (A_{C_2}, D_{C_2})$

Investigation – $C_3(O_4)$. This class describes the problem posed by the researcher. Attributes include the name of the class, a branch of science, the name of the researcher or research team, the date of implementation. $C_3(O_4) = (A_{C_3}, D_{C_3})$

Research report – $C_4(O_4)$. This class is used to describe the data obtained from studies presented in the form of reports or raw data that are used in further scientific achievements. Attributes include: type of study, name, description, author, type of access. $C_4(O_4) = (A_{C_4}, D_{C_4})$

Ontology classes are related by the following relations :

Associative relations – $R_{AS}(O_1) = \{C_i(O_1) \times C_j(O_1)\}$:

Prepare" – connects researchers with a scientific report. $R_{AS_1} = \{C_1(O_4) \times C_4(O_4)\}$.

"Has access to the" – relates researchers with scientific documentation. $R_{AS_2} = \{C_1(O_3) \times C_2(O_4)\}$.

"Holds" – links researchers with the study. $R_{AS_3} = \{C_1(O_4) \times C_3(O_4)\}$.

Relation class-data – $R_{CD}(O_1) = C_j(O_1) \subseteq D_i(O_1)$. Such type of relation is used for all classes of this ontology.

The picture 4 shows classes and relations of RO.

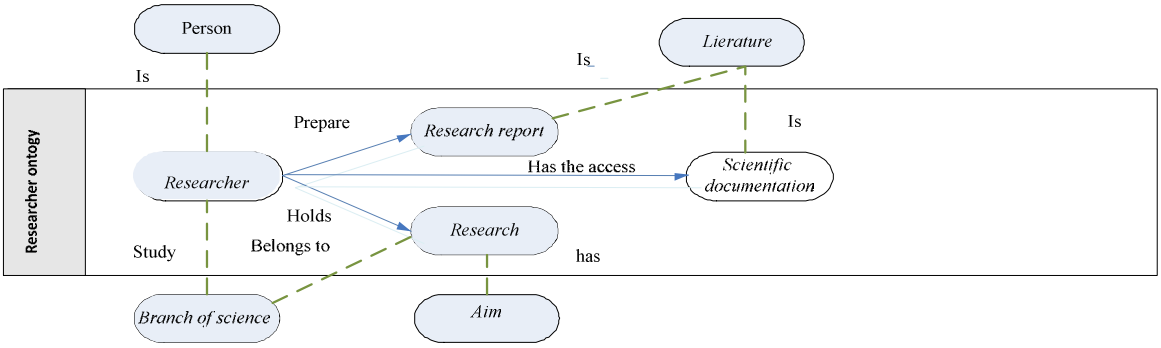


Figure 3. Researcher ontology classes and connections

2.3 External ontology

External ontology includes only the ontology for outside user. Outside user ontology (OUO) includes general classes of notions related to the organization of Antarctic research activities such as Person, Organization, Activity, Event, Literature, Documents, Teaching materials, Publication and Location: $O_1 = \{C_{o_1}, A_{o_1}, R_{o_1}, T_{o_1}, F_{o_1}, D_{o_1}\}$. This ontology was built in the same way as two ontologies described in paragraphs given bellow.

2.4 The ontology of subject arias

The ontology of subject area (SAO)presents the general domain knowledge, such as class hierarchy of concepts, semantic relationships in these classes. The basis for ontology portal for Antarctic knowledge in the field of Antarctic research was the decision of the Cabinet of Ministers of Ukraine [4].

Domain ontology describes the classification of research areas, which are held at the Antarctic stations (Fig. 5), in various branches of science includes formal and informal description of the concepts and relations between them [5].

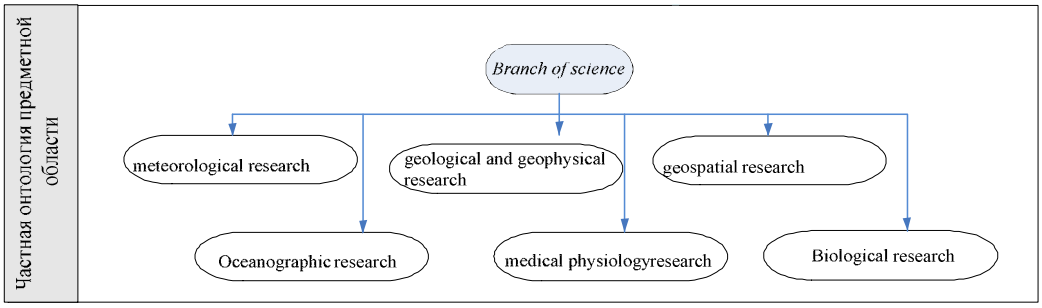


Figure 4. The classification of NADC research areas

These concepts are implementations of the Antarctic metaponyaty ontology knowledge, and can be arranged into a hierarchy of private and common-part-whole. For example, the objects of study correspond to natural phenomena such as the Antarctic: bottom topography and land, surface and underwater ecosystems, the electromagnetic field, etc., and as methods of research are concepts such as seismographic studies, probing the soil, weather, etc. Example domain for geological and geophysical studies is shown in (Fig. 6).

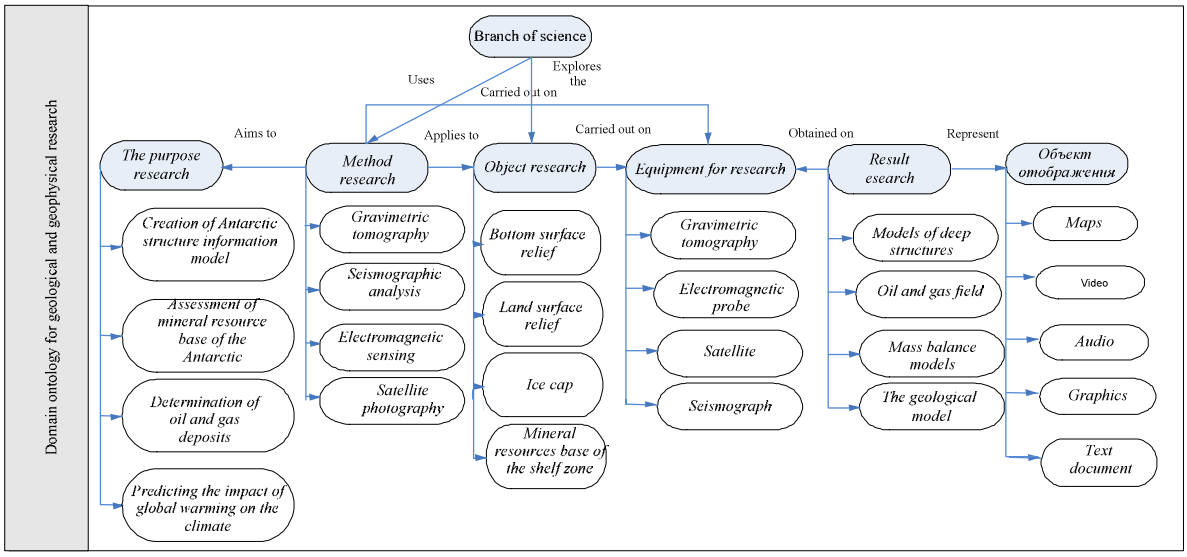


Figure 5. A fragment of a subject area model

3. Conclusion

We propose a conceptual model for building a portal of the National Antarctic Data Centre. Information base of the portal is presented like ontology, with the help of which the ordering and structuring of information is made, as well as an effective search and navigation of information space of Antarctic knowledge portal is organized.

Development and implementation of NTSAD as a single information environment for data of Antarctic research will provide significant economic benefits across the country through commercialization of their unique data.

References

[1] Боровикова О. И., Загорулько Ю. А., *Организация порталов знаний на основе онтологий*, Тр. междунар. семинара «Диалог 2002» Компьютерная лингвистика и интеллектуальные технологии (Протвино, 6–11 июня 2002 г.), стр. 76–82, М.: Наука, 2002. Т. 2.

[2] Глоба Л. С., Новогрудская Р.Л. *Подход к построению портала инженерных знании*, Труды конференции «Интеллектуальный анализ информации ИАИ - 2010», стр. 53-62, Киев: КПИ; 2010.

[3] Глоба Л.С., Новогрудская Р.Л. *Модель интернет-портала «Прочность материалов»*, Международный научно-технический сборник «Надежность и долговечность машин и сооружений», стр. 183-189, Киев 2011.

[4] Постанова Кабінету Міністрів України від 3 листопада 2010 р. N 1002 . «Про затвердження Державної цільової науково-технічної програми проведення досліджень в Антарктиці на 2011-2020 роки» [WWW документ]. URL http://zakon.rada.gov.ua/cgi-bin/laws/main.cgi?nreg=1002-2010-%EF (3 жовтня 2011).

[5] Загорулько Ю. А., Попов И. Г. *Описание сложных предметных областей на основе интеграции средств представления знаний*, Тр. междунар. семинара «Диалог'97» по компьютерной лингвистике и ее приложениям, стр. 110–115, М, 1997.

On asymmetry and fuzziness of cognitive-mental traits of personality, possibility of the mental worker productivity theory creation

Leonid Kompanets, Piotr Milczarski

University of Lodz, Department of Theoretical Physics and Computer Sciences

Education. How do we improve education? The single best way is to get them to understand that what they are being told is a whole lot of nonsense. ... *Pollution, energy shortage, environmental diversity, power* – how do we make stable societies? *Longevity,* there are lots of problems to worry about. ... The question I think people should talk about – and it's absolute *taboo* – is, how many people should there be? And I think it should be about 100 millions or may be 500 millions. (*The Authors*: It has over 7 billions of people on the Earth now).

We could be smarter, but we're not smart enough to deal with the colossal problem that we face, either in abstract mathematics or in figuring out economies or balancing the world around. So one thing we can do is live longer. ... We know that people live twice as long as chimpanzees, almost, and nobody lives more than 120 years, ... But lots of people now live to 90 or 100, … And so maybe if we lived 200 years, we *could accumulate enough skills* and k*nowledge to solve some* problems. *Marvin Minsky* [13]

Thinking is the hardest work there is. Which is the probable reason why so few engage in it. *H. Ford*

The most, and indeed the truly unique, contribution of management in the 20th century was the *fifty-folds* increase in the productivity of the manual worker in manufacturing. The most important contribution management needs to make in 21th century is similarly to increase the productivity of knowledge work and the *knowledge worker*... One cannot manage change, one can only be ahead of it [1]. *Peter F. Drucker* – the visionary, management guru of guru.[1]

Abstract:

Based on Ernst Muldashev's ophthalmic geometry pattern, not visible for modern human being and visible with suitable resolution by computer, Avtandil Anuashvili's first novel objective 2D personality typology, we present Nikita Moiseev's general idea that stagnant, alive, and social matter (the matters that exist on the Earth) is the same one that passes a process of matter self-organization on the different stages. That is, there is a co-evolution of technical, alive, and social system that takes place with peculiarities of modern human being "problem domain". We propose to create the variant of the new productivity theory of mental worker also. According to P. Drucker, this theory will objectively determine the concrete power of the state into 21th century.

Keywords:

Science objectification and integration stage, E. Muldashev's ophthalmic geometry pattern, A. Anuashvili's 2D personality typology, productivity theory of mental worker

[1] The first two epigraphs call modern problems of humanity that living father of artificial intelligence (AI) M. Minsky formulates in [13]. The third epigraph proves that thinking is the heaviest way of HB existence. The forth one puts the contents of the paper main problem and defines the method of problem's solving for modern biometrics. (The paper must be read with References [1]-[30] because the majority of the ideas are announced in them. The references are in English and Polish.)

1. Problem statement and short introduction to stage of research

1.1. The purpose of the essay-paper, and direction of the Facial Asymmetric Biometrics

One of the seven Peter F. Drucker's challenges of 21st century is discussed in [1]: creation of the productivity theory of mental worker contra the productivity theory of manual worker. The last theory is elaborated basing on principles of Henry Ford and some brilliant programmers (it must include multimedia productivity of modern human being (HB). The last theory has been created during the last 100 years. According to P. Drucker, the creation and usage by the concrete state of the productivity theory of mental workers, which in USA are about 30% population now, will make a concrete state and their economic competing in 21st century. If the analogue of manual worker productive theory is the G8 state club, the concrete state that puts into a practice the mental productivity theory will enter to the 21st c. A new country club called G21st by the authors.

The HB is the most complex system that we know. And the problem is very complicated one. To force the pace of solving it and another vital problems, the authors propose to create the suitable interdisciplinary unit, for example, *International Association of Interdisciplinary Research into Informatics, Psychology, Biology, Sociology, Business, and Medicine* (IAIR IPBSBM). The unit may be created on the base of Department of Theoretical Physics and Computer Sciences of Lodz University, Poland. One person cannot research a complicated problem all alone at the moment.

In other words, the main purpose of the essay-paper is to obtain a modern methodology that allow us to compose in an automatic mode appropriate psychological models of any personality *based on the (x,y)-frontal facial photo projection and on the invisible ophthalmic geometry pattern (OGP)* for the quantitative biometrics decision making [1]-[6] (not eugenics). (Appropriative trait means the following. When, mainly, HB is thinking about issues that requires verbal comprehension/thinking or mathematical/logical operations, the her/his/its eyes are necessarily sent to the right, but when HB requires the visual-spatial/musical tasks or perception of music, rhythmic sounds of nature - to the left; that is not the eugenics manner!)

The object of study is live HB as species (irrespective of sex, skin colour, religion, nationality etc.) with one's own unique conduct. HB functions on syntactical, semantic, pragmatic and social levels. HB is not a machine. A computer of Turing-Neumann's architecture works on syntactical level only. It is a machine but input vector of face traits may be measured with resolution needed by the end-user (usually more precise one). Bilateralness of the brain after/during HB birth has been investigated during researches. But after special research of the bilateralness, no new knowledge has been found.

Using Anuashvili's theory, does not require direct price detectors of electromagnetic waves and the unique technologies for measuring parameters of waves that are produced by usual HB hemispheres. Colour (or white-black) personality photos are used as input signals. Computer calculates/interprets appropriate cognitive-mental personality traits. The examples of thinker type may be: *a)* person is a neutral-sided, right-sided, or left-sided *b)* person possesses one of psychological types based on C. Jung's personality theory using Meyers-Briggs index MBTI (linear model) [16] or D. Keirsey temperament model (non linear model) [17].

The difference between resolution of computer method and usual HB current senses is the possibility of HB research traits by means of simple artefacts built on the basis of bulk-technology (large) contra current micro-miniaturized technology or nanotechnology. Lim-

ited technology (Atom-by-Atom) is invisible for usual HB senses now and then. So, the white race dexterous student of Polish university about 23 years of old (one fits physically and psychological fits) during creation of asymmetric face database is able to hold vertically head state in intervals of $\pm 1^0$ 20' (measurement makes by means of the ***Photoshop***). It means that points of facial elements (corner of eyes, nose, mouth etc.) may be measured in precise way [2] (as the lines and angles).

In other words, the main goal of the essay-paper is to give a reply to the possibility of creation of the mental worker productivity theory and, in positive case, to give requirements to precise face base measurements. All that we know about productivity of mental worker is presented in [1] (the extra examples of US Presidents, ***visualizer*** and ***listener***, are given in it).

The methodology analysis indicates that the correlation would be a significant one: the correlation coefficient is about ±0.15. In authors' opinion, it is an approximate level of sensitivity of usual HB perception (10±0.56 mm or 9.44-10.56% [3, p.14]. The methodology must include: appropriate cognitive-mental traits of concrete personality and may be explicitly measured/interpreted by computer with bigger resolution than HB one. The traits must base on personality's neutral vertical face ***(x-y)***-projection. In papers [8]-[11] (written during 2002-2011), Kompanets and his working team have created essentials for computer using Muldashev's and Anuashvili's phenomena.

1.2. Big picture of modern biometrics, M. Kleiber about science objectification and integration; state of personality theories, and Jungian cognitive-mental traits

Big picture descriptions, myths of biometrics, modern social demands by automatic personality identification occur in modern biometrics. The modern and future stage we may read in [19 p.47]: "If the demand for personal identification applications are so ubiquitous and the conventional means for personal identification are indeed inadequate, why, then, is biometrics technology not as pervasive and widespread as many of us expect to be? … Issues affecting performances include accuracy, cost, integrity, and ease of use by end user".

In [7], the eminent scientist professor president of the Polish Academy of Sciences M. Kleiber writes that "without the knowledge of elementary particles, artificial intelligence, chaos theory, non-equilibrium processes, and the network society to fully understand the modern world is virtually impossible". Kleiber recommends the canon of sixteen books that will help us to grasp reality. Hereby, he confirms that present stage of sciences is in a stage of knowledge integration and objectification for precise and descriptive sciences.

But A. Anuashvili writes in [2] that "frequently, researchers of one school not admit results research of another school. Teachings of different schools contradict one another and are contradict inside one. The teaching not builds by strictly logic, not is systemic one, and not brings to one and only beginning". V. Makarov, president of European Association of Psychotherapy in his book ***Psychotherapy of new century***, Moscow, OPPL 2001 writes that "Each psychotherapy school have own metaphors, create and specific language offers, individual map of psychological space prepares… Each of school strictly holds for own methods. Change of ideas look in seldom case". But HB finds the nature laws that have one mutual beginning.

It means that there exist many models for estimation of HB personality. G. Boeree [23] lists over 30 personality theories, including Big Five theory. We select the cognitive-mental theory based on C. G. Jung attitude; the models [16] and/or [17] are wide spread between people and are robust enough to eugenics effects.

2. Programming (learning) artificial neuron net as before computer

2.1. Strong and weak of AI version, modern psychology stage and description psychological models, peculiarity of modern HB

Electronic model of McCullough-Pitts neuron was constructed in 1943 year. In the beginning of AI, it showed the strong version of AI, or GOFAI, or „reverse engineering", or wired model give analogue of HB brain. In the 70's, the majority of scientists change such attitude to AI: the wired model could not exchange of HB creativeness, intuition. It could not create knowledge/information that does not exist in HB heads or books. The time has shown that wired model could not explain the bilateral asymmetry of HB brain, which R. Sperry discovered in the end of 20^{th} century, unconsciousness and overconsciousness.

Development of current scientific method is as it is. But HB has over 60 billions of live cells that dynamic change in different scales. HB grows up in peculiar environment, influenced by different impacts, has own individual or unique history. Such unrestricted HB must relent by simplification (for example, from over 7 billion on the Earth people define 16 psychological types also). Sciences try to penetrate into complexity of HB through theories' creation and their verification. It causes the great number of different personality conceptions. The factors, which stimulate HB, are: energy or ego strength, instinct, orientation, inheritance, entelechy, experience accumulation, motivation, socialization, cognitivisation, homeostasis etc. If we imagine every possible relation between the main factors, psychological descriptive limitations have to be understood.

Our psychological research uses anthropocentrism principle and Darwin's natural selection principle taking into consideration of HB peek „multimedia possibilities" now [24]. Alive HB remains not too much changed during the last some thousands years (see notion "koinphilia"). That is why the need of personality typology research by scientific methods emerges. The needs are intangible but essential ones.

2.2. The asymmetry problem and fuzzy programming

It is common knowledge that all alive entities including HB, assemble with left-asymmetric molecules [14]. Now, drugs, food from genetic modified objects assemble from symmetric molecules. Because very things are not sure, we use symmetrical molecules now. For example, no one knows how number of asymmetrical organs may change by symmetrical ones in modern medicine now? Or in what way will the forth HB medicine generation be created [3]: in what manner will be created alive entities with symmetrical molecules?

After solving the main problems it a computer of Turing-Neumann's architecture will be possible programmed, the resources will be calculated, the productivity will be computed, and the artificial neuron nets will be learnt. That is, modern HB will be acted in such manner as before the computing discovery. We propose to do such now in case when an environment is so complicated that no mathematical model exists or the mathematical model is severely non-linear one.

Nevertheless, rumination about the non-biological intelligence, super-intelligence, singularity, transcended intelligence, that is, R. Kurzweil's hypothesis about the enhanced HB (epihuman, hyperhuman or parahuman, allohuman), in our opinion, are previous ones. But, we propose to create the main part of the *BrainThinker* advisor [6] in this case.

3. M. Minsky's thinking ways and our selected ideas

The authors use the methods of Minsky way of thinking [12] with T. Kida's slip-ups. To illustrate the paper, we select some ideas:

1. Based on E. Muldashev's phenomenon [4], the authors build the computer visible OGP.
2. Based on A. Anuashvili's theory [2], the authors accept his minimum 2D HB cognitive-mental typology and the biofeedback effect.
3. The authors accept Anuashvili's phase and harmonized portraits. As pattern to next research, we build precise computer phase (left and right and harmonized one). We put forward hypothesis about measuring appropriate degree of asymmetry by algebraic length of line of the noselabial furrow on special Muld-normalized (*x, y*)-projection. (At the moment, the noselabial furrow of HB, probably, appears in uterus, but not as a period of HB maturation).
4. The authors accept principles of anthropocentrism, Darwin's theory of HB life in aggressive environment, unknown dynamic environment changes, lateralization of the right/left half-hemisphere makes in uterus, mainly, and/or during HB maturation (the experiment shows).
5. The authors accept the principles of programming artificial NN (as it was before the discovery of Turing-Neumann's computer) also.

4. Proofs of some modern HB curiosities

4.1. Modern difference between bulk-technology and Atom-by-Atom one

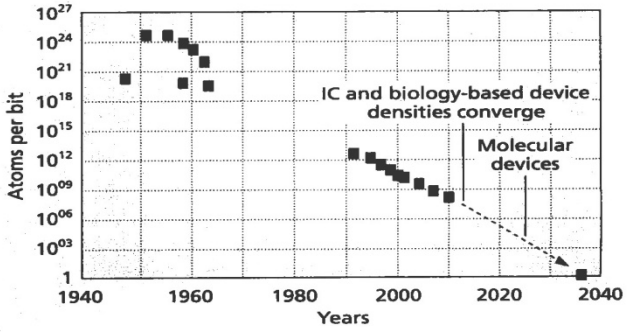


Figure 1. Modern state of a persuading bulk-technology move towards Atom-by-Atom ones (with enthusiastic prognosis) [27]

Modern stage and tendency of objective moving from bulk-technology to Atom-by-Atom limited technology (A-b-A) describes Fig.1 and Table 1, which is the original one. Progress in micro-miniaturization is, but it is still far from the potentially possible. The fact notices: a modern difference between bulk-technology and A-b-A one reaches $10^{22} - 10^{11} =$ ***10^{11} atoms per bit.***

4.2. Design of technical, alive, and social HB system

Table 1. Essential distinction between traits of modern machine/computer system and alive/social HB one

#	Relationship	Manmade machine/ computer system	Alive/Social HB system
1	Input signal	Data	Knowledge (?)
2	AI version	Strong version of AI	Weak version of AI

3	Method of construction	Construction of any technical machine	Self-organization (current state of micro-miniaturization) of matter with intelligence (?)
4	Relationship to a life-cycle	It passes a life-cycle of technical system	From birth to death, system passes through reproduction phase (another phases are auxiliary ones)
5	Where the energy for functioning system is going from	Energy resources to system gives a projecting HB	Open system that during its functioning/rest, which is in any HB cells, have an energy transmitters that execute so called Maslow's psychological needs deficit
6	Design/Fabrication principle	Fabrication on the level manmade bulk-technology with its designed by man functional parties	Self-organization and/or self-assembly based on precise Atom-by-Atom (A-b-A) technology and modern state of micro-miniaturized bulk-technology (MMBT) that aims at A-b-A one
7	Design/Fabrication technology	Manmade bulk-technology (without modern MMBT that not reached of A-b-A one)	MMBT that might be reached by A-b-A one
8	Functioning principle	Modern science/ engineering (technical) functioning	From birth...to death functioning based on Darwin's natural selection principle (with the East meditation methods)
9	Structure of system	Symmetric molecules	Left-asymmetric molecules

4.3 On A. Maslow's needs deficit, scientific method, and 16 typology types

The modern scientific method has the following conventionalities [18, vol.2, p.5]: 1) state-of-the-art of system approximation 2) signal is measurable 3) system signals has a form $exp(st)$ only 4) system has quantitative effect 5) signal with "smallness" of system output effect is not considered 6) linearity, small dimension, and another simplifications of system 7) effect of smoothing signal 8) system is researched in stable mode, when all temporary processes are finished 9) situation prognosis (statistics also) is available one, etc.

For all this, a technical system performs in a more precise way than any biological or social HB system, that is, the technical system is more approximate system. On limited capability of system mode N. Wiener joked: "For incomprehensible cases, mathematicians invent Infinity but people – the God".

In our case, there are 16 types based on the theory of C. Jung types. The type is a hypothetical concept, which refers to behaviour HB traits; it is revealed through subjective questionnaires made by Meyers-Briggs or D. Keirsey. In Mayers-Briggs's type model for description of personality there are introduced 4 coordinates in the space of observation: the first – the dichotomy of extravert (E) - introvert (I), the second one – iNtuitive (N) - sensitive (S); the third one – thinking (T) - feeling (F); and fourth one – deciding (J) - perceiving (P). Cognitive-mental qualities are described by the second coordinate and third one. The Anuashvili's brain model puts psychology knowledge in systemic order and collects in 2D shape (such shape is usual for the management).

5. Muldashev's and Anuashvili's theories: sketch of sketch

5.1. On Muldashev's OGP

About year 2000 E. Muldashev after research states that HB visual system after second year of living works with resolution 10±0.56 mm (9.44-10.56%). It has the same visual scale until the death, but HB reproduction was still sexual [3]. The Kompanets' team uses it in [8]-[11].

5.2. On Anuashvili's theory, and his psycho-correction system

About year 2000 A. Anuashvili creates the first objective 2D personality typology based on HB brain bilateralism and discovers many psychological moments about multimedia possibility of HB. He introduced ***spiritual*** (right), ***vital*** (left), and ***harmonized*** portraits. He manipulated with the lines and angles of the ears, mouth, and noselabial furrow as well as created the system of visual psycho-diagnostics and psycho-correction. Our new idea is using computer possibility of precise (with tolerance that gives of the end-user, usually % or %%) manipulation with portraits and programs of artificial neuron nets – creation of the main part of the ***BrainThinker*** advisor. The advisor gives possibility of better understanding of HB thinking process and works as adaptive devise to phenomenon.

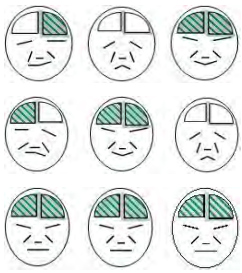


Figure 2. The identification scheme of the left-sided, right-sided, and neutral-sided thinker (we manipulate with the portraits are building by computer in precise way (with resolution of 10' for "truthful" vertical axis)

6. Unsolved problems in modern biometrics

There are two unsolved problems at the moment: F. Crick's problem and problem of precise measurements of input vector of face elements (see our paper ***On Hypothesis Realization of the Neutral-sided, Right-sided, Left-sided Personality, and Creation Necessity of the Precise Generation Face Base, for Modern Biometrics***). Crick's problem is formulated as follows: "How from very big stagnant algorithmic modules comes an alive conscious HB brain" [15].

7. Conclusions

We take into consideration Minsky's opinion "how the mind works - remains inconceivable" today. And the authors propose use of computer for precise extraction of appropriate facial cognitive-mental traits from ***(x, y)***-projection of personality's vertical photo (white-black or colour one). The authors propose the creation of a fuzzy-programming advisor by using neural nets for better acquaintance of process learning. This is one of the way to realize that the W. Wernadsky's (and N. Moiseev's) idea that a special matter self-organization on the Earth or how a stagnant matter transforms in alive one and then in social form.

C7.1 It may be created the productivity theory of mental worker: But for its creation it must be created personality theory that, in any manner, has necessarily classified the HB as the new personalities – the visualizer, listener (as the US Presidents [1]) or toucher. Problem is qualitatively novel and dissimilar. It is much unknown one.

C7.2 HB as entity has some properties: HB limited multimedia possibilities, age, sexual reduplication, different type of memory, dynamic aggressive unpredictable outside environment, lateralization etc. We need to look at HB as at the special "problem domain".

C7.3 It may be created a face base for precise measurements of facial elements, for example, the face database *UniLodz_v.1*.

C7.4 It may be created the main part of the *BrainThinker* advisor. The main part of advisor is created because we have only one example from Anuashvili's presentation (without many expert data).

References

[1] Drucker P.: *Management Challenges for the 21st Century*, Elsevier, Oxford 2007

[2] Anuashvili A. N.: *Principles of objective psychology*, 4 th ed. Corrected and Refilled. Intern. Institute of Control, Psychology and Psychotherapy in Warsaw, Centrum of Human Being Ecology by Institute of Control Problems in Moscow, Moscow, 2005 (In Russian). At: http://anuashvili.ru/images/kniga1.doc (Accessed on Dec. 2011)

[3] Muldashev E.: *Whom Did We Descend From?*, OLMA Press, Moscow, 2002 (In Russian)

[4] Zadeh L. A.: Web site http://www.cs.berkeley.edu/~zadeh/ Also, on the site of Prof. L. A. Zadeh: see… /Publications/ Selected Digitized Paper/ (Accessed on Feb. 2009)

[5] Zadeh L. A.: *From Computing with Number to Computing with Words – from Manipulation of Measurement to Manipulation of Perception*, Intern. Journal of Appl. Math. Comput. Sci 2002, Vol.12, No.3, 307–324

[6] Martin M. A.; Mendel J. M.: *Flirtation: a Very Fuzzy Prospect: a Flirtation Advisor,* At: http://sipi.usc.edu/~mendel 41 pages, (Accessed on Dec. 2011)

[7] Kleiber M.: *Canon scientific books that you must know,* At: http://www.polskatimes.Pl/forum.polska/krakow/55415,michal-kleiber-kanon-ksiazek-naukowych-ktore-nalezy-znac.id,t.html, (Accessed on Feb. 2011)

[8] Kompanets L.: *Facial Composites, Ophthalmic Geometry Pattern, and Based on Stated Phenomena the Test of Person/Personality Aliveness*, Proc. 7th Intern. Conf. on Intelligent Systems Design and Applications", 22-24 October 2007, Rio de Janeiro, Brazil, In: Proc. ISDA'07. Mourelle I; Nedjach N; Kacprzyk J; Abracham A (Eds), IEEE Computer Society Press*, Los Alamitos, CA, USA, 831-836.*

[9] Kompanets L., Kurach D.: *On Facial Frontal Vertical Axes Projections and Area Facial Asymmetry Measure*, In: Computing, Multimedia and Intelligent Techniques, Czestochowa Univ. of Technology Press, 2007, vol. 3, No 1, August, 61-86.

[10] Kompanets L., Kubanek M., Rydzek Sz.: *Czestochowa's precise model of a face based on the facial asymmetry, ophthalmogeometry, and brain asymmetry phenomena: the idea and algorithm sketch*, In: Enchanced Methods in Computer Security, Biometric and Artificial Intelligence Systems. Pejas, J; Piegat, A (Eds). Kluwer Academic Publishers, Springer Science+Business Media Inc. 2005, 239-251.

[11] Milczarski P., Kompanets L., Kurach D.*: An Approach to Brain Thinker Recognition Based on Facial Asymmetry*, Proc. 10th Intern. Conf. on Artificial Intelligence and Soft Computing – ICAISC. Zakopane, Poland, 3-17 June 2010. In: LNAI, # 6113, Rutkowski L; Żurada J; Tadeusiewicz R; Zadeh L (Eds), 436-445.

[12] Minsky M.: *The Emotion Machine: Commonsense Thinking, Artificial Intelligence, and the Future of the Human Mind*, The Emotion Machine 9/6/2006 (book) draft (1 2 3 4 5 6 7 8 9). At: http://web.media.mit.edu/minsky/E1/eb1.html (Accessed on Febr. 2011)

[13] Minsky M.: *Talks on health and the human mind*, At: http://www.ted.com/talks _marvin _minsky _and_ the_human_mind. html (Accessed on Oct. 2011)

[14] Moiseev N.: *Algorythms of development,* Moscow, Science, 1987 (In Russian)

[15] Crick F.: *The Astonishing Hypothesis: The Scientific Search for the Soul.* Charles Scribner's Sons, 1994
[16] Meyers-Briggs: *MBTI @ Personality Type Applications*. At: http://www.Personality pathways.com/typeinventory.html , (Accessed on Dec. 2011)
[17] Keirsey D.: *Keirsey Temperament Web*. At: http://www.kiersey.com/. (Accessed on Dec. 2011)
[18] Siebert W. McC.: Circuits, Signals, and Systems, vol. 1, 2. Moscow, Mir, 1988 (In Russian)
[19] Pankanti Sh., Bolle R. M., Jain, A. K.: *Biometrics: The Future of Identification*, IEEE Computer. Febr. 2000, 46-49
[20] Penrose R.: *The Imperor's New Mind. Concerning Computers, Minds, and the Laws of Physics,* 3th ed. Warsaw, PWN, 2000 (In Polish)
[21] Wang, L.-X., Mendel J. M.: *Generating Fuzzy Rules by Learning from Examples,* In: IEEE Transactions on Systems, Man, and Cybernetics, vol.22 No 6, Nov./Dec. 1982, 1414-1427 At: http://sipi.usc.edu/~mendel (Accessed on Dec. 2011)
[22] Luger G.: *Artificial Intelligence. Structures and Strategies for Complex Problem Solving*, 4th Ed. Publ. House Williams, Moscow - St Petersburg - Kiev, 2005 (In Russian)
[23] Boeree G.: *Personality Theories*, At: http://webspace.ship.edu/cgboer/perscontents.html (Accessed on Jan. 2011)
[24] Busan T., Busan B.: *The Mind Map Book. Radiant Thinking. The Major Evolution in Human Thought*, 2000. BBC Worldwide Lim, London
[25] Dawkins R.: *The God Delusion,* Bantam Press, 2006
[26] Liu Y, Weawer R. K.; Schmidt K., Serban N.; Cohn J.: *Facial asymmetry: A new biometric,* Technical Report CMU-RI-TR-01-23, The Robotics Institute, Carnegie Mellon Univ. 2001
[27] Zhirnov V., Herr D.: *New Frontiers: Self-Assembly and Nanoelectronics*, IEEE Computer 2001, Jan., 34-45
[28] Jain A. K. et al: *Soft Biometric Traits for Personal Recognition Systems,* Proc. the First Intern. Conf. on Biometric Authentication, Hong Kong, July 2004. In: LNCS #3072. Zhang David; Jain, Anil (Eds), 731-739
[29] Kuhn T.: *The Structure of Scientific Revolutions*, Chicago Univ. Press, 1962
[30] Russ J. C.: *The image processing handbook*, 5th ed. Taylor & Francis Group, Boca Raton, London, New York, 2006

On hypothesis realization of the neutral-sided, right-sided, left-sided personality, and creation necessity of the precise generation face base, for modern biometrics

Piotr Milczarski, Leonid Kompanets

University of Lodz, Department of Theoretical Physics and Computer Sciences

Abstract:

In the paper, which is linked with the other work "On asymmetry and fuzziness of suitable cognitive-mental traits of usual personality, possibility of the mental worker productivity theory creation" [18], we develop propositions: on realization of the neutral-sided, right-sided, left-sided personalities the hypothesis for switching path during learning of people, and necessity of the precise description of the face bases for input of suitable a face landmarks vector. Both problems concern with the modern biometrics. During learning of people we prove the possibility of creation of the precise generation a face bases. We formalize the demands to the creation of the face database of novel type, the face base UniLodz_v.1 and build the main part of the BrainThinker advisor also.

Keywords:

Neutral-sided, right-sided, left-sided personalities hypothesis; Demands to the precise of face base; BrainThinker Advisor.

1. Problem statement, short introduction "on realization of hypothesis", and necessity of the face databases precise generation

The paper is connected the previous book paper "On asymmetry and fuzziness of suitable cognitive-mental traits of usual personality, possibility of the mental worker productivity theory creation" [18]. Based on the Anuashvili's first novel 2D objective personality typology and the Muldashev's ophthalmic geometry pattern (OGP), we develop the propositions: on realization of the neutral-sided, right-sided, left-sided personalities hypothesis for switching path during learning people (and not talented ones also), and necessity of the precise generation of face bases for suitable input of precise face landmarks vector, in modern biometrics. During learning of people and proving possibilities of creation of the precise generation of face bases, we formalize demands for the creation of the face base of novel type, the face base *UniLodz_v.1* and build the main part of the *BrainThinker* advisor also.

Hence, the main purpose of the paper is to obtain a modern methodology that allows us to compose in automatic mode appropriate (non-eugenics) models of any personality based on *the (x,y)-frontal facial projection, the OGP* and other phenomena. The main phenomenon are inexpensive building of the right/left- and harmonized portraits of a concrete personality, and use of the OGP invisibility for usual senses of modern human being (HB).

According to modern knowledge, the neutral-sided thinker is more creative or intuitive one (S. Jobs, P. Drucker); the left-sided thinker accepts mental decision logically or digitally (Klitchko-brothers, Barack Obama), and the right-thinker – mental decision accepts in holistic way (Joe Biden). We work out the necessary procedures of realization of the neutral-sided, right-sided, left-sided personalities hypothesis for switching path during (e)-learning

of people and necessity of the precise description of the face bases for input of suitable face landmarks vector. Both problems concern the modern biometrics. In [11, 18], we give necessary information of the modern biometrics.

Apart from this we would like to present short description about the state of art about biometrics. In the sources [1]-[5] there are solved some fundamental problems. In the [6]-[9] there are solved the main and indispensable Anuashvili's and Muldashev's problems using the methods elaborated by Kompanets' working team. In the work [10] it is used one of the usual HB multimedia limitations. In the papers [11], [12] there are illustrated the needs of elaborating the soft classifiers of age, sex, ethnicity, in modern biometrics.

The outcome of [13] shows that traditional measurements (or in the ordinary source of science according to T. Kuhn, "normal science") are not sufficient. Such classifiers do not work because normal science without shift paradigm is a "straitjacket" for scientists thinking. The author in [14] reference starts the research of asymmetry problem in modern biometrics also. In the [15] there is contained the AI manual. One can find in [16] an interesting technical work, which presents curiosities of modern scientific methods. And at last the [17] one is a good handbook about different image processing methods with great understanding of peculiarity of HB vision.

2. Face recognition new horizon in 21th c. by means of precise measurements

In the paper (PhD thesis) written by Esme, Bilgin; Sankur, Bulent: *Effects of Aging over Facial Feature Analysis and Face Recognition* [11], it is said about unsatisfactory approach of the traditional face elements measurements. For example, the coordinates of the nose tip, silhouette, and other choice elements depend on the light, pose, etc., that is, traditional measurements are statistical (with errors) ones. The situation is the indeterminate one: in (cosmetics) art and biometrics are gaps concerning HB visual resolution (9.44-10.56%) [3,18]. In art the asymmetry traits of HB less than 1 mm may put by maquillage; in recognition, if we have a small photo, the object may not be recognized with required probability in principle. There is an indeterminate situation about the answer to the question "woman-man" at the moment. (On this idea and the precise measurement of personality brain parts, we work monochromatic biometrics algorithm of finding the answer to the question "woman-man".)

We must to find the way to reduce the errors to the necessary level. The authors' opinions about precise measurements are the following: *a*) we must select the suitable elements and explicitly define it for different personality race *b*) in the precise face databases, the values of landmarks *must be added c*) based on the *photo size*, in photographic devices there is attached the "*recognition*" mode that gives a possibility to recognize face features and personality with big probability (about 99.9). (Sections "Technical specs" and "So, in conclusion..." about HB vision are given at pp.83-133 in [17]). *Precise element measurements may give a face recognition new horizon in 21th century.*

Manually, use of clicking mode only by the end-user is enough to do precise measurements of face element (for example, for the eye corners, mouth corners, etc.). In Fig.1 and 2, we interpret essence of the traditional and precise styles in biometrics measurements. Corners of eyes, wings, top of a nose, features of brows, etc. for the white/black/yellow race are not defined precisely during the passage from 4D of HB object to 2D-vertical (x,y)-projection. Therefore, the result of measurements may be imprecise or subjective.

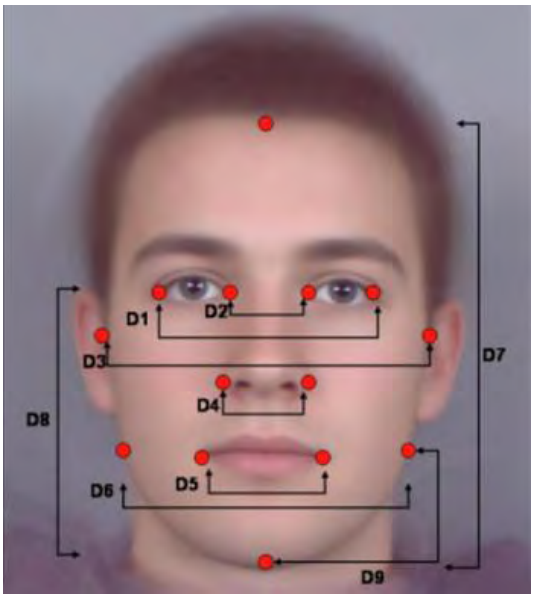


Figure 1. The example of HB points obtained in traditional measurements (the problem of measurements with non equivalence about 3-7 pixels) by clicking during post-procedures (the image size, different type of compression, computer format, etc.). The picture from Internet, after using the Photoshop effects; the "photoshoped" colour photo of symmetrical man works usually. What do we measure if element points depend of the light, sex, age, pose, race and measure in non-precise way?

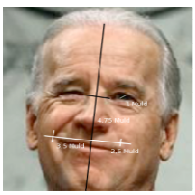
Figure 2. Example of precise measurement and computer processing of Kompanets' great-grandson Orest colour photo in about seven month old, (in the experiment of lateralization, Orest gives uniform right-sided lateralization; Orest's mother is left-handed, and father is right-handed one; his noselabial furrow forms, probably, in uterus). For precise measurements we must define explicitly what are the right and left corners of mouth for people of different race, the right and left corners of eyes, etc. Remembering this we draw, for example, a circle for explicit defining of touch point of the corner eyes/mouth and then we can protect of defining synonymy during any photo transformations

3. Hypothesis of on precise measuring of features with two examples

Hypothesis: To define the neutral-, left-, right-sided thinker in art and biometrics, at the beginning, we estimate thinker to our classification and finish the process of measurements visually. (On Fig.3, the woman is the left-sided thinker and man is a right-sided one.) Then, we find the "truthful" vertical (*x, y*)-projection axis and along in the distance of 4.9 below it Muld we build the perpendicular line to the "truthful" axis that is normalized by maximal length of the noselabial furrow (in both directions). Along the chosen axis we derive the algebraic value of the measure, $\pm\Delta As$. The thinker type is defined by means of the *Brain-Thinker* advisor. Sign "+" means the right-sided thinker, but sign "-" – the left-sided thinker in normalized fuzzy-intervals. During building the advisor it is used for prompt appropriated and suitable fuzzy- intervals given by end-user instead of expert collective.

According to A. Anuashvili [2, 18], the usual people are about 50% of the population now; the usual=normal +accentuated are the ones, in which hemisphere oscillations are fewer than 4 times.

Figure 3. Visual finding of the left-sided – woman and right-sided – man (the photo from Internet and represents of two Brazil TV speakers); according to the part 1 of our hypothesis



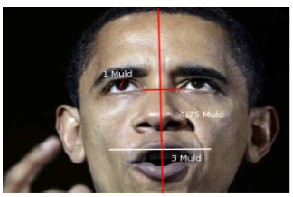


Figure 4-6. Vertical photos of Peter Drucker, Joe Biden, and Barack Obama (the photos are from the Internet) that demonstrate need of precise measurements. (Those were not done in precise way because P. Drucker was the old man (over ninety years of old) and has the glasses, J. Biden has the short nose, and B. Obama has the short nose, not see straight ahead, has small visible right noselabial furrow, but in traditional way those are use)

4. Pseudocode of Kurach-Kompanets' algorithm of facial asymmetry measures calculating

In the Fig.7 [9, 18], there are presented Kurach-Kompanets' new facial asymmetry measures, $\pm\Delta As$, AAs^{+}, AAs^{-} of appropriate facial regions of HB: vertical (*x,y*)-projection of photo or silhouette. The values AAs^{+}, AAs^{-} are calculated as the left and right parts of the right-sided and left-sided facial asymmetry measures, which defines by the vertical projection axis of the facial binary images (BI). The values, $\pm\ \Delta As$, represent the algebraic meas-

ure of difference of the Muld normalized facial image projection in relation to "truthful" axis (see the hypothesis). Sign "+" represents- domination of right hemisphere activity and "-" – left one respectively. The idea of finding "truthful" axis is: on the (x,y)-projection finds the axis with resolution about 10' by means of adaptive rotation around the anthropologic point, P_0, - middle point of the inside corners, that distributes the face projection on the right/left part with minimum noise, which provides lighting, pose variation, facial hair, etc.

The *AsymmetryMeasure* algorithm (pseudocode) gives us the way to calculate the values of the asymmetry measures, $\pm \Delta As$, AAs^+, AAs^- in adaptive way by a computer.

Input data: P_0 – precise anthropologic point (x, y-coordinates), φ[°] - rotation angle, (remind for explicitness finding of the corner of eyes, we draws a circle at race personality), BI - silhouette's binary image, *Dr* – the right eye iris diameter. Data in the algorithm are input in manually or precise way.

Output data: $\pm \Delta As$, AAs^+ and AAs^- values for *BrainThinker* and advisor decision: the neutral-sided, right-sided, left-sided personality.

Step 1. Define the "truthful" vertical axis, on a binary facial silhouette's image based on coordinates point P_0, and the value of angle φ[°] (see BI projections)

Step 2. Create a vertically mirrored binary image MBI based on the information about BI image and the axis (see MBI projection)

Step 3. Calculate AAs^+, AAs^- values as the difference between the right and left sided of BI and MBI projections.

Step 4. Use the $\pm \Delta As$, AAs^+ and AAs^- values to solve the switching problem.

We believe that the computed asymmetry type by *BrainThinker* corresponds to the brain thinker type and a person is the neutral-, right- or left-sided thinker.

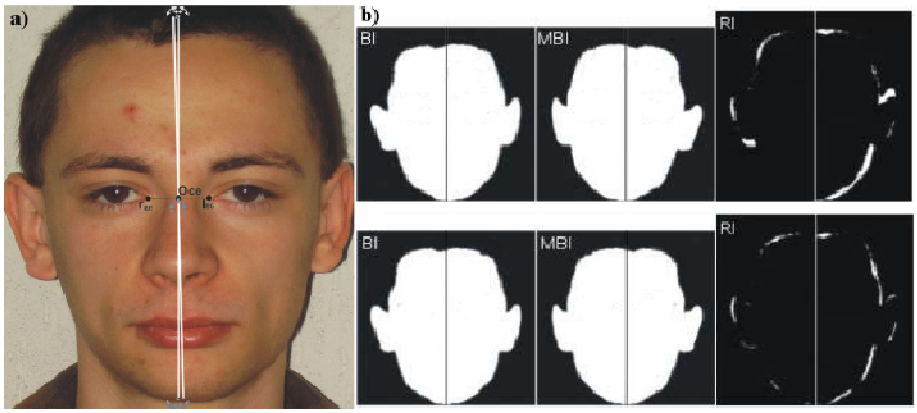


Figure 7. Graphic illustration of finding the facial asymmetry: a) input image with two candidate axis (or more ones); b) BI binary images, MBI mirrored binary images, and RI result images; begin axis - 0° axis and "truthful" axis - 1,5° axis

5. On demands of the face database *UniLodz_v.1*

The main aim of this part is checking weather A. Anuashvili: correctly selected his elements of phase portraits. And in this part of the paper, we give requirements to prepare the precise face base *LodzUni_v.1*. That will give us the possibility of the precise measurements of the input landmarks face vector (with end-user resolution, usually more precise).

Demands to the precise photos

The required RGB-photo is the colour one with 8x8x8 bits. From such photos, it is easy to derive the monochromatic photo with resolution of 8 bits. The pictures must be the Muld-normalized, [16], using the dimension of the right eye and vertical/horizontal compensation. The face expression is a neutral one. The head position must be vertical and HB must look straight ahead in the lens' direction. The resolution of the face photo must be at least 640x480 and no more than 1920x1080. A distance (camera lens – face) is less than 5 m. An appropriate vertical axis y of (x, y)-projection of the picture must be equalized. Illumination: daylight CIE D65 or fluorescent lamps. Background must have uniform illumination (white or blue). The photos must be received in sessions with remoteness of minimum two weeks and at least two times. The Muld-normalization photo on the right eye must work; the (x, y)-coordinates must be fixed after the procedure.

6. The main part of the *BrainThinker* advisor

Fuzzy logic is an important tool for two main reasons. First of all, it allows us to form a model of what is going on when decisions are made. Having a model it can help us to understand more about our decision making processes and learn how to make the best possible decisions. Secondly, we can use this model to design computer that can actually make fuzzy decisions [6, 18, 19]. Let us remember that fuzzy logic is used in case when a model situation is complicated. The advisor has no thinking model. Therefore, in the *BrainThinker* advisor, we sketch the main part of the advisor under conditions that the data examples lack an unknown HB model thinking. We may match fuzzy-intervals as in [6, 17,18], mainly.

In three-level crisp decision making model, we would have to decide between "thinking one is the left-sided", "thinking one is the right-sided", or "thinking one is the natural-sided" type. In fuzzy decision making we provide a degree of similarity to thinking, ranging from zero to ten in symmetrical scales. Then, we give the results in the linguistic label. The rules are established cooperatively by expert groups.

7. Conclusion and propositions

CP7.1 During learning about personality, it may be realized the switching of the path of learning for suitable educated model and worked out the main part of the *BrainThinker* advisor.

CP7.2 Usual HB as entity has some multimedia properties and we can formalize such HB by precise measurements of face landmarks and also face recognition with solving soft-problems such as the sex, age, ethnicity [12, 18], etc. It may be a new horizon in 21^{th} c. in the modern biometrics.

References

[1] Anuashvili A. N.: *Principles of objective psychology*. 4th ed. Corrected and Refilled. Intern. Institute of Control, Psychology and Psychotherapy in Warsaw, Centrum of Human Being Ecology by Institute of Control Problems in Moscow, Moscow, 2005 (In Russian). At: http://anuashvili.ru/images/kniga1.doc (Accessed on December 2011).

[2] Muldashev E.: *Whom Did We Descend From?* OLMA Press, Moscow, 2002 (In Russian)

[3] Zadeh L. A: From Computing with Number to Computing with Words – from Manipulation of Measurement to Manipulation of Perception. *Intern. Journal of Appl. Math.*

Comput. Sci 2002, Vol.12, No.3, 307–324. At: http://www.cs.berkeley.edu/~zadeh/papers/ (Accessed on Febr. 2011).

[4] Martin M. A, Mendel J. M.: Flirtation: a Very Fuzzy Prospect: a Flirtation Advisor, At: http://sipi.usc.edu/~mendel, 41 pages, (Accessed on Dec. 2011).

[5] Wang L.-X., Mendel J. M.: Generating Fuzzy Rules by Learning from Examples. In: *IEEE Transactions on Systems, Man, and Cybernetics*, vol.22 No 6, Nov./Dec. 1982, 1414-1427 At: http://sipi.usc.edu/~mendel, (Accessed on Dec. 2011).

[6] Kompanets L.: Facial Composites, Ophthalmic Geometry Pattern, and Based on Stated Phenomena the Test of Person/Personality Aliveness, Proc. 7th Intern. Conf. on *Intelligent Systems Design and Applications*, 22-24 October 2007, Rio de Janeiro, Brasil, In: Proc. ISDA'07. Mourelle I; Nedjach N; Kacprzyk J; Abracham A (Eds), IEEE Computer Society Press, Los Alamitos, CA, USA, 831-836.

[7] Kompanets L., Kurach D.: On Facial Frontal Vertical Axes Projections and Area Facial Asymmetry Measure, In: *Computing, Multimedia and Intelligent Techniques*, Czestochowa Univ. of Technology Press, 2007, vol. 3, No 1, Aug. 61-86.

[8] Kompanets L., Kubanek M., Rydzek Sz.: Czestochowa's precise model of a face based on the facial asymmetry, ophthalmogeometry, and brain asymmetry phenomena: the idea and algorithm sketch, In: *Enchanced Methods in Computer Security, Biometric and Artificial Intelligence Systems*. Pejas, J; Piegat, A (Eds). Kluwer Academic Publishers, Springer Science+Business Media Inc. 2005, 239-251

[9] Milczarski P., Kompanets L., Kurach D.: An Approach to Brain Thinker Recognition Based on Facial Asymmetry, Proc. 10th Intern. Conf. on *Artificial Intelligence and Soft Computing – ICAISC*. Zakopane, Poland, 3-17 June 2010. In: LNAI, # 6113, Rutkowski L; Żurada J; Tadeusiewicz R; Zadeh L (Eds), 436-445.

[10] Busan T., Busan B.: *The Mind Map Book. Radiant Thinking. The Major Evolution in Human Thought*, Revised and updated in 2000. BBC Worldwide Lim. London

[11] Esme, Bilgin; Sankur, Bulent: Effects of Aging over Facial Feature Analysis and Face Recognition, March 2010 (unpublished paper), At:bilgin.esme.org/Portals/0/PhD/EffectsOfAgingOverFacialFeatureAnalysisAndFaceRecognition.pdf.

[12] Jain A. K. et al: Soft Biometric Traits for Personal Recognition Systems, Proc. the *First Intern. Conf. on Biometric Authentication*, Hong Kong, China, July 2004. In: LNCS #3072. Zhang David; Jain, Anil (Eds), 731-738

[13] Kuhn T.: *The Structure of Scientific Revolutions*. Chicago Univ. Press, 1962.

[14] Liu Y., Weawer R. K, Schmidt K., Serban N., Cohn J.: Facial asymmetry: A new biometric, Technical Report Carnegie Mellon Un-RI-TR-01-23, 2001.

[15] Luger G.: Artificial Intelligence. Structures and Strategies for Complex Problem Solving. 4th Ed. Publ. House Williams, Moscow-St Petersburg-Kiev, 2005 (In Russian).

[16] Siebert W. McC.: *Circuits, Signals, and Systems*. vol. 1, 2. Transl. from English, Moscow, Mir, 1988 (In Russian).

[17] Russ J. C.: *The Image Processing. Handbook*, 5fh ed. CRC Press. Taylor and Francis Group, Boca Raton, (2007).

[18] Milczarski P., Kompanets L.: On asymmetry and fuzziness of suitable cognitive-mental traits of usual personality, possibility of the mental worker productivity theory creation, *in preprint*.

[19] Wang L.-X., Mendel J. M.: *Generating Fuzzy Rules by Learning from Examples,* In: IEEE Transactions on Systems, Man, and Cybernetics, vol.22 No 6, Nov./Dec. 1982, 1414-1427 At: http://sipi.usc.edu/~mendel (Accessed on Dec. 2011)

Computing with words in communication processes at sea

Zbigniew Pietrzykowski, Paweł Banaś, Anna Wójcik

Maritime University of Szczecin

Abstract:

Navigational systems introduced on sea-going ships increasingly use modern information-communication technologies. The basic aim of these systems is to provide safe navigation by supporting the operative ship handling by the navigator. Marine accidents, however, still happen in most cases due to human errors. One of such errors is a lack of communication and co-ordination of actions between navigators steering their ships in vicinity of each other. A solution may be achieved by developing principles of automatic communication and co-operation, based on the standard marine communication phrases. As navigators communicate in a natural language, its formalization requires a mathematical form. To this end the authors propose computing with words. The article introduces a concept that extends the functionalities of automatic communication systems with negotiations. It also analyzes some typical dialogs that take place in the process of communication between navigators steering their ships in an encounter situation. A representation of such dialogs using computing with words approach is proposed and conclusions are set forth.

Keywords:

e-marine, e-navigation, computing with words, ontology, automatic communication

1. Introduction

Assurance of safe navigation, thus the safety of personnel, cargo, ships and the environment, requires, apart from access to information and capability of its automatic interpretation, communication aimed at determining or verifying the interpretation of a current and predicted situation, as well as intentions of all the vessel traffic participants concerned.

In maritime shipping, principles of communication between navigators handling their ships and coastal stations are formulated in proper regulations [1, 2, 3]. There are a number of modes of communication, strictly connected with their priority:

- distress communication,
- urgency communication,
- safety communication,
- non-priority communication.

Although the relevant regulations impose definite obligations on vessel traffic participants, they do not eliminate possibilities of dangerous situations that may develop from failing to communicate or errors in communication. These errors may include choosing improper means of communication, incorrect information, misunderstanding or incorrect interpretation of exchanged information. That is why increasingly more attention is paid to problems of information exchange and the automation of co-operation processes [4, 5, 6] taking place between ships [7, 8].

There are a number of shipboard devices and systems for automated acquisition and information exchange, such as AIS, GMDSS, ARPA, Navtex, GPS. However, none of these provides for exchange of messages that would allow to solve situations that are ambiguous from the viewpoint of binding regulations or so complex that that co-operation is required between the ships (navigators) concerned [8].

Rule 2a included in Part A of the International Collision Regulations (COLREGs) is a legal basis for the navigator in ship-to-ship communication, in order to, e.g. settle the manner and time of passing the other ship [1]. The rule does not indicate directly the need to communicate, it only obliges the parties involved to take relevant safety measures, for instance by radiotelephone in the VHF band, operating up to about thirty nautical miles. VHF communication is used in the mentioned earlier routine communication between ships. Unfortunately, RT VHF communication in sea practice often remains careless, incorrect, sometimes just neglected. Examples of such irregularities are often stated in maritime court decisions, where in many cases lack of or improper communication is indicated as one of indirect causes of a marine accident.

One of the methods for solving communication problems may be a set of principles of automatic communication and co-operation, based on developed standards of navigational information, using formal records of messages between navigators within the frames of established standards. Guidelines for ship-to-ship communication are:

- recommendations and legal regulations for communication between ships (GMDSS, AIS, Standard Marine Communication Phrases SMCP [2, 3]),
- factors resulting from sea practices,
- set of data that is (should be) exchanged in the communication process.

To formalize the communication between navigators, carried out in a natural language, the authors propose computing with words. This method allows to extend the possibility of formulating and interpreting messages [7, 8, 9] with non-crisp phrases, as normally used by human beings, e.g. close, closer, safe, hardly safe etc..

2. Computing with words

Interpersonal communication in a natural language is characterized to a large extent by perception. Thanks to it in dialogs both precise (crisp) ad imprecise (non-crisp) phrases are used with no significant effect on understanding [10]. However, when it is essential to convey and exchange information precisely, the message has to be accurately worded to avoid misinterpretation, and consequently, wrong, sometimes dangerous decision. Computer systems may be helpful in situational assessment, but to make their participation in dialogs possible, communication has to be formalized. Precise phrases have values attributed to them, from e.g. measurements. These values become instantly adopted and transformed by computer systems. Imprecise phrases are not measurable.

The first step towards the automation of communication is a mathematical description of imprecise phrases, such as 'safe', 'not far' or 'high'. Each of these linguistic values can be written down by means of a fuzzy set described by a membership function. Some of the phrases may additionally be determined by a function of probability [11] of qualifying a given word to classes of phrases. For instance, the word 'risky' may, with a certain probability, be assigned to these words: 'dangerous', 'not very safe' and 'safe', whose example membership functions are shown in Figure 1.

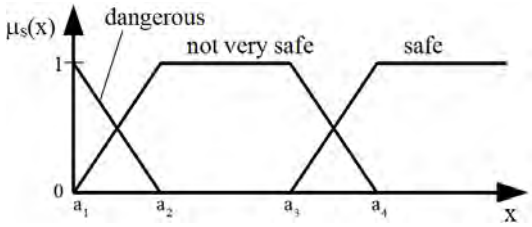


Figure 1. Example functions of membership for words 'dangerous', 'not very safe' and 'safe'

Regarding the word 'risky' as a supplement to the word 'safe', we can define the membership function as:

$$\mu_{NS} = 1-\mu_S \tag{1}$$

where: μ_{NS} is a function of membership of the word 'risky', and μ_S is a function of membership of the word 'safe'. The membership function of the word 'risky' is presented in Figure 2.

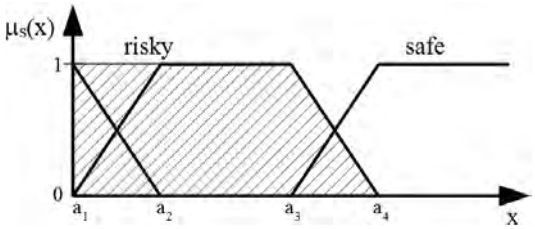


Figure 2. Example membership function for the word 'risky'

Thus we have a scope of words that limit it, e.g. for *distance* we have a group of three basic words and a few words derived from that basis. Figure 1 depicts a trapezoidal membership function. For the words 'dangerous' and 'safe' the function (μ_A), respectively, has a left-hand and right-hand edge. The general formula of this function is given below, while for the words 'not very safe' it is a case of the trapezoid function (μ_B) defined by the formula also given below:

$$\mu_A(x)=\begin{cases} 1 & x \le a_1 \\ \frac{a_2-x}{a_2-a_1} & a_1 < x < a_2 \\ 0 & a_2 \le x \le a_3 \\ \frac{x-a_3}{a_4-a_3} & a_3 < x < a_4 \\ 1 & x \ge a_4 \end{cases} \tag{2}$$

$$\mu_B(x)=\begin{cases} 0 & x \le a_1 \\ \frac{x-a_1}{a_2-a_1} & a_1 < x < a_2 \\ 1 & a_2 \le x \le a_3 \\ \frac{a_4-x}{a_4-a_3} & a_3 < x < a_4 \\ 0 & x \ge a_4 \end{cases} \tag{3}$$

The next step is to define rules for mathematical description of the transmitted message. The basic record of a message (p) has this form:

$$p \rightarrow X \text{ is } R \tag{4}$$

where: X is a variable, is – sentence-creating functional, R – relation limiting the variable X.

In the case of more extensive communication, it is convenient to additionally define a wider range of the variable origin in the form of a function and to write the message record in this form:

$$p \rightarrow A(X) \text{ is } R \tag{5}$$

where: A is a group in the subontology to which X belongs.

Let us take into consideration the statement:

Ship Gamma is a sailing ship.

It can be written as:

$$A(B, C) \text{ is } D \tag{6}$$

where: A – type, B – ship, C – Gamma, D – sailing ship

Considering this statement:

Ship Gamma is at a safe distance to the shore.

we can write it down in this form:

$$E(C, F) \text{ is } G(H, \mu_S) \tag{7}$$

where: E – position, C – Gamma, F – shore, G – safe, H – distance, and μ_S is the membership function for the word 'safe'.

In a natural language we have to do with various forms of sentences – affirmative or interrogative. For precise interpretation of message content we will add functions defining the sentence form. The whole message record will be the argument of that function. In this way, without significant changes of the message record, we will get multiple meanings of schematic records depending on the notations, and interpretation will not be negatively affected.

3. An example dialog

One of the ways of message interchange is the use of radiotelephone VHF communication. The basic element of radiotelephone dialogs between objects such as ships, coast stations, etc., is a single message. Each message consists of at least one sentence. In practice sentences are usually simple ones and contain one piece of navigational information, e.g. for ship encounter situation. Let us consider a dialog between ships Alpha and Beta [8], sailing on opposite courses, as depicted in Figure 3. An example dialog between navigators can be exchanged by voice or using computer systems, or between computer systems automatically messages in a form understood by people.

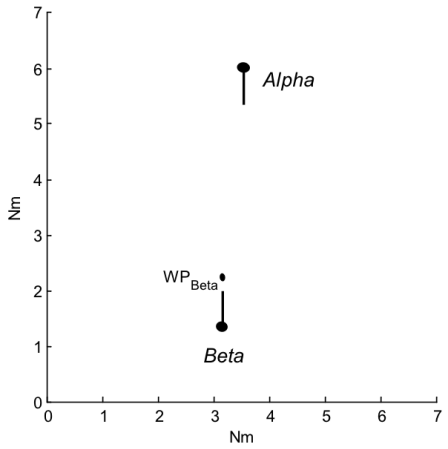


Figure 3. An encounter situation of ships Alpha and Beta [8]

According to regulations in force, the ships should pass each other on their port sides. Although starboard-to-starboard passing is possible, it has to be agreed on by the two ships.

It is possible to use systems that automatically interpret or generate messages at various levels:

- interpretation of information obtained from the navigator only (transferred by voice or through appropriate forms) and from systems carried on own ship,
- interpretation of own information and automatic acquisition of information from the other ship,
- automatic recognition of the situation, determination of possible solutions, automatic negotiation of an optimal solution with the other ship and its presentation to the navigator for approval.

Due to the complex character of problems relating to voice communication (recognition and interpretation of voice messages), it seems recommendable to solve in the first place problems of interpreting text messages containing phrases that are used by the navigator and have a fixed structure within the defined ontology.

The following exchange takes place between the ships:

1. Alpha: 'Our CPA is close to 0.'
2. Alpha: 'Is it possible that we pass starboard to starboard?'
3. Beta: 'I intend to alter my course to starboard soon and cross ahead of you at a safe distance.'
4. Alpha: 'OK, cross ahead of me at a distance of more than 1.2 Nm.'

Once the presented situation is recognized, there are two potential solutions:

- complying with the COLREGs recommendations, where the trajectory does not need to be agreed on,
- one with a much shorter trajectory that, however, has to be agreed on between the ships concerned.

The above dialog is an example of negotiations that are aimed at solving a collision situation by choosing a shorter trajectory.

The presented dialog includes information that can be used in computer systems, but due to the natural language of the dialog, the information is not readily accessible to computers. By applying the adequate subontology [9, 12] and mathematical tools, we can identify useful information and interpret it. A fragment of the ontology used is presented in Figure 4.

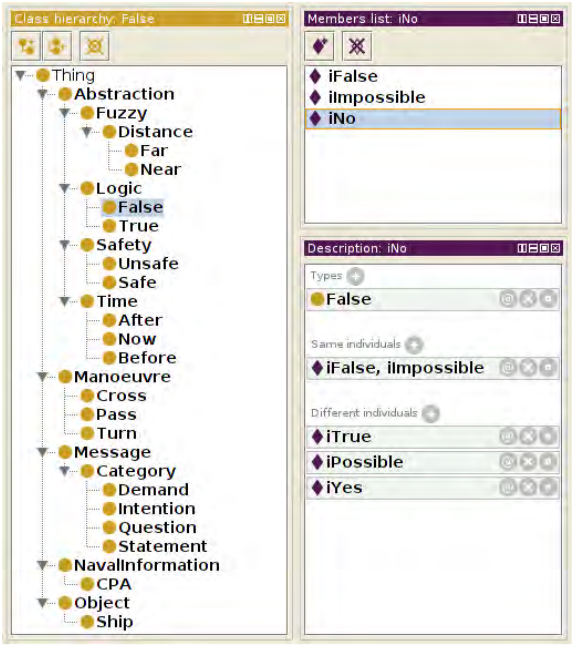


Figure 4. Part of the ontology of navigational communication

Using the above subontology and the mechanism of computing with words, we can present the dialog in this form. The stages of transformation have been indicated:

Message 1:
'Our CPA is close to 0' →
WARNING(CPA(Alpha, Beta) is NEAR(0, μ_D)) →
W(C(A, B) is N(0, μ_D))

Message 2:
'Is it possible that we pass starboard to starboard?' →
QUESTION(PASS(Alpha.starboard, Beta.starboard) is possible) →
Q(K(A.R, B.R) is F)

Message 3:
'I intend to alter my course to starboard soon and cross ahead of you at a safe distance. ' →
INTENTION((TIME(TURN(B, R)) is NEAR(now, μ_T)) and (CROSS(B, A) and (DISTANCE(A, B) is SAFE))) →
I((Z(T(B, R)) is N(G, μ_T)) and (E(B, A) and (D(A, B) is S)))

Message 4:
'OK, cross ahead of me at a distance of more than 1.2Nm.' →
DEMAND(CROSS(B, A) and ((MORE(DISTANCE(A, B), 1.2Nm) is TRUE)) →
F(E(B, A) and (M(D(A, B), 1.2NM) is P))

The subontology of navigational communication for the example shown:

Types of messages:

- question (Q)
- intention (I)
- demand (F)

- warning (W)

Types of navigational information:

- CPA (C) – closest point of approach
- left-hand side, port side, to port (L)
- right-hand side, starboard side, to starboard (R)

Types of manoeuvres:

- passing (K), attached to L and R
- course alteration (T), attached to L and R
- crossing course (E)

Abstract terms:

- near, about (N) – using fuzzy functions μ_D and μ_T that define the membership of an element to a fuzzy set
- more, above, over (M)
- possible (F)
- safe (S)
- now, present time (G)
- true (P)

Objects:

- ship, appears in two instances (A, B), attached to L and R.

4. Summary

One of the causes of marine accidents is lack of proper communication and coordination of actions between navigators handling their ships. This issue can be solved by developing principles of automatic communication and coordination, based on established standards of navigational information. To formalize the messages exchanged by navigators of two ships in a natural language, the authors have proposed computing with words.

Solutions proposed so far required the use of precise values, not permitting the use of non-crisp statements as used by humans while negotiating. Terms such as 'risky', 'closer' or 'faster' could not be used because their precise value could not be determined in measures used during computations. By using the right ontology we can precisely define the meaning of such terms. Computing with words together with fuzzy functions of membership allows to define the values of parameters related to the term considered. The determination of their values depends on contexts in which these terms were used. These mathematical tools may also be employed for generating messages in a form comprehensible for the human being, where precise values are replaced by approximate terms dependent on the context.

To implement automatic communication in the processes of negotiations one has to develop basic dialog scenarios first, then solve problems of speech recognition and interpretation. At present scenarios for the mentioned processes are being developed.

References

[1] Rymarz W.: *A handbook of Collision regulations* (in Polish), Trademar, Gdynia 1995.
[2] International Maritime Organization (IMO), Resolution A.918(22) adopted 29.11.2001, IMO Assembly 22-nd Session 25.01.2002.
[3] International Maritime Organization (IMO): *Standard Marine Communication Phrases (English-Polish edition)*, Maritime University of Szczecin, Szczecin, 2005.

[4] Beam C., Segev A., 1997, *Automated Negotiations: A Survey of the State of the Art*, Wirtschaftsinformatik vol. 39:263-268, 1997.
[5] Paurobally S., Turner P. J. and Jennings N. R.: *Automating negotiation for m-services*, Proc. of the IEEE Transactions on Systems, Man, and Cybernetics (Part A: Systems and Humans), Special issue on M-services. 33(6):709-724, 2003.
[6] Karp A. H.: *Rules of Engagement for Automated Negotiation*, Proc. of the First IEEE International Workshop on Electronic Contracting (WEC'04):32-39, San Diego, USA, 2004.
[7] Pietrzykowski Z., Chomski J., Magaj J., Niemczyk G.: *Exchange and Interpretation of Messages in Ships Communication and Cooperation System*, Advanced in Transport Systems Telematics, pages 313-320, ed. J. Mikulski, Publisher Jacek Skalmierski Computer Studio, Katowice 2006.
[8] Pietrzykowski Z., Hołowiński G., Magaj J., Chomski J.: *Automation of Message Interchange Process in Maritime Transport*, International Recent Issues about ECDIS, e-Navigation and Safety at Sea – Marine Navigation and Safety of Sea Transportation, ed. A. Weintrit, Gdynia 2011.
[9] Pechmann P.: *Reprezentacja znaczenia komunikatu w języku polskim za pomocą ontologii* (in Polish), Metody Informatyki Stosowanej, nr 1/2011, Gdańsk 2011.
[10] Zadeh L. A.: *From Computing with Numbers to Computing with Words - From Manipulation of Measurements to Manipulation of Perceptions*, IEEE Transactions on Circuits and Systems vol. 45:105-119, 1999.
[11] Bronsztejn I. N., Siemiendiajew K. A., Musiol G., Mühlig H.: *Nowoczesne kompendium matematyki* (in Polish), pages 393-394, PWN 2009.
[12] Kopacz Z., Morgaś W., Urbański J.: *Information on Maritime Navigation; Its Kinds, Components and Use*, European Journal of Navigation, Vol.2, Nr.3:53-60, Aug 2004.

Security management in database servers

Anna Plichta[1], Szymon Szomiński[2]

[1]*Cracow University of Technology, Faculty of Physics, Mathematics and Computer Science*
[2]*AGH University of Science and Technology,*
Faculty of Electrical Engineering, Automatics, Computer Science and Electronics

Abstract:

The main part of contemporary IT systems are databases. They stores data and provides access to them. When we try to define the concept of database, we do not define how the data is organized in database's structure. We can distinguish few types of databases: simple (card-index type and hierarchical) and complex (relational, object, relational and object mixed, stream and temporal). Data security issues in database systems concerns all of the types mentioned above, no matter on the data organization method. The most common database type nowadays is relational. Data security doesn't mean only to protect them, it is more complex and it contains also authorization mechanisms and user management. Another aspect is something we can call "secure storage" – data encryption and backup policies. When we need to make the database system secure, we need to define how the data is valuable. We need to think about factors which can have influence on the data integrity, we should plan also how to avoid them or how to minimize the risk connected with them. Hacker's attacks, software bugs or hardware faults can make the data inaccessible or destroyed.

Keywords:

authentication, encryption, data recovery

1. Introduction – definition of security

The notion of security, in regard to bases of knowledge, is hard to define, because it primarily pertains to computers and Information Technology (IT) systems. In fact, when it comes to computer systems, so many areas can be protected, that one can barely create a definition of security embracing all of them. In the field of bases of knowledge it could be described as a state in which risk level is sufficiently low thanks to the implementation and continual monitoring of the security policy and procedures. The notion of "tolerable risk level" is essential in this context, as is can be measured and achieved by means of making up rules to abide by.

For instance, when there is a SQL Server instance, we should not only secure the database engine, but we need also to think about the Local Area Network (LAN) security and client stations security. When the database is accessible from the Internet, we need to know, what threats are waiting for our system. Simply, there is no single definition for all aspects of the database security.

We know the environment in which our base of knowledge system works and we can estimate the risk. Therefore, we can ask why exactly we should protect the databases and which mechanism to apply to it. These are data which are ought to be protected in bases of knowledge, simply because of their value. Access to data should be granted only authorized users after signing on to the system. How can we define the user authorization in the base of knowledge systems and how does it work?

2. User authentication

Authentication is a process of verification of declared identity of the particular user, device or service involved in data exchange. Authentication ensues identification (in other words, it follows declaration of identity stated by the user who tries to sign on to the system), so it is the declared and not the verified identity, that is checked by means of entering password.

There are two ways of authentication. The first one is called Windows mode because it requires the agency of the Windows system and only the particular users and groups of trusted accounts are granted access to the base of knowledge server. This mode is widely used in computer applications, in NT LAN (NTML) or in Kerberos. Windows accounts uses a set of encrypted messages to authenticate user in database management system. During this process, no passwords are transmitted.[1]

The second one is called mixed mode, because the user is obliged to have two accounts: both the Windows user account and the account of the base of knowledge server. When we used a mixed-mode authentication, it is less secure, because user's password needs to be sent by network. The advantage of using Windows authentication is a big simplification of access to data source – the authentication process is done only once.[1]

Due to the fact, that in such method password is mutually transferred, one should primarily stick to the Windows mode.

Principles governing user and password management are among the principles governing the whole operation system. Groups of users and passwords within the operation system are subdued to IT system administrators whereas administrators of base of knowledge are authorized to grant users various levels of access to the stored data. When we start to provide access to database, we call this process authorization. It is a next part of database security management process. One of the most significant elements is Identity Management System (IMS). IMS stores information about accounts, their permissions and methods of connecting. Identity of users can be confirmed using certificates, biometrical data or tokens.

What should we take account of choosing between various database user authorization policies?

3. Letters of authenticity

There are many modes of authentication used while signing on to the protected system. Choosing one of them we should take into account whether it is onerous and troublesome for the system users, because too complicated solutions can generate higher costs of maintenance and management. It should provide redundancy, enable for integration with other systems (e.g., Active Directory) and support intelligent cards, biometric data (and similar identity proofs). In advanced system real data required for authentication are either sent encrypted or their position in the stream of data is not obvious. Many systems make use of data hashing instead of transmitting real certificates of authentication. Such process is a perfect encrypting key password hashing means that the password is converted by one-way function to a constant-sized hash.

It cannot be reversed and thus it prevents potential hacker from recovering the original password from the intercepted data transmission. Among others, Windows uses hashing to store passwords on its domain.

Choosing the proper authorization policy and preventing any potential intruder from gaining and recovering password is crucial, but the question remains how to safely store logins, passwords or certificates.

4. Storing data required for authentication

Certificates, user name and password (or hashed password) must be stored in some place, which is safe and convenient. It should resist attacks (e.g., brute force attacks) but the authorized user should still access the data quickly enough to avoid anxiety. It is common to store certificates in the central base (called certification server) and to use Lightweight Directory Access Protocol (LDAP) as an data accessing tool which uses catalogue method. The above-mentioned data can comprise many objects, but it is typical of it to embrace also users' authenticity data.[1]

Having at our disposal some mechanisms which enable us for storing the authorization data in the secure and quite comfortable manner – as the above-mentioned LDAP – we should consider how to tighten the security of the data stored at the hard disc.

5. Security of the data while storing

It is essential to store physical files of the base of knowledge securely. The secure file storing system should guarantee the authorized user immediate access to all necessary data. It is so in New Technology File System (NTFS) which has protected inner discs and outer driving systems. Storage Area Networks (SAN), networks of mass memories, are able to provide the desired security, too.

Encryption is a process which converts data from plain-text to non-eligible version, called 'cryptogram'. A special encryption algorithm should be defined in order to proper converting our data. The algorithm uses a secret key, which should be used also in decryption process.[7] Encryption in database systems is used to protect data saved in storages. Encryption can be implemented also in application which uses the database. The database can be encrypted also with algorithms provided by operating systems, such as BitLocker or Encrypting File System (EFS).[9]

BitLocker uses an Advanced Encryption Standard (AES) algorithm with 128 or 256-bit key length. Each sector of disk partition is encrypted on the lowest, hardware level, so it makes the encryption mechanism transparent for system and application. Independently of AES, a diffuser mechanism designed by Microsoft, is used. It can be disabled when we do not want to force users using not certified cryptographic algorithms.

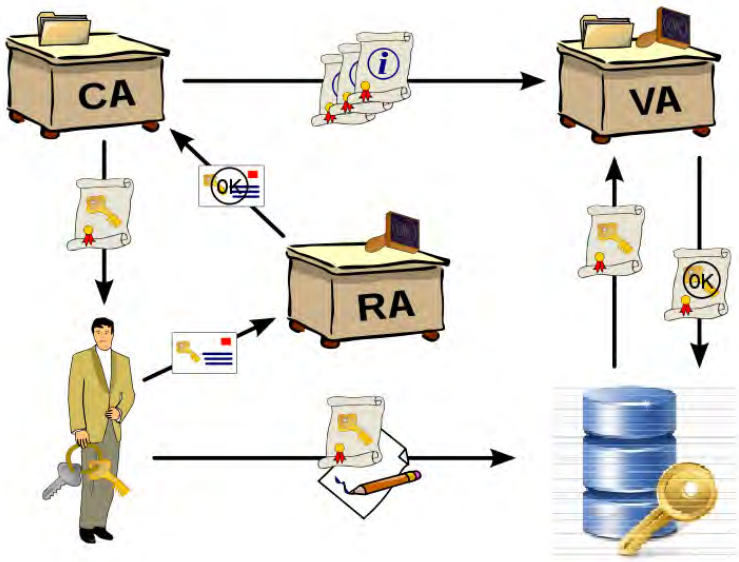


Figure 1. Public-Key Infrastructure [11]

EFS file system is implemented since NTFS 3.0. The encryption is realized on file level and it provides protection for confidentiality against physical attacks on the computer system. By default, EFS is available since Windows 2000, but it is disabled. We can encrypt single folders, single files or whole disk. Some settings can be managed from domain level or group policy mechanisms.

Independently of these mechanisms, a special security for encryption keys should be provided. And here comes the Public Key Infrastructure (PKI).

In the Fig.1. it is shown the diagram of the user authorization by means of public key architecture. The user identity is authorized by the CA office (Certificate Authority) which gives certificate that the identity is true. The identity must be unique in each CA domain. Every user declaring his will to obtain identity certificate must be verified: he either brings his papers to the Registration Authority or is verified electronically, which is much faster. The subscribing individual must have clear identity. The certificate authenticity is verified by the third unit called VA (Validation Authority). If the user want to get any data from the base of knowledge, he sends a request with the certificate which the system passes to VA which verifies it and grants access to the data or not. [2][3][4]

In database systems the data encryption can be provided on several levels: operation system (OS), database management system and the database. The PKI can be used parallel.

Typically, the base of knowledge systems gather or store information which is useful for some users or other database systems. Therefore, one should provide the safe data transfer between clients making use of our data supplies.

6. Security of the data while transfer

In order to protect data while transfer between base of knowledge server and a client, we should take into account, that a client can use other operation system then we do; end client can be also a server to whom we will grant access to data. Hence, we ought to establish encrypted connection between two access points, which is absolute must while any wireless transmission. Although it is all little bit harder to intercept data if the transmission is not wireless, encrypting proves once again advisable, because it reduces hazard of data theft to minimum.[1][8]

Despite so many security levels sometimes we are compelled to restore database system which is damaged – due to the successful attack, flaws and errors in our software or unexpected hardware failure.

7. Data recovery and backing up a database

In contrast to typical OS backup, which can be made only when there was any modifications made, the database should be backed up regularly. Frequency of backup depends on data importance and frequency of modification. An effective backup strategy should provide information, which data should be archived, how and when the backup should be done and how we can roll back when a fault would occur.

Data recovery is also essential for the security of base of knowledge data. There are some criminals who intend not to steal the data, but to make them inaccessible (or to destroy them). As a result, we should keep the encrypted backup data in an isolated and well protected environment. The main cause for creating backups is to provide a possibility of recovery data in case of any faults or attacks and in result to provide continuity for our system.[1]

8. Conclusions

Providing security and continuity of database system work is a multilevel process. It should contain physical security, access to database, encryption, network and OS security. Each database require its own definition and customization of security level. Protection mechanisms should be adequate to possible threats, we always need to find a happy medium between security technology and costs of security management.

References

[1] Carpenter T. *Microsoft SQL Server 2008 Administration – Real-World Skills for MCITP Certification and Beyond*. Wiley Publishing 2010.
[2] Kenan K. *Cryptography in the Database: The Last Line of Defense*, PWN, Warszawa 2007.
[3] Ferguson N. *Practical Cryptography*, Helion, Gliwice 2004.
[4] Ogiela M. *Podstawy kryptografii*, Wydawnictwa AGH, Kraków 2000.
[5] Buchmann J. *Wprowadzenie do kryptografii*, Wydawnictwo Naukowe PWN, Warszawa 2006
[6] Pomykała J. M. *Systemy informacyjne: Modelowanie i wybrane techniki kryptograficzne*, MIKOM, Warszawa 1999.
[7] *Database Security Issues*, *http://databases.about.com/od/security/Database_Security_Issues.htm* (11.04.2012)
[8] *Top Ten Database Security Threats*, *http://www.schell.com/top_ten_database_threats.pdf* (10.04.2012)
[9] *Cryptography in SQL Server*, *http://technet.microsoft.com/en-us/library/cc837966(v=sql.100).aspx* (28.03.2012)
[10] *Bezpieczeństwo baz danych*, *http://www.securitum.pl/baza-wiedzy/publikacje/bezpieczenstwo-bazy-danych* (20.03.2012)
[11] Public-Key-Infrastructure, *http://pl.wikipedia.org/w/index.php?title=Plik:Public-Key-Infrastructure.svg&filetimestamp=20080407092644* (21.03.2012)

Nonlinear model of a straight line

Andrzej Piegat

West Pomeranian University of Technology, Szczecin
Faculty of Comupter Science and Information Technology

Abstract:

A straight line seems to be the symbol of linearity. Scientific books on the analytical geometry present only linear mathematical models of a straight line in space. This paper shows that linear models of a straight line dominate only in the 1D- and 2D-space. Instead, in the 3D- and higher spaces also non-linear models can be applied. They define a line with only one equation, whereas linear models consist of a number of separate equations. Non-linear models of a straight line can be used in many applications, e.g. in constructing non-regular fuzzy models [1].

Keywords:

straight line, straight-line model, straight-line equation, analytic geometry

1. Introduction – classic models of a straight line in 2D-space

A straight line is one of basic and, it would seem, good defined objects in the analytical geometry [2, 3]. Its mathematical models can be found without difficulties in handbooks and mathematical encyclopedia, e.g. in [4, 5, 6], in the Internet, e.g. in Wikipedia [7]. All these sources give identical linear models of a straight line. Further on 3 mostly popular linear models of a line will be presented. In 2D-space a line can be described by one general equation (1), Fig. 1.

$$a_2 x_2 + a_1 x_1 + a_0 = 0 \tag{1}$$

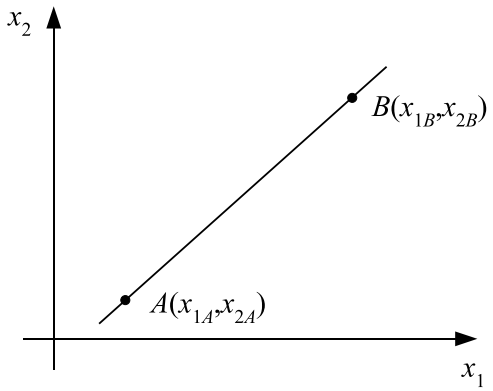


Figure 1. A straight line determined by two points A and B with known coordinate values in 2D-space.

Coefficients a_1 and a_2 simultaneously can't be equal to zero. Coefficients a_1, a_2, a_0 can be identified if coordinate values of two points A and B which lie on the line are known. The second popular model of a line in 2D-space is the slope-intercept equation (2).

$$x_2 = b_1 x_1 + b_0 \tag{2}$$

Coefficient b_1 means the slope or gradient of a line and bo the intercept with the x_2-coordinate. Coefficients of the model (2) can easily be identified on the basis of the points A and B. An interesting model is the parametric model of a line, Fig. 2.

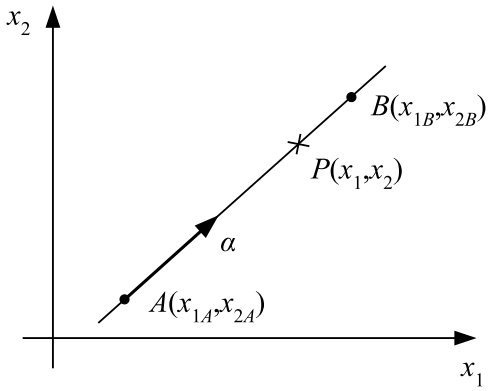


Figure 2. Illustration of the parametric line-model. A and B are known points.

The parametric line-model is not a one- but a multi-equation model. In case of 2D-space it consists of 2 separate components (3).

$$\begin{aligned} x_1 &= x_{1A} + \alpha(x_{1B} - x_{1A}) \\ x_2 &= x_{2A} + \alpha(x_{2B} - x_{2A}) \end{aligned} \tag{3}$$

A great advantage of the parametric model is the parameter α. The value $\alpha > 0$ informs that the point under consideration $P(x_1, x_2)$ lies above the point A, the value $\alpha < 0$ means that the point P lies beneath A. The absolute α-value is a distance measure of the point P from A. It also should be noted that nor from the general model (1) or from the slope-intercept model (2) an information can be achieved about position of the considered point P in relation to the reference point A. Models (1) and (2) can only inform us whether the considered point P lies or doesn't lie on a line. In case of 2D-space calculation of the calibrated distance-parameter α is simple. For any non-generated line at least one of differences $(x_{1B} - x_{1A})$ or $(x_{2B} - x_{2A})$ is different from zero. If e.g. the difference $(x_{1B} - x_{1A}) \neq 0$ then the parameter α can be calculated from formula (4).

$$\alpha = \frac{x_1 - x_{1A}}{x_{1B} - x_{1A}} \tag{4}$$

The parametric model (3) can easily be transformed into the general model (1) or the slope-intercept model (2) by eliminating α from equations (3). Testing whether the considered point $P(x_1, x_2)$ lies on a line is with the parametric model (3) more labour-consuming than with non-parametric models (1) and (2). In case of the parametric model one should separately calculate two values α_1 and α_2 from both component equations (3). If these values are equal then the point P lies on a line. If $\alpha_1 \neq \alpha_2$ then the point does not lie on a line. Modeling a straight line in 3D- and higher spaces becomes more complicated.

2. Classical, parametric, linear-multiple-equation model of a straight line in the 3D-space (P-L-MEq-3D-model)

As the previously cited references inform 2 basic types of a line model in 3D-space can be applied: the parametric, linear-multiple-equation model and the edge-model. The edge model will be presented in Section 4. Fig. 3 explains some notations connected with the P-L-MEq-3D model.

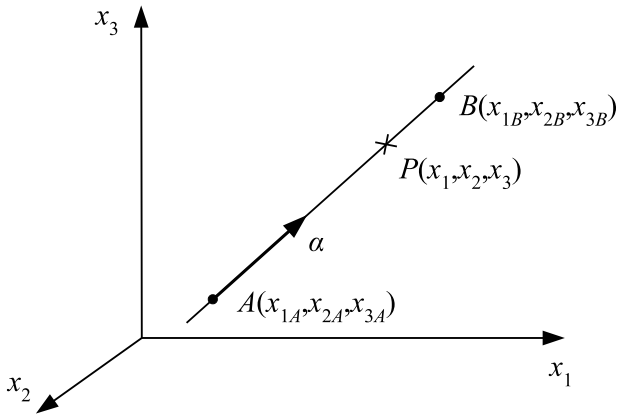


Figure 3. Illustration of notations connected with the parametric, linear-multi-equation model of a straight line in the 3D-space. Points A and B are known. P – any point on the line.

In the 3D-space the P-L-MEq-3D model is expressed by (5).

$$\begin{aligned} x_1 &= x_{1A} + \alpha(x_{1B} - x_{1A}) \\ x_2 &= x_{2A} + \alpha(x_{2B} - x_{2A}) \\ x_3 &= x_{3A} + \alpha(x_{3B} - x_{3A}) \end{aligned} \tag{5}$$

Equations (5) can be expressed in a shorter form (6).

$$x_i = x_{iA} + \alpha(x_{iB} - x_{iA}), \quad i = 1, \ldots, 3 \tag{6}$$

To check whether the point under consideration $P(x_1, x_2, x_3)$ lies on a line its coordinates are to be inserted in equations (5) and 3 parameter values α_1, α_2, and α_3 are to be calculated from each of the equations. Only if $\alpha_1 = \alpha_2 = \alpha_3$ then the point P lies on the line. It should also be noted that model (5) is not a one-equation model. It is a set of 3 linear component models describing one geometrical object: a line in 3D-space. It also should be noted that in the analytical geometry rather compact (one-equation) models are preferred by scientists. E.g. a circle with the centre in the origin of the coordinate system, of radius R, Fig. 4, is mostly modelled by only one, single equation (7) though it also can be modelled by 2 parametric equations (8).

$$x_1^2 + x_2^2 = R^2 \tag{7}$$

$$\begin{aligned} x_1 &= x_{10} + R\cos(\alpha) \\ x_2 &= x_{20} + R\sin(\alpha) \end{aligned} \tag{8}$$

Further on a parametric, nonlinear, one-equation model of a straight line in 3D-space will be presented.

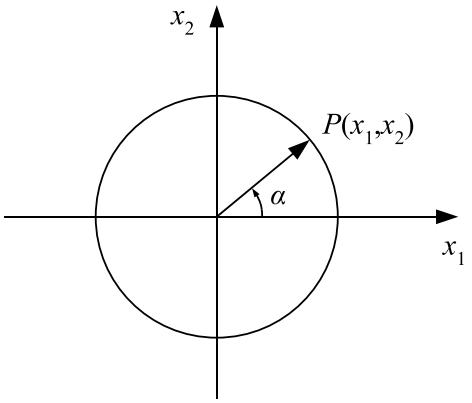


Figure 4. A circle of radius R with the centre in the origin of the coordinate system, $x_{10} = 0$, $x_{20} = 0$.

3. Parametric, nonlinear, one-equation model of a straight line in 3D-space

This model will be constructed with use of *the method of logical aggregation of conditions* conceived by the author. Below a short description of the method. If a considered mathematical object has to satisfy m conditions, then the conditions should be transformed in the normalized form (9).

$$W_i = 0\,, \quad i = 1, \ldots, m \tag{9}$$

Next, all conditions should be aggregated in one, single condition which globally defines the object under consideration. Aggregation of the component conditions can be realized conjunctively, disjunctively, or in a mixed way dependently on the problem specificity. A good example of the method application will be aggregation of conditions concerning a straight line. In case of the parametric line-modeling in the 3D-space, Fig. 3, its classical, linear, multi-equation model consists of 3 conditions given by (10).

$$\begin{aligned} x_1 &= x_{1A} + \alpha(x_{1B} - x_{1A}) \\ x_2 &= x_{2A} + \alpha(x_{2B} - x_{2A}) \\ x_3 &= x_{3A} + \alpha(x_{3B} - x_{3A}) \end{aligned} \tag{10}$$

According to the method of logical aggregation of conditions equations (10) are transformed in the normalized form $W_i = 0$, (11).

$$\begin{aligned} (x_1 - x_{1A}) - \alpha(x_{1B} - x_{1A}) &= 0 \\ (x_2 - x_{2A}) - \alpha(x_{2B} - x_{2A}) &= 0 \\ (x_3 - x_{3A}) - \alpha(x_{3B} - x_{3A}) &= 0 \end{aligned} \tag{11}$$

A convincing proof that a test-point $P(x_1, x_2, x_3)$ lies on the line is the simultaneous satisfying (conjunction) all three conditions W_i for this point at one and common value of the parameter $\alpha_1 = \alpha_2 = \alpha_3 = \alpha$. All three conditions (11) in the conjunctively aggregated form are given by (12).

$$\begin{aligned} [(x_1 - x_{1A}) - \alpha(x_{1B} - x_{1A})]^2 + [(x_2 - x_{2A}) - \alpha(x_{2B} - x_{2A})]^2 + \\ + [(x_3 - x_{3A}) - \alpha(x_{3B} - x_{3A})]^2 = 0 \end{aligned} \tag{12}$$

In equation (12) the particular conditions W_i were squared to prevent their mutual reduction in case when one condition W_i has a negative value and the second a positive one. The squared conditions (12) can be transformed in a more compact form (13).

$$\sum_{i=1}^{3}[(x_i - x_{iA}) - \alpha(x_{iB} - x_{iA})]^2 = 0 \tag{13}$$

After squaring the components of equation (13) the form (14) was achieved, which enables calculation of the α-value corresponding to the considered point $P(x_1, x_2, x_3)$.

$$\alpha^2 \sum_{i=1}^{3}(x_{iB} - x_{iA})^2 - 2\alpha \sum_{i=1}^{3}(x_i - x_{iA})(x_{iB} - x_{iA}) + \sum_{i=1}^{3}(x_i - x_{iA})^2 = 0 \tag{14}$$

Equation (14) can be solved for the point P in respect of α with use of the known Vieta-formulas. If the point $P(x_1, x_2, x_3)$ lies on a line then only one real root $\alpha_1 = \alpha_2 = \alpha$ of equation (14) is achieved. If the point P does not lie on the line two different compound numbers $\alpha_1 \neq \alpha_2$ are achieved. Equation (14) can easily be adopted to the space on any n-dimension. The version for the n-dimension is given by (15).

$$\alpha^2 \sum_{i=1}^{n}(x_{iB} - x_{iA})^2 - 2\alpha \sum_{i=1}^{n}(x_i - x_{iA})(x_{iB} - x_{iA}) + \sum_{i=1}^{n}(x_i - x_{iA})^2 = 0 \tag{15}$$

Further on 2 examples will be presented.

Example 1.

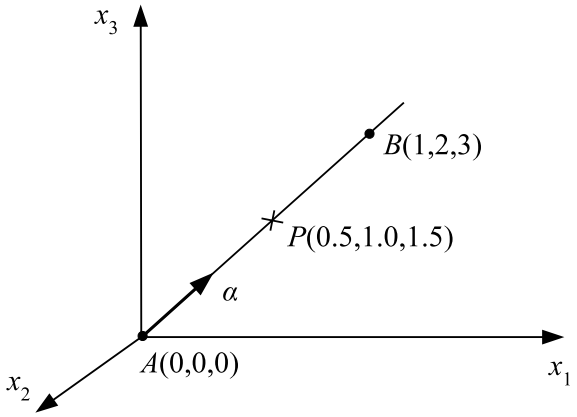


Figure 5. Illustration to Example 1. The test point P lies on the line AB.

In Example 1 the test point $P(0.5, 1.0, 1.5)$ lies on the line AB in the middle between points A and B. After inserting coordinate values of the points A, B and P in equation (14) equation (16) was achieved.

$$14\alpha^2 - 14\alpha + 3.5 = 0 \tag{16}$$

Solving this equation delivers only one root $\alpha = 0.5$. This result is in agreement with position of the point P on the line. Parameter Δ in Vieta-formulas equals zero.

Example 2.

It should be tested in Example 2 whether point $P(1,1,1)$ lies on the line AB. After inserting coordinate values of points A, B and P into the non-linear line-model (14) equation (17) was achieved.

$$14\alpha^2 - 12\alpha + 3 = 0 \tag{17}$$

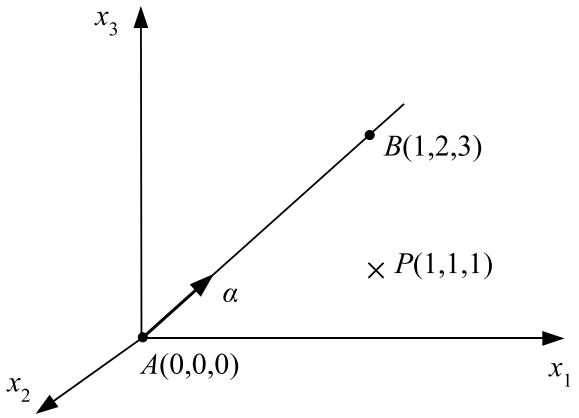


Figure 6. Illustration to Example 2. The test point P does not lie on the line AB.

This time parameter Δ in Vieta-formulas does not equal zero ($\Delta = -24$) and 2 compound roots are achieved.

$$\alpha_1 = \frac{3}{7} + \frac{i\sqrt{6}}{7}, \quad \alpha_2 = \frac{3}{7} - \frac{i\sqrt{6}}{7}$$

The compound roots α_1 and α_2 inform us that the test point P does not lie on the line AB.

Further on the classical, linear, edge-model of a line in 3D-space will be presented.

4. Classical, linear, multi-equation, edge-model of a straight line in 3D-space

The classic, linear, multi-equation edge-model (L-ME-E-model) described in previously cited references is based on perception that a straight line in 3D-space can be defined as the intersection edge of two planes on which simultaneously the line lies, Fig. 7.

A single plane in 3D-space can be described with one linear equation, thus a line as the intersection edge of two planes has to satisfy two conditions given by (18).

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + a_{13}x_3 &= D_1 \\ a_{21}x_1 + a_{22}x_2 + a_{23}x_3 &= D_2 \end{aligned} \tag{18}$$

Point $P(x_1, x_2, x_3)$ lies on the line AB only when its coordinates simultaneously satisfy the two conditions given by (18). If a line exists in space of dimension n then the line, as the intersection-edge of $(n-1)$ planes has to satisfy $(n-1)$ conditions and the line-model consists of $(n-1)$ linear component-models (19).

$$\begin{aligned} a_{11}x_1 \quad &+ a_{12}x_2 \quad + \ldots + a_{1n}x_n \quad &= \quad D_1 \\ a_{21}x_1 \quad &+ a_{22}x_2 \quad + \ldots + a_{2n}x_n \quad &= \quad D_2 \\ \vdots \quad & \qquad\qquad\qquad\qquad\quad &\vdots \\ a_{(n-1)1}x_1 &+ a_{(n-1)2}x_2 + \ldots + a_{(n-1)n}x_n &= \quad D_(n-1) \end{aligned} \tag{19}$$

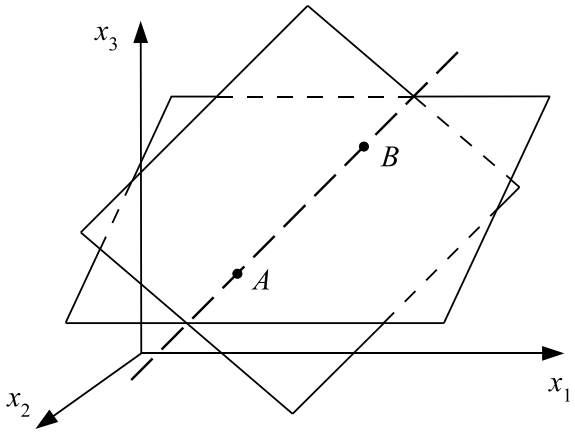


Figure 7. Illustration to the edge-version of a line-definition in 3D-space where a line is interpreted as intersection of two planes.

To check whether a test-point $P(x_1, x_2, \ldots, x_n)$ lies on a line $(n-1)$ conditions given by (19) have to be checked.

5. Non-linear, one-equation, edge-model of a straight line in 3D-space (N-1Eq-E-3D model)

Any line in 3D-space can be presented as the intersection edge of an infinite pair-number of planes. However, the simplest way for derivation of the line model is using one plane that is vertical to the coordinate plane $X_1 \times X_3$ and second plane that is vertical to $X_1 \times X_2$, Fig. 8.

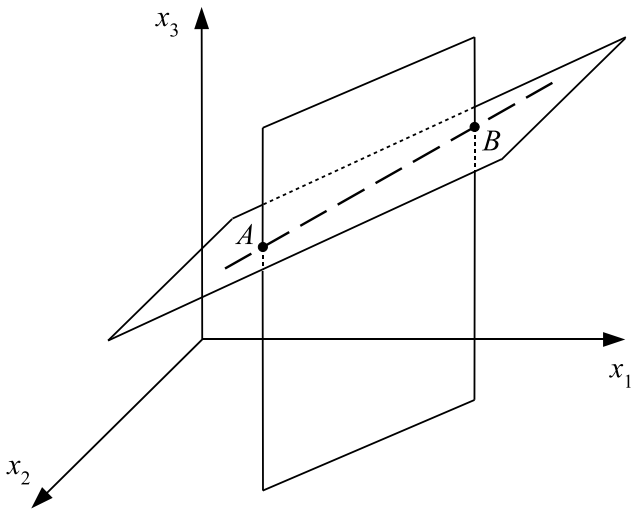


Figure 8. Intersection of two planes defining the line AB: one vertical to $X_1 \times X_3$ and second vertical to the $X_1 \times X_2$ coordinate-plane.

Fig. 9 presents projection of the plane which is vertical to $X_1 \times X_2$. The plane vertical to $X_1 \times X_2$ can be expressed by equation (20).

$$a_1 x_1 + a_2 x_2 + a_3 = 0\,, \quad x_3 : \text{ any}\,, \tag{20}$$

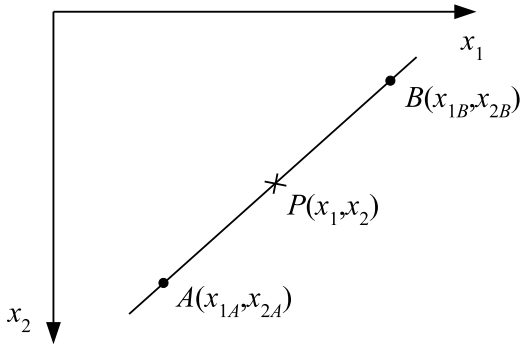


Figure 9. Projection of the plane co-defining the line AB on the plane $X_1 \times X_2$ of the coordinate system.

where:

$$\begin{aligned} a_1 &= (x_{2B} - x_{2A}) \\ a_2 &= -(x_{1B} - x_{1A}) \\ a_3 &= x_{2A}(x_{1B} - x_{1A}) - x_{1A}(x_{2B} - x_{2A}) \end{aligned}$$

Projection of the plane vertical to $X_1 \times X_3$ is presented in Fig. 10. The plane vertical to $X_1 \times X_3$ can be modeled with formula (21).

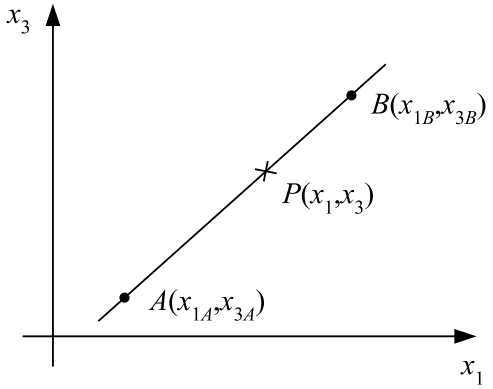


Figure 10. Projection of the plane vertical to $X_1 \times X_3$, co-defining the line AB, on this plane.

$$b_1 x_1 + b_2 x_3 + b_3 = 0\,, \quad x_2 : \text{ any}\,, \tag{21}$$

where:

$$\begin{aligned} b_1 &= (x_{3B} - x_{3A}) \\ b_2 &= -(x_{1B} - x_{1A}) \\ b_3 &= x_{3A}(x_{1B} - x_{1A}) - x_{1A}(x_{3B} - x_{3A}) \end{aligned}$$

Equations (20) and (21) are conditions which have simultaneously (conjunction) be satisfied by point $P(x_1, x_2, x_3)$ to lie on the line AB.

$$\begin{aligned} W_1 : \quad a_1 x_1 + a_2 x_2 + a_3 &= 0 \\ W_2 : \quad b_1 x_1 + b_2 x_3 + b_3 &= 0 \end{aligned} \tag{22}$$

To achieve a one-equation model of a line the both conditions (22) have to be conjunctively aggregated. To prevent their mutual reduction in case when one condition would be of negative and the second positive their squares will be aggregated (23).

$$(a_1x_1 + a_2x_2 + a_3)^2 + (b_1x_1 + b_2x_3 + b_3)^2 = 0 \tag{23}$$

After squaring both expressions in formula (23) the edge-model of a line (24) was achieved.

$$(a_1^2 + b_1^2)x_1^2 + a_2^2x_2^2 + b_2^2x_3^2 + 2a_1a_2x_1x_2 + 2b_1b_2x_1x_3 + \\ + 2(a_1a_3 + b_1b_3)x_1 + 2a_2a_3x_2 + 2b_2b_3x_3 + (a_3^2 + b_3^2) = 0 \tag{24}$$

The edge-equation of a line can be expressed also in a more general form (25).

$$C_1x_1^2 + C_2x_2^2 + C_3x_3^2 + C_4x_1x_2 + C_5x_1x_3 + C_6x_1 + C_7x_2 + C_8x_3 + C_9 = 0 \tag{25}$$

The model (25) seemingly contains 9 coefficients. However, because part of the coefficients has to be different from zero the model can always be transformed to equation with 8 or less coefficients.

In case when the edge-model of a line has been derived not on the bases of projections on ($X_1 \times X_2$ and $X_1 \times X_3$) but on coordinate planes ($X_1 \times X_2$ and $X_2 \times X_3$) in equation (25) instead of the component $C_5x_1x_3$ the component $C_5x_2x_3$ will occur. For spaces higher then 3D the edge model of a line can be derived on the basis of $(n-1)$ conditions of type (22) and (23) which represent $(n-1)$D-spaces. The line-equation is for higher spaces also of second order (square one), only number of components increases in comparison with equation (25). Further on 2 examples of application of the non-linear edge-model will be shown.

Example 3.

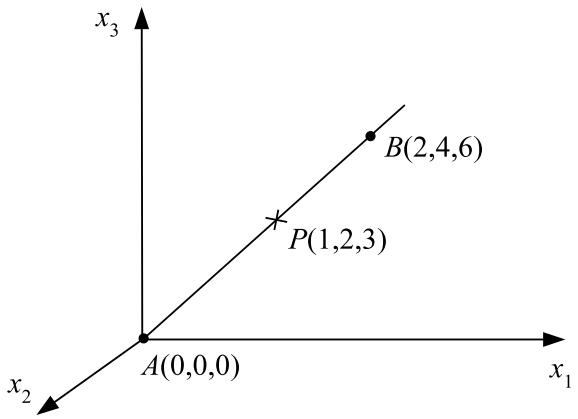


Figure 11. Illustration to Example 3. The test-point $P(1, 2, 3)$ lies on the line AB.

It should be checked on the basis of the edge-model whether the test-point $P(1, 2, 3)$ lies on the line AB. After inserting coordinates of points A and B in equations (20) and (21) the non-linear edge-model (26) was achieved.

$$13x_1^2 + x_2^2 + x_3^2 - 4x_1x_2 - 6x_1x_3 = 0 \tag{26}$$

Inserting coordinates of the test-point $P(1, 2, 3)$ in equation (26) gives zeroing of the left side of this equation $(13 + 4 + 9 - 8 - 18 = 0)$. This result confirms position of the point P on the line AB.

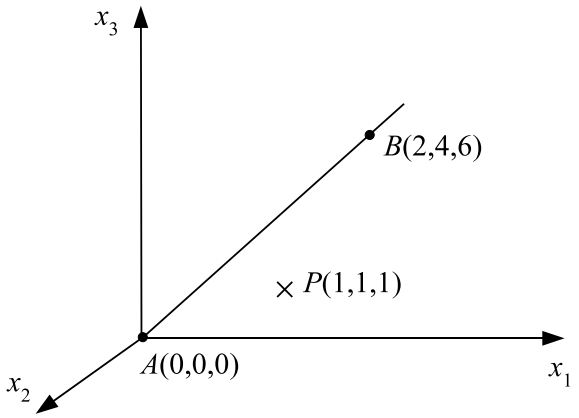


Figure 12. Illustration to Example 4. The test-point $P(1,1,1)$ does not lie on the line AB.

Example 4.

It should be checked with use of the nonlinear edge-model whether the point $P(1,1,1)$ lies on the line AB. The model of this line was determined in Example 3 and is given by equation (26). Inserting coordinates of the test point P in this equation does not give zeroing the equation (26).

$$13 + 1 + 1 - 4 - 6 = 15 - 10 = 5 \neq 0$$

This result proves that the test-point $P(1,1,1)$ does not lie on the line AB.

6. Conclusions

The paper presented non-linear, one-equation models of a straight line in space 3D and in higher spaces. By presentation of the nonlinear model of a straight line the author of this paper, under no circumstance does not want to say that a straight line is a non-linear object, i.e. that a straight line is a curve. Information, which is transferred by this paper can be summarized in points as below.

6.1. A straight line can be modelled with one linear equation only in 1D- and 2D-space, only in low-order spaces. In case of 1D-space it is the parametric equation (27), in case of 2D-space it is the non-parametric linear equation (28).

$$x = x_A + \alpha(x_B - x_A) \tag{27}$$

$$a_2 x_2 + a_1 x_1 + a_0 = 0 \tag{28}$$

6.2. In case of 3D-space and of higher nD-spaces the straight line can't be modelled with only one linear equation. The line model has to consist here or of n linear parametric equations (5) or of $(n-1)$ linear equations (18) and (19) representing $(n-1)$ mutually intersecting hyper-planes.The both models are not one-equation but multiple-equation models.

6.3. The only one-equation model that in the 3D-space and in higher spaces can be constructed is a non-linear model. It can be of parametric form (29) or of the edge form (30).

$$\alpha^2 \sum_{i=1}^{3}(x_{iB} - x_{iA})^2 - 2\alpha \sum_{i=1}^{3}(x_i - x_{iA})(x_{iB} - x_{iA}) + \sum_{i=1}^{3}(x_i - x_{iA})^2 = 0 \tag{29}$$

$$C_1x_1^2 + C_2x_2^2 + C_3x_3^2 + C_4x_1x_2 + C_5x_1x_3 + C_6x_1 + C_7x_2 + C_8x_3 + C_9 = 0 \qquad (30)$$

6.4. One-equation, nonlinear models of the straight line in no circumstances exclude the hitherto used linear multi-models. Both model types can be used parallelly.

6.5. It seems that non-linear, one-equation models are more useful than linear multi-models in derivations of formulas. Author used them successfully in investigations connected with the contextual, non-regular coordinate-system $\alpha \times \beta \times \gamma$, Fig. 13.

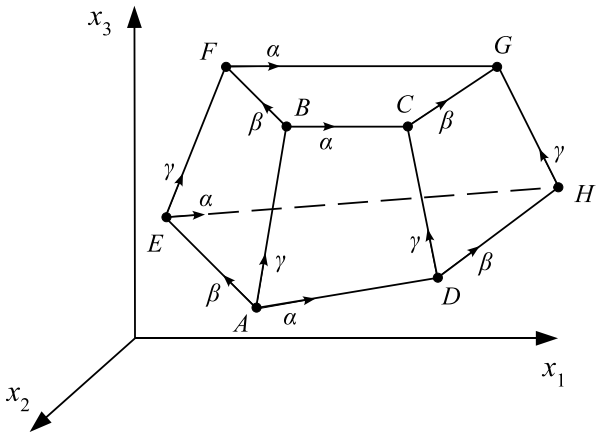


Figure 13. Example of the contextual, non-regular, coordinate-system $\alpha \times \beta \times \gamma$ based on the knowledge points A, B, C, D, E, F, G, H with known coordinate values delivered by the problem expert.

The contextual coordinate-system is necessary for construction of reasoning methods in non-regular, fuzzy-models [1]. The coordinates α, β, γ mean in these models compatibility (membership) degrees.

6.6. Analyses of the paper-text suggests an interesting general conclusion. Linearity or non-linearity of the one-equation mathematical model describing a mathematical object depends not only on the number and type of conditions which have to be satisfied by the object but also on order of the space in which the object is defined. If the condition number is very small (1,2), the conditions are linear and the space-order is low then the object can be modeled with only one linear equation. If the conditions number and the space order increase then constructing linear, one-equation models become impossible.

References

[1] A.Piegat, M. Olchowy. *Kontekstowy, jednosektorowy, nieregularny model rozmyty oparty na 4 punktach wiedzy. (Contextual, one-sector, non-regular fuzzy model based on 4 knowledge points). Pomiary-Automatyka-Kontrola (PAK)*, 56(10):1193–1196, 2010.

[2] R. Faber. *Foundations of Euclidian and non-Euclidian geometry*. Marcel Dekker, New York, 1983.

[3] F. Leja. *Analytical geometry*. Wydanie 10-te, PWN [in Polish], Warszawa, 1977.

[4] W. Bartol, et al. *Encyclopedia for everybody – Mathematics*. Wydawnictwo Naukowo-Techniczne [in Polish], Warszawa, 2000.

[5] K. Bieńkowska, et al. *Engineer handbook – Mathematics*. Wydawnictwo Naukowo-Techniczne [in Polish], Warszawa, 1971.

[6] N. Bernstein, K. Semendjajew, G. Musiol, H. Muehlig. *Taschenbuch der Mathematik (Handbook of Mathematics)*. 5-te ueberbeitete und erweiterte Auflage, Verlag Harri Deutsch, Thun und Frankfurt am Main, 2001.

[7] *Line (geometry)*. Wikipedia 2011. http://en.wikipedia.org/wiki/Line-(geometry).

Automatyczne przetwarzanie nagrań dźwiękowych do celów analizy

Edward Półrolniczak, Michał Kramarczyk

West Pomeranian University of Technology, Szczecin
Faculty of Comupter Science and Information Technology

Abstract:

In this article the authors present the idea of automatic processing of singing recordings to create a database and for later analysis. The recordings contain sequences of sounds of the piano and singing separated by short segments of silence. The aim of the presented approach is the precise segmentation of the recordings and standardization of the resulting segments (termed further as samples) for later assessment of vocal quality. The taken approach can reduce the amount of time required for preparation of a database with sequences of singing for the analysis of singing voice quality. The samples obtained in the process must meet the prescribed requirements to enable carrying out the planned analyzes such as estimation of the fundamental frequency, vibrato estimation, Singing Power Ratio and formants estimation.

Keywords:

singing voice, segmentation, sound preprocessing, singing voice database

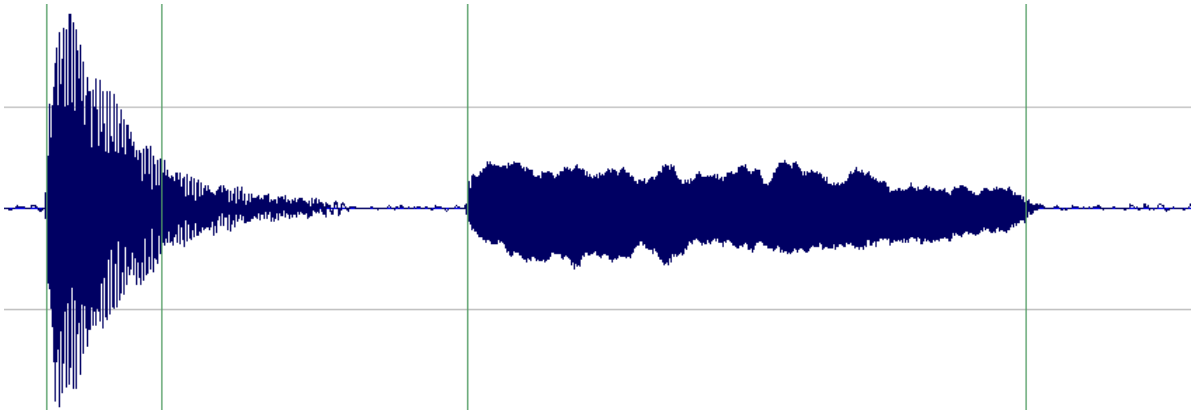
1. Wprowadzenie

Praca przedstawiona w poniższym artykule została wykonana w kontekście przygotowania bazy danych do analizy głosu śpiewaczego pod kątem jego jakości. Nagrania były dokonywane w sposób masowy, aby uzyskać jak największą objętość pozyskanych danych. Z uwagi na ilość przyjętego do nagrań materiału odstąpiono od realizacji zadania segmentacji manualnej na tym etapie. Wiązałaby się ona z ręcznym przycinaniem próbek, poprawą jakości, etykietowaniem. Wykonywana by ona była po każdej zaśpiewanej przez nagrywaną osobę sekwencji jednak wydłużyłoby to znacząco czas nagrań i nie pozostało bez wpływu na kondycję osoby nagrywanej a co za tym idzie na zawartość jakościową próbek śpiewu. W rezultacie do przetworzenia powstał duży objętościowo materiał, który należało przystosować do celów stworzenia bazy danych pod kątem późniejszych analiz [1] [2]. Analiza nieprzetworzonego sygnału dźwiękowego, wiąże się takimi problemami jak przekłamania wyników uzyskanych dla całości próbki czy zniekształcenia wyników na krańcach próbki dźwiękowej. Odpowiednie przetwarzanie wstępne pomaga zmniejszyć artefakty związane z próbkowaniem i kwantyzacją sygnału dźwiękowego podczas jego zapisu do postaci cyfrowej. Przetwarzanie wstępne jest zazwyczaj wieloetapowym procesem. Jeden etap nie daje pożądanych efektów. Przykładowo, w omawianym dalej rozwiązaniu, dokonywana jest normalizacja amplitudy sygnału zwiększając zarówno badany sygnał śpiewu jak i, niestety, zarejestrowany szum. Zabieg ten jednak daje możliwość rzetelnego porównywania sygnałów zarejestrowanych w inny sposób. Na przykład za pomocą innych mikrofonów albo zaśpiewanych przez różnych wykonawców. Segmentacja sygnału jest podstawowym krokiem mającym na celu ekstrakcję pożądanej

informacji z analizowanej próbki. W przypadku analizowanego tutaj sygnału zawierającego nagranie śpiewu, właściwa segmentacja pozwala nam usunąć fragmenty sygnału nie zawierające głosu śpiewaka, a będące tylko zarejestrowanym szumem otocznia. Sygnał poddawany tutaj segmentacji zawiera nagranie składające się z dźwięku pianina określającego ton oraz zaśpiewanej przez śpiewaka głoski. Segmentacja posłużyła tutaj do wyodrębnienia z ćwiczeń poszczególnych sekwencji pianina i śpiewu. Ważne było prawidłowe wnioskowanie na temat przerw w trakcie wybrzmienia dźwięku wzorcowego, które trwa praktycznie do rozpoczęcia śpiewania przez śpiewaka. Od pewnego momentu posiadana przez wybrzmienie energia jest równa, a nawet mniejsza niż szum, dlatego ważne jest, aby proces segmentacji przycinał próbkę w odpowiednim momencie, tak aby dla celów analizy przekazywany był pełnowartościowy sygnał bez udziału zbędnych próbek.

2. Możliwości powszechnie dostępnego oprogramowania do segmentacji nagrań dla celów analizy

Na rynku istnieje szereg aplikacji, które umożliwiają przeprowadzenie procesu automatycznej segmentacji. Problemem pozostaje cel w jakim programy te są zaprojektowane. W tym przypadku chodzi o podział dźwięku na segmenty, które zawierają cały sygnał dźwięczny nawet kosztem tego, że na krańcu sygnału znajduje się głównie szum. Chodzi tu głównie o podział na części pod kątem eksportu nagrania do ścieżek audio, albo pod kątem edycji rozdzielnych sekwencji w sensie masteringu. Nie jest ich celem przycięcie sygnału tak, aby dawać możliwie najlepsze rezultaty w analizie dźwięku, gdyż nie zostały przygotowane dla takiego zastosowania. Na rysunku 1 przedstawiono przykładowy wynik uzyskany w programie Sony Sound Forge Pro 10.0.



Rysunek 1. Przykładowy wynik uzyskany w programie do edycji dźwięku

W konfiguracji programu istnieje kilka opcji umożliwiających dostosowanie czułości modułu segmentacji na zjawiska narastania i opadania sygnału w nagraniu. Próby dostosowania parametrów pokazują, że nie można ich dobrać tak, aby uzyskać zamierzoną dla późniejszej analizy jakość odcięcia. Na pokazanym przykładzie wynik został wygenerowany przy ustawieniach: attack sensitivity: 75%, release sensitivity: 30% i minimum level 10%. Ilustracja pokazuje, że ustawienia te dość dobrze radzą sobie z sygnałem śpiewu, zupełnie źle z sygnałem pianina (po lewej). Należy jednak zauważyć, że również sygnał śpiewu został przycięty zbyt agresywnie - została utracona część wybrzmienia śpiewanej sekwencji. Dodatkowo zwraca

uwagę fakt silnego obcięcia sekwencji dźwięku z pianina (próbka po lewej stronie rysunku). Poszukiwania parametrów jak i programów, które bardziej nadawałyby się do zadania przygotowania próbek do późniejszej analizy spowodowały opracowanie własnego rozwiązania przedstawionego dalej.

3. Przetwarzanie wstępne do celów segmentacji

Etap ten jest istotnym dla osiągnięcia obranego celu jakim jest zgodna z zamierzeniami segmentacja nagrania dźwiękowego zawierającego dźwięk nagranego pianina i śpiewaka. Przetwarzanie wstępne ma wpływ na to, w którym miejscu zostaną wyznaczone punkty ocięcia próbki. Etap przetwarzania jest wieloetapowy i został zbudowany na podstawie powtarzanych czynności doboru poszczególnych etapów i ich parametrów. Niniejsze postępowanie doprowadziło do wyselekcjonowania optymalnego zestawu metod dla etapu wstępnego dalej wymienionych i opisanych.

3.1. Filtr preemfazy

Po etapie akwizycji w przypadku sygnałów mowy i śpiewu zaleca się zastosowanie filtru preemfazy [3]. Jako filtr preemfazy można zastosować:
- filtr górnoprzepustowy IIR (ang. Infinite Impulse Response) pierwszego rzędu,
- filtr górnoprzepustowy FIR (ang. Finite Impulse Response) pierwszego rzędu.

Filtr preemfazy dokonuje przeskalowania mocy sygnału. Wzmacniane są sygnały składowe w górnej części pasma przenoszenia. W prezentowanym podejściu zastosowano cyfrowy filtr górnoprzepustowy pierwszego rzędu o skończonej odpowiedzi impulsowej (FIR). W dziedzinie czasu filtr można opisać za pomocą równania (1). W prezentowanym rozwiązaniu przyjęto za parametr a wartość 0.97.

$$\acute{y}[n] = y[n] - ay[n-1], \tag{1}$$

gdzie:
- $\acute{y}[n]$ - sygnał wyjściowy,
- $y[n]$ - sygnał wejściowy,
- n - numer kolejnej próbki,
- a - parametr filtru preemfazy.

Zastosowany filtr powoduje zmniejszenie dynamiki sygnału, podbicie składowych wyższych częstotliwości, tłumienie składowych niskich częstotliwości. Dodatkowym efektem zastosowanej filtracji jest wyskalowanie sygnału ciszy i szumów do wartości bliskiej zeru. Takie przeskalowanie sygnału ułatwia wyznaczenie granic wydzielanych segmentów, co jest tutaj pożądaną cechą. Porównanie wyników przeprowadzanej tutaj segmentacji w warunkach z filtrem preemfazy i bez jego udziału pokazuje, że etap ten jest niezbędny. W przypadku nie zastosowania filtru preemfazy proces segmentacji nie wydzielił poprawnie 20% segmentów.

3.2. Usunięcie składowej stałej

Składowa stała jest zarejestrowanym składnikiem sygnału określanym w literaturze specjalistycznej jako DC. Problem ten powstaje w wyniku niewłaściwej pracy urządzeń rejestrujących lub zapisu [4]. Jest on trudny do usuwania w czasie rzeczywistym, ale dzięki jego niezmienności w czasie, dla już zarejestrowanych próbek istnieje efektywna metoda jego

przybliżenia. Polega ona na usunięciu wartości średniej z całej zarejestrowanej próbki, tak jak we wzorze (2).

$$\acute{y}[n] = y[n] - \frac{1}{N}\sum_{k=1}^{N} y[k], \tag{2}$$

gdzie:
- $\acute{y}[n]$ - sygnał wyjściowy,
- $y[n]$ - sygnał wejściowy,
- n - numer kolejnej próbki,
- N - liczba próbek.

Jest to szczególnie ważne np. w przypadku szacowania parametru wibrato w śpiewie. Dzięki temu etapowi możliwe jest również porównywanie nagrań z różnych mikrofonów. Bardziej istotne było tutaj zniwelowanie wpływu DC na wyniki analiz (np. wyznaczania składowej podstawowej).

3.3. Normalizacja do maksymalnej wartości

Ważnym elementem mającym na celu ujednolicenie oraz usystematyzowanie wyników, jest normalizacja ze względu na amplitudę sygnału. Normalizacja polega na przeskalowaniu amplitudy analizowanego sygnału do przedziału $< -1, 1 >$. Zabieg ten wykonuje się dzieląc wartości całego sygnału przez maksymalną zarejestrowaną wartość dla całego sygnału tak jak we wzorze (3).

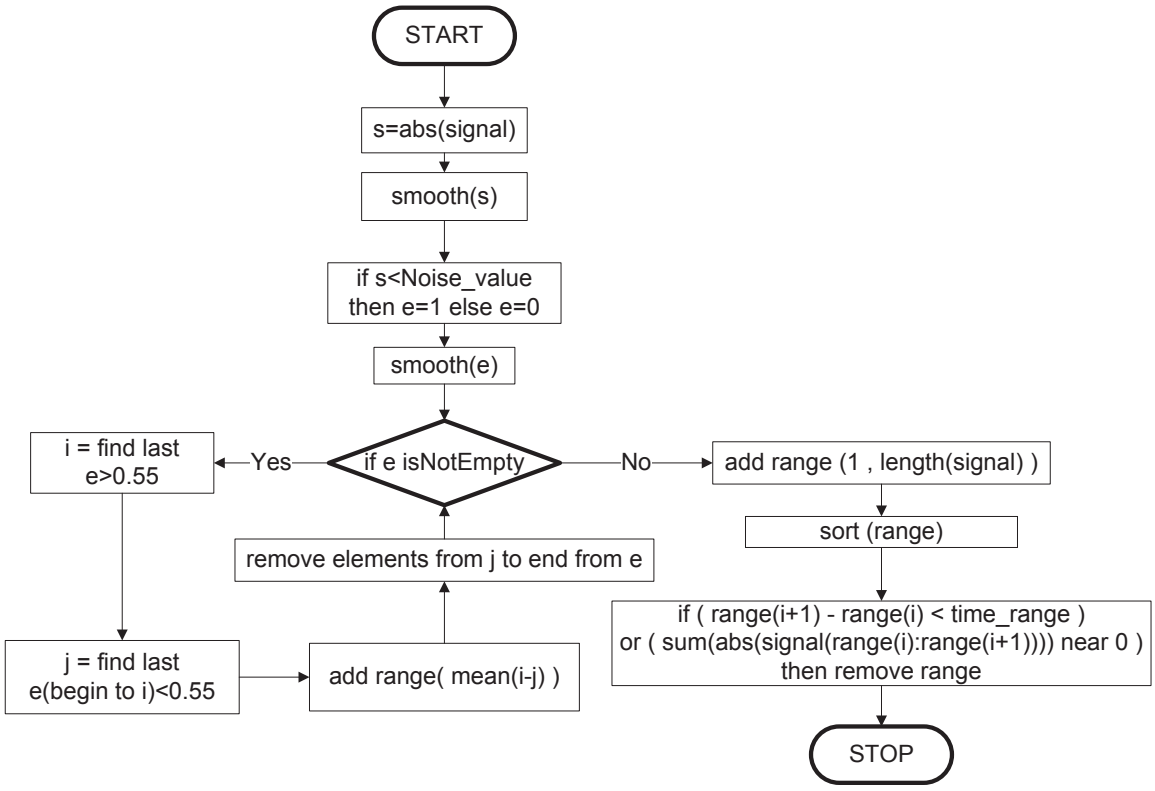
$$\acute{y}[n] = \frac{y[n]}{\max(abs(y[n]))}, \tag{3}$$

- $\acute{y}[n]$ - sygnał wyjściowy,
- $y[n]$ - sygnał wejściowy,
- n - numer kolejnej próbki.

Normalizacja całego nagrania ma za zadanie zachować naturalne zmiany amplitudy śpiewu, które zmieniały się w obserwowalny sposób w zależności od wysokości śpiewanego dźwięku. Etap ten jest niezbędny z uwagi fakt, iż poszczególne nagrywane osoby w różny sposób radziły sobie z wykonywaniem ćwiczeń, jak również posiadają zróżnicowane umiejętności wokalne. Istotną wartością tego etapu w kontekście segmentacji jest ujednolicenie różnic między fragmentami śpiewu i pianina a segmentami ciszy.

4. Segmentacja nagrań

Segmentacja nagrań w prezentowanym przypadku jest kluczowym elementem procesu przetwarzania nagrań pod kątem ich analizy z wykorzystaniem parametrów stosowanych w dalszej kolejności do oceny jakości śpiewu (analiza F0, analiza formantów, analiza Singing Power Ratio). Wstępnie przetworzone nagrania (preemfaza, usunięcie DC, normalizacja) muszą zostać poddane procesowi precyzyjnej segmentacji, aby każda próbka zawierająca istotne informacje (nagranie pianina, nagranie śpiewu) była właściwie przygotowana do celów analizy. Oczekiwaną cechą precyzyjnej segmentacji próbki dźwięku było takie odcięcie próbek zawierających szum, aby jak najmniejsza jego ilość pozostawała w próbce dźwięcznej. Dodatkowo należało spełnić warunek odcięcia w miejscu przejścia sygnału przez 0. Zapobiega to występowaniu wartości skokowych zarówno przy odsłuchu jak i analizie sygnału. Proces segmentacji jest tutaj realizowany metodą rozrostową z wykorzystaniem odpowiednio dobranych kryteriów.



Rysunek 2. Schemat pierwszego etapu segmentacji

Segmentacja składa się z dwóch etapów. Pierwszy z nich został przedstawiony na rysunku 2. Etap ten pozwala ustalić środki przedziałów, które wyznaczane są w środku dłuższych fragmentów ciszy. Algorytm do działania potrzebuje ustalenia kilku parametrów, które najlepiej wybrać empirycznie, na podstawie kilku próbnych uruchomień programu. Tymi parametrami są:

- time_range - minimalny czas jaki może trwać zarejestrowany segment (np. 750 ms),
- window_size - rozmiar okna, na jakim będzie wykonywane uśrednianie sygnału, ale wartość ta nie powinna być dużo dłuższa niż zarejestrowane w sygnale najkrótsze fragmenty szumu (np.20 ms),
- noise_value - najważniejszy parametr określający wartość uśrednionego sygnału, poniżej której założono, że znajduje się szum tła, a nie interesujący sygnał. Dla sygnału rejestrowanego dobrymi jakościowo mikrofonami zastosowanymi do zarejestrowania sygnału wartość parametru oscylowała w okolicach 10^{-3}.

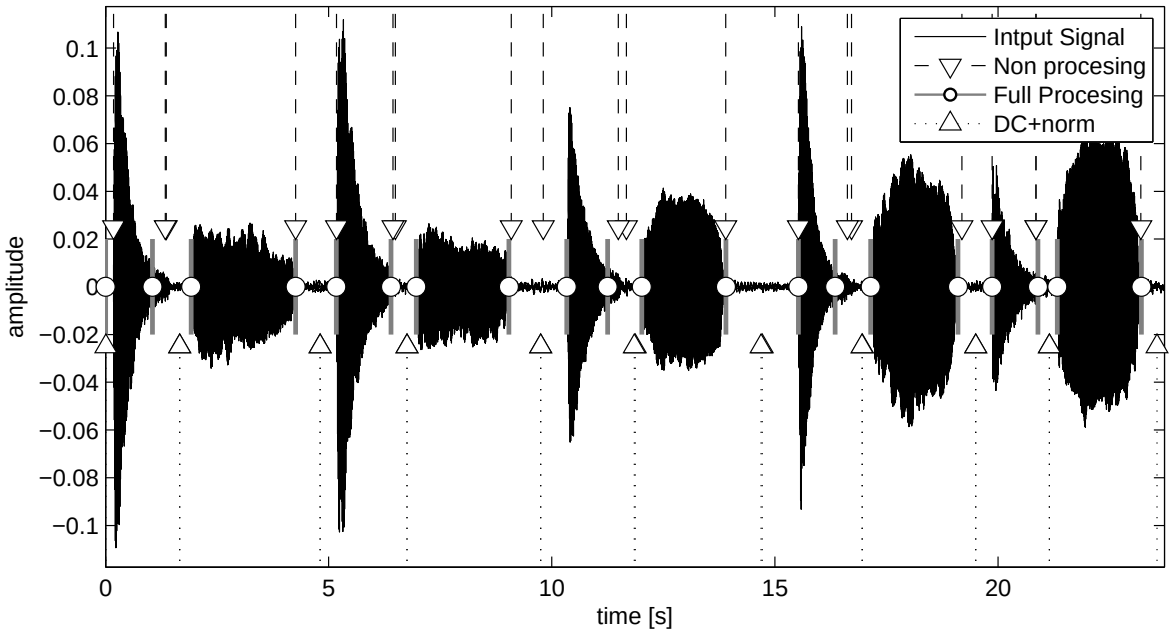
Segmentację rozpoczyna proces stworzenia średniej obwiedni sygnału. Proces ten przeprowadzany jest przez poddanie działaniu filtru uśredniającego wartości bezwzględnej sygnału. Należy pamiętać, aby okno było dostatecznie szerokie, aby obwiednia była odpowiednio gładka. Kolejnym etapem jest analiza otrzymanego wektora pod kątem wykrywania ciszy występującej w sygnale. Zadanie to realizuje się tworząc nowy wektor posiadający jedynki w miejscach, gdzie analizowany wektor posiadał wartości mniejsze od noise_value oraz zera w pozostałych miejscach. Następnie poddaje się ten wektor działaniu filtru uśredniającego, o takiej samej wielkości okna co poprzednio. Sprawia to że przedziały ciszy, które są krótsze od połowy okna filtru nie będą miały wpływu na dalszy etap segmentacji.

Dalej następuje iteracyjny etap poszukiwania przedziałów. Wykonuje się go tak długo dopóki, wektor nie jest pusty lub nie daje się już wykryć kolejnych przedziałów. W każdej iteracji wyszukiwany jest ostatni element o wartości powyżej 0.55, określany dalej jako i. Po operacji uśredniania wartość ta występuje tam, gdzie fragment ciszy ma długość zbliżoną do długości okna, na którym została przeprowadzona operacja uśredniania. Dalej należy wykryć ostatni element o wartości poniżej 0.55, ale o niższym indeksie niż ten wykryty w poprzednim kroku,określany jako j. Wyliczany jest środek przedziału pomiędzy znalezionymi indeksami i dopisywany do wektora $ranges$, w którym przechowywane są środki przedziałów ciszy . Następnie usuwane są z wektora wartości od indeksu j do końca wektora.

Po zakończeniu pętli do wektora przedziałów ciszy ($ranges$) dodawany jest dla pewności początek i koniec sygnału. Cały wektor sortowany jest rosnąco i iteracyjnie sprawdzane są kolejne pary przedziałów. Jeżeli odległość między dwoma kolejnymi elementami jest mniejsza niż time_range lub gdy zawartość przedziału zawiera sygnał zbliżony do zera to usuwany jest mniejszy element z pary i należy przejść do kolejnego zestawu.

Przy odpowiednio dobranych parametrach otrzymuje się w rezultacie wektor zawierający indeksy próbek, które oddzielają poszczególne segmenty. Dobierane są one dalej w pary tak, że tworzą zestawy zawierające początek i koniec przedziału. Wszystkich par powinno być o jeden element mniej, niż ilości elementów w wektorze zawierających indeksy próbek. Ostatnim etapem segmentacji jest dokładne dopasowanie granic przedziału. Dokonuje się tego dla każdego segmentu osobno, przesuwając obie skrajne granice przedziału do miejsca, w którym wartość obwiedni jest większa, niż trzykrotna wartość noise_value.

Należy również pamiętać, aby każdy marker opisujący granicę segmentu został umiejscowiony w miejscu przejścia sygnału przez zero. Zabieg ten powinien zmniejszyć negatywne efekty, które mogą mieć miejsce z powodu przesunięcia fazowego sygnału.



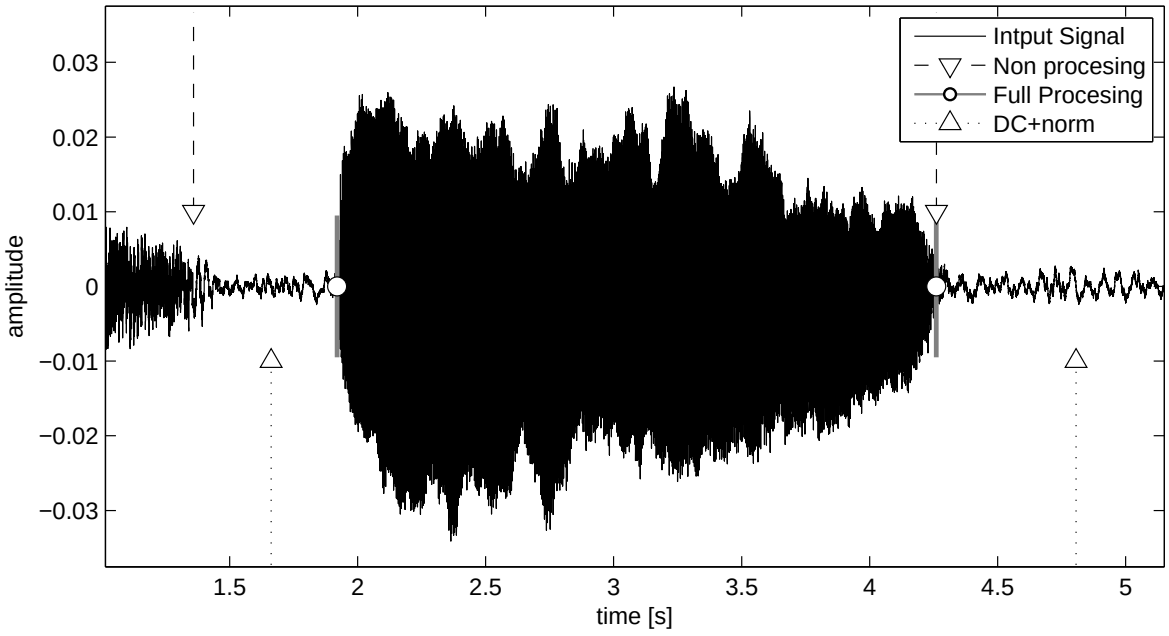
Rysunek 3. Przykład wpływu przetwarzania wstępnego na segmentację - fragment pliku dźwiękowego

5. Wpływ przetwarzania wstępnego na segmentację

Celem przetwarzania wstępnego było zarówno polepszenie działania segmentacji jak i unormowanie wyników pozwalające na dalsze porównywanie próbek z różnych źródeł. Do przedstawienia ich wpływu na proces segmentacji przygotowano trzy przykładowe zestawy testowe:

- Sygnał bez żadnych modyfikacji,
- Sygnał z pełnym przetwarzaniem wstępnym,
- Sygnał tylko z usunięciem DC i normalizacją bez usuwania preemfazy.

Segmentacja na wszystkich zestawach miała identyczne parametry, dlatego można porównywać dobór segmentów tylko pod kątem wpływu przetwarzania wstępnego na segmentację. Na rysunku 3 przedstawiono fragment nieobrobionego sygnału dźwiękowego wraz z naniesionymi granicami przedziałów. W górnej części wykresu znajdują się granice, wyznaczone na sygnale nie poddanemu obróbce. W większości przypadków prawidłowo zostało wykryte zakończenia segmentów. Duży problem jest z wykryciem początków segmentów. Zdarza się nawet, że początek jest wykrywany na poprzednich sygnałach. Na dole wykresu zostały przedstawione segmenty wykryte na podstawie sygnału z usuniętą składową stałą i znormalizowane. Jak widać segmenty zostały bezpiecznie wydzielone pomiędzy segmentami, ale wydzielone w ten sposób przedziały zawierały duże przedziały zawierające szum. Ostatni zestaw został zaprezentowany na środku wykresu i został oznaczony okręgami, przedstawia on efekt segmentacji osiągnięty na sygnale z pełnym przetwarzaniem wstępnym. Otrzymane przedziały zostały umiejscowione w zakładanych uprzednio miejscach i nie zawierają niepożądanego szumu tylko pożądany sygnał.



Rysunek 4. Przykład wpływu przetwarzania wstępnego na segmentację - jeden segment

Dokładniej samej segmentacji można przyjrzeć się na rysunku 4. Przedstawiony na nim został jeden wyodrębniony segment wraz z zaznaczonymi granicami w ten sam sposób jak rysunku 3. Początek segmentu wykrytego na nie przetworzonym sygnale, zaczyna się jesz-

cze na wybrzmieniu poprzedniego dźwięku i zawiera przez to duży fragment sygnału nas nieinteresującego. Mimo to koniec segmentu został wyznaczony bardzo dobrze. Jeżeli chodzi o granice wyznaczone na sygnale z częściowym przetworzeniem wstępnym, widać jak duże są błędy z jakimi zostały wyznaczone marginesy. Ten sygnał również zawiera duże ilości niepotrzebnego sygnału. Sygnał przetworzony wstępnie w sposób pełny, zgodnie z tutaj opisanym sposobem segmentacji, uzyskał najlepsze parametry wielkości przedziału, minimalizując udział szumu w wyodrębnionym fragmencie. Podobna jakość segmentacji została zachowana niezależnie od stopnia zakłóceń sygnału. Ponieważ dla pewności rejestracji wartościowych danych w procesie używano kilku mikrofonów o różnej charakterystyce, dlatego możliwe było sprawdzenie zaproponowanej metody automatycznej segmentacji również na odpowiadających sobie próbkach pochodzących z różnych urządzeń wejściowych. Tu również uzyskano powtarzalność wyników segmentacji. Można zatem stwierdzić, że opracowana metoda segmentacji jest uniwersalna, przynajmniej dla przyjętego toku postępowania przy tworzeniu bazy danych sekwencji śpiewu.

6. Wnioski

W niniejszym artykule przedstawiono ideę automatycznej segmentacji sekwencji śpiewu w celu przygotowania bazy danych do oceny parametrów jakościowych głosu śpiewaczego. Przygotowanie bazy danych, która byłaby wartościowa z punktu widzenia naukowego musi spełniać określone kryteria. Pierwszym z nich jest spełnienie wymogów co do próbek zgromadzonych w bazie pod kątem jej zastosowania, jakim jest ocena jakości głosu śpiewaczego. Drugim kryterium jest zgromadzenie w bazie odpowiednio dużego materiału pozwalającego na przeprowadzenie szerokich i wiarygodnych analiz. Opracowana metoda segmentacji pozwoliła spełnić powyższe kryteria jak również znacząco skrócić czas tworzenia bazy danych. Bez przygotowanego w opisany sposób procesu nie byłoby możliwe stworzenie bazy danych do badań nad jakością głosu śpiewaczego a co za tym idzie prowadzenie dalszych badań.

Praca powstała w ramach projektu finansowanego przez Ministerstwo Nauki i Szkolnictwa Wyższego, Nr N N516 517539.

Bibliografia

[1] E. Półrolniczak, M. Łazoryszczak. *Quality assessment of intonation of choir singers using F0 and trend lines for singing sequence*. *Metody Informatyki Stosowanej, Polska Akademia Nauk Oddział w Gdańsku, Komisja Informatyki*, 4:259–268, 2011.

[2] E. Półrolniczak, M. Łazoryszczak, T. Mąka. *The impact of the type of sung vowels on the singing quality*. *Elektronika – konstrukcje, technologie, zastosowania*, 11:112–113, 2011.

[3] K. Paliwal. *Effect of preemphasis on vowel recognition performance*. *Speech Communication*, 3(1):101 – 106, 1984.

[4] G. Hu, D. L. Wang. *Monaural Speech Separation*. *ADVANCES IN NEURAL INFORMATION PROCESSING SYSTEMS*, 15:1245–1252, 2003.