Viktor MOKEROV[1], Waldemar WÓJCIK[2], Tatyana BALOVA[1]

D. Serikbayev East Kazakhstan state technical university (1), Lublin University of Technology (2)

Ontology based method for process oriented systems design

Abstract. *Article considers use of ontology to solve task of information systems design based on service oriented architecture. To support the design process, ontology corresponding to conceptual and logical levels of functional representation of Zachman's model is developed. Evaluation of semantic similarity of functions for forming system architecture based on processes model is suggested. Use of suggested ontology in system designing allows organizing the knowledge management process at initial stages of a system lifecycle.*

Streszczenie. *W artykule rozpatrzono zastosowanie ontologii w zadaniu projektowania systemów informacyjnych, opartych na architekturze zorientowanej na usługach. Dla wsparcia procesu projektowania opracowano ontologię odpowiadającą konceptualnemu i logicznemu poziomowi funkcjonalnego przedstawienia modelu Zachmana. Zaproponowano ocenę semantycznego podobieństwa funkcji do tworzenia architektury systemu na podstawie modelu procesów. Zastosowanie zaproponowanej ontologii do projektowania systemów pozwala organizować proces zarządzania wiedzą na wstępnym etapie cyklu życia systemu. (**Metoda projektowania systemów zorientowanych procesowo oparta na modelu ontologicznym**)*

Keywords: information systems design, semantic similarity, clustering.
Słowa kluczowe: projektowanie systemów informatycznych, podobieństwo semantyczne, klasteryzacja.

doi:10.12915/pe.2014.10.51

Problem definition

Approach of system architecture division into architectural views describing systems from a certain point of view is used for integrated description of complex information systems [1]. The most applied in practice is the model of architectural representations suggested by Zachman. Zachman's model describes different aspects of a system from position of different participants of design process [2]. On design stage architect solves the task of reflecting elements of conceptual view of a system at logical level. To solve this task it is necessary to provide for transparent reflection between these levels of system representation. Traditionally modeling languages: UML, IDEF, BPML and others are used for expression of different views of a system. However, although these languages, particularly UML, support semantic extensions, more precise description of system semantic can be reached using knowledge description languages.

Process oriented information systems can be implemented in form of web-services [3], or in more general sense using service oriented architecture [4]. To automate formation of service oriented system structure, it is necessary to develop quantitative characteristic of relationships between system elements. Semantic similarity of system elements, that determines the possibility of including operations into one service, can be used as such characteristic. Use of such characteristic will allow implementing clustering algorithms for structuring of system elements.

Analysis of recent researches and publications

Ontologies and other semantic technologies are widely used to solve tasks of information systems design. In paper of Kremen and Kouba [5] approach to information systems design based on ontological model is offered. The nature of the approach lies in formation of subject oriented system structure based on knowledge described in ontology. Ontology describes contents of domain area. Authors suggest algorithms of checking systems integrity based on establishing match between ontological model and object representation. Object oriented architecture can encapsulate functions of service oriented system. Technology Windows Communication Foundation can be an example of such implementation where application programmers work within object oriented system framework. This is why approach used by the authors can also be used in service oriented systems. However, such approach requires formation of additional elements hiding service oriented functionality.

In work Strasunskas and Hakkarainen [6] method for recognizing dependable elements of a system based on annotation of all elements in accordance with semantic is presented. Domain area is presented as ontology. To determine relationships of system elements its semantic similarity is calculated based on concepts reflected in annotation. Determination of such relationships is one of the most current tasks of system management. In ontology relationships system structures are determined through roles. In both cases relationships shall be shown well defined.

In paper of Bolloju and Sugumaran [7] approach to development of object models based on knowledge is presented. Analysis of object model is done to determine system ontology match. System ontology expresses connections and hierarchy of system elements. Rules for naming, inheritance and implementation of classes are determined in ontology. Control is done at level of object model classes.

Method for generation of system architecture elements based on system description using formal logic is suggested in [8]. Due to introduction of strict formalization, system elements can be created automatically, which increases transparency of a system and fastens design process. Additional expenses for formalization of system elements using formal logic means as well as necessity to organize additional visual views of a system can be considered as disadvantages of the method. In paper [9] approach to information systems design automation based on task data analysis is suggested. Author developed task semantic analysis system with further formation of classes diagrams and data flows. Advantage of such approach is the decrease of system design costs due to automatic generation of data model. Dependence on quality of requirements processing can be considered as main disadvantage of such approach. Methods described above built on the principle of more detailed formalization of system elements. At the same time in [10] approach to visual design that does not require detailed processing of technical task or other formalized descriptions is given. Nature of such method lies in provision of unity of system visual representation elements at all levels of system lifecycle. I.e. design, implementation and testing are done using the same representation of a system. Method is initially developed for automation of process for design of

physical objects, this is why its use for information systems design is difficult due to large quantity of system elements and difficulty to develop one representation for system at all stages of its lifecycle.

In addition to system elements generation based on ontology and metamodel, information system interface can be formed. In article [11] approach to design of services using metamodeling is considered. The approach is aimed at automation of service interface formation based on metamodel. Metamodel summarizes representations of users, customer, experts and designers on service functionality.

System re-engineering or modification can be considered as another direction of design automation. Main distinction of this direction is presence of existing system, i.e. design process was done at least one time. Automation in this direction is mostly related to system preparation to modernization. It is often necessary to solve tasks of integration with other systems and tasks of transferring system into new technological platform during system modernization. In paper [12] UELM, unified corporative modeling language, is presented. The main purpose of this language is integration of information systems models expressed in different languages. Generic ontology lies in the base of UEML. Information system model expressed in one of modeling languages can be reflected on generic ontology. In UEML the views of most popular modeling languages are determined. Presently works on extending supported modeling languages are being performed.

Summarizing all mentioned above, it can be maintained that at present approaches to design based on system ontology are widely used in scientific society. However, almost all approaches are aimed at interaction with object oriented systems. For service oriented systems having the same concept automation of design shall implement a little bit different quality attributes: louse coupling, performance, scalability.

The **main goal** of the research is to develop design automation technology for service oriented system based on ontological model.

The following **main tasks** are presented based on this:

1. development of ontological model for service oriented information system;
2. development of algorithm for formation of system services structure based on knowledge of processes.

Ontology of service oriented system

To solve first task, ontology of service oriented system including knowledge of system structure and domain area was developed.

Conceptual level of service oriented information system can be represented by set of processes in a system. Let the stages of business process be a function. In general case the same function can be used by different processes. For example, calculating current balance of the account.

At logical level system based on service oriented architecture consists of services and messages. System services are represented by a set of operations that they provide. Messages contain information, which is exchanged by services within process organization. As an assumption let us take that information participating in exchange always belongs to some content – document, table, etc. In the process of exchange service operation can perform recording and reading of data and some contents. Composition of contents that shall be reflected in knowledge database depends on information system domain area. Separate ontologies are developed for different domain areas. This is why it is feasible to perform ontological engineering of domain area, in order to organize separate ontology containing semantic of domain area. Therefore, information system structure ontology in notation of formal logic OWL 2 DL (SROIQ) can be presented as follows:

- Concepts (Classes):
  Entity – domain area content;
  Process – process;
  ProcessFunction – function of process;
  Service – service;
  ServiceOperation – operation of service.
- Roles (Relationships):
  ConsistsOf – consists of;
  CallFunction – calls for function;
  ReadInfo - extracts;
  Writeinfo – renews;
  Include – includes;
  isOperationOf – operation of service;
  Reflect – function reflection.
- Axioms:
  $Process \equiv \exists\ ConsistsOf.F$
  $F \equiv \exists\ ReadInfo.E \sqcup \exists\ Writeinfo.E$
  $S \equiv \exists Include.F$
  $isOperationOf \in F$, where F - set of functional roles.

Structure of presented ontology formed using package Protégé is given in Figure 1.

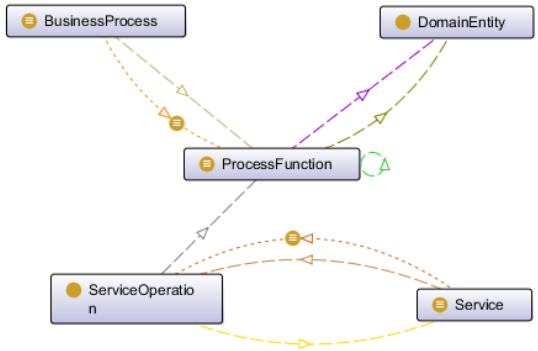

Fig.1. Structure of service oriented system ontology

Let's introduce the following interpretation.

Entity concept describes all contents of domain area described in domain ontology. *Entity* works as superconcept for all contents of domain area ontology. Concept *Process* represents domain area processes. Instances of this concept are formed as a result of system analysis of domain area. In accordance with Zachman's model processes are reflected in conceptual representation. Visual modeling languages UML [13], BPMN [14], IDEF0 [15] can be used to model processes. Description of processes in given notations can be brought to ontology as suggested in [12, 16, 17]. Concept *ProcessFunction* describes functions that provide for processes completion. Concept *Service* describes portal services. *ServiceOperation* describes service operations. Service operation is used to perform one or few functions in the process.

The following assumptions are used as limitations:

– every process consists of at least one function;
– every function shall record or read data from at least one content;
– one process stage can initiate another;
– service consists of at least one operation.

Role *ConsistsOf* determines membership of function in process. To express call of one function from another role *Call* is used. Role *Read* sets reading relationships between function and domain area content. *Write* – writing relationship. Role *Include* determines inclusion of operation into service as an operation. Reverse role *isOperationOf*

sets functional relationship of function membership in service. To reflect service operation at process function, role Reflect is introduced.

Algorithm of system services structure formation

Process of system design can be represented as creation of logical representation based of conceptual representation. Then for information system using service oriented architecture one of the main design tasks is to reflect processes at services implementing them. Process can be represented as set of some functions and service as set of operations. In this case to solve this task, first, it is necessary to determine composition of services, second, set reflection of function at service operation.

In given ontology role *Reflect* is used to set reflection of function at service operation. To automate task decision it is necessary to determine some algorithm connecting instances of service operations and process functions by role *Reflect*.

When services are formed architect follows functions semantic to integrate them into services. Evaluation of semantic similarity of process functions suggested in paper is based on the following assumptions:

- semantic similarity of functions that are called together within framework of larger quantity of processes is higher than similarity of functions used separately;
- the bigger the quantity of contents interacted together with function compared to total quantity of related contents, the higher semantic similarity of functions.

Following given assumptions and developed ontology, the following measure of semantic similarity of two functions is suggested:

$$
\begin{aligned}
&S(f_x, f_y) = \sum_{i=1}^{n} w_i SI_i \\
&\sum w_i = 1 \\
&SI_{BP}(f_x, f_y) = \\
&\frac{\left|\exists Process.ConsistsOf.f_x \cap \exists Process.ConsistsOf.f_y\right|}{\left|\exists Process.ConsistsOf.f_x \cup \exists Process.ConsistsOf.f_y\right|} \\
&SI_{ER}(f_x, f_y) = \\
&\frac{\left|\exists Entity.ReadInfo.f_x \cap \exists Entity.ReadInfo.f_y\right|}{\left|\exists Entity.ReadInfo.f_x \cup \exists Entity.ReadInfo.f_y\right|} \\
&SI_{EW}(f_x, f_y) = \\
&\frac{\left|\exists Entity.WriteInfo.f_x \cap \exists Entity.WriteInfo.f_y\right|}{\left|\exists Entity.WriteInfo.f_x \cup \exists Entity.WriteInfo.f_y\right|}
\end{aligned}
\tag{1}
$$

where: $SI_{BP}(f_x,f_y)$ – determines measure of functions relationship as ratio of cardinal number of process set using both functions to cardinal number of process set using at least one function;

$SI_{ER}(f_x,f_y)$ – determines measure of functions relationship as ratio of cardinal number of set of contents extracted in both functions to cardinal number of set of contents extracted by at least one function;

$SI_{EW}(f_x,f_y)$ - determines measure of functions relationship as ratio of cardinal number of set of contents renewed in both functions to cardinal number of set of contents renewed by at least one function.

Using given measure of functions similarity, the task for automation of system logical representation formation can be brought to the task of clustering of functions into sets matching services with further integration of functions inside sets in service operations.

Functional maximizing semantic similarity between functions of one service can be a criterion of clustering quality:

$$G = \sum_i^k \sum_{f_x \in S_i} SI(f_x, f_i) \to \max$$

where: k – quantity of clusters, f_i – center of cluster S_i; SI – measure of semantic similarity of processes functions.

Operation flow processes can have linear areas implemented by functions belonging to one service. Since for functions inside area outside interfaces are not defined, it is reasonable to replace such areas of processes by one operation encapsulating behavior of internal functions.

Definition. Linear sequence of calls (*LSC*) of functions in some service S_l will be considered as subset $F^* \subseteq F$, for which the following conditions exists:

- each function $f_i \in F^*$ related through role Include with S_l;
- any two functions of this subset are transitively connected by role *Call* only through functions of this subset;
- any function of this subset is not transitively connected by role *Call* with only one function of this subset;
- any function of this subset is not transitively connected by role *Call* with only one function of this subset.

Reflection of functions at service operations can be determined by the following rules:

a) if for function $f_i \in S_l$ there are no function $f_j \in S_l$ that calls for function f_i and no function $f_k \in S_l$ that is called by f_i, then function f_i serves as separate operation of a service;
b) if there are set of functions $F^* \in S_l$, forming *LSC*, then this *LSC* serves as a separate operation of a service;
c) functions that are not included into a) and b) serve as separate operations of a service.

Thereby, at the design stage the automation of conceptual representation transformation into logical can be done using roles set by system ontology. Implementation of this transformation is limited by service oriented architecture since using other architectural patterns can require another definition of semantic similarity and extension of ontology composition.

Experimental evaluation of the method

Experimental evaluation of suggested method was done at processes of electronic trading platform developed within framework of grant financing program No. 210 dated May'31 of 2010 (118-420-10). The fragment of electronic platform used to evaluate the method included the following:

- processes: Purchases, Sales, Marketing.Misc;
- contents: Agreements, Opportunities, Product, Classificator, Offers, OfferRequestAccounts, Products, MarketingCompany;
- functions: ProductClassification, CompanyStatistics, Dealing, DealsAnalisys, LocalContentEstimation, AnalyseOffer, InsertOffer, FindBestOffer, PaymentValidation, PriceAnalisys, ProductPromotion, AccountRegistration, RequestInsert, RiskAnalisys, Subscribe,MarketExplore.

Based on process model of electronic trading platform the relationships of functions and processes were determined by role *ConsistsOf* . They are presented in Table 1. Based on the results of analysis of information requests to database, functions relationships with domain area contents through roles *ReadInfo* and *WriteInfo* were determined. They are shown in the Table 2.

There are functions that change condition of only one content in considered fragment of electronic trading platform. If information is both written and read from a certain table, then it is assumed that role *WriteInfo* is used, therefore there are no functions performing reading of content Product Classificator as well as relationships for functions writing and reading data from same contents in the Table 2 (F1, F12).

Table 1. Matrix of relationships of functions and processes of electronic trading platform

Function Name	Id	Purchases	Sales	Marketing	Misc
ProductClassification	F1	1	1		
CompanyStatistics	F2			1	
Dealing	F3	1			
DealsAnalisys	F4	1		1	
LocalContentEstimation	F5	1	1	1	
AnalyseOffer	F6	1			
InsertOffer	F7		1		
FindBestOffer	F8	1			
PaymentValidation	F9	1	1		
PriceAnalisys	F10			1	
ProductPromotion	F11			1	
AccountRegistration	F12				1
RequestInsert	F13		1		
RiskAnalisys	F14	1		1	
Subscribe	F15				1
MarketExplore	F16			1	

Table 2. Matrix of relationships of functions and domain area contents through roles WriteInfo / ReadInfo

	Agreements	Opportunities	Product Classificator	Offers	Offer Request	Accounts	Products	Marketing Company
F1			1/				1/	
F2				/1	/1	/1	/1	
F3	1/	1/			1/	/1	/1	
F4	/1	/1				/1	/1	
F5				/1			/1	
F6		/1		/1	/1		/1	
F7				1/		/1	/1	
F8				/1			/1	
F9	/1			/1	/1	/1		
F10				/1	/1		/1	
F11						/1	/1	1/
F12						1/		
F13	1/	1/		/1	1/	/1	/1	
F14		/1		/1	/1		/1	
F15						/1		/1
F16				/1		/1	/1	/1

Table 3. Semantic similarity of functions

	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14	F15	F16
F1	0.70	0.00	0.25	0.17	0.33	0.25	0.25	0.25	0.50	0.00	0.00	0.00	0.25	0.00	0.00	0.00
F2	0.00	0.80	0.15	0.35	0.32	0.18	0.15	0.15	0.18	0.73	0.65	0.00	0.23	0.68	0.06	0.68
F3	0.25	0.15	1.00	0.40	0.27	0.56	0.30	0.60	0.31	0.08	0.30	0.00	0.40	0.06	0.10	0.15
F4	0.17	0.35	0.40	0.80	0.39	0.35	0.15	0.31	0.27	0.30	0.40	0.00	0.12	0.35	0.06	0.35
F5	0.33	0.32	0.27	0.39	1.00	0.32	0.47	0.47	0.39	0.37	0.27	0.00	0.37	0.32	0.00	0.32
F6	0.25	0.18	0.56	0.35	0.32	0.80	0.06	0.65	0.35	0.23	0.06	0.00	0.12	0.30	0.00	0.10
F7	0.25	0.15	0.30	0.15	0.47	0.06	1.00	0.10	0.31	0.08	0.30	0.00	0.70	0.06	0.10	0.15
F8	0.25	0.15	0.60	0.31	0.47	0.65	0.10	0.80	0.31	0.20	0.10	0.00	0.20	0.15	0.00	0.15
F9	0.50	0.18	0.31	0.27	0.39	0.35	0.31	0.31	0.80	0.12	0.06	0.00	0.37	0.10	0.06	0.10
F10	0.00	0.73	0.08	0.30	0.37	0.23	0.08	0.20	0.12	0.80	0.58	0.00	0.15	0.73	0.00	0.62
F11	0.00	0.65	0.30	0.40	0.27	0.06	0.30	0.10	0.06	0.58	1.00	0.00	0.20	0.56	0.10	0.65
F12	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.70	0.00	0.00	0.70	0.00
F13	0.25	0.23	0.40	0.12	0.37	0.12	0.70	0.20	0.37	0.15	0.20	0.00	1.00	0.12	0.08	0.23
F14	0.00	0.68	0.06	0.35	0.32	0.30	0.06	0.15	0.10	0.73	0.56	0.00	0.12	0.80	0.00	0.60
F15	0.00	0.06	0.10	0.06	0.00	0.00	0.10	0.00	0.06	0.00	0.10	0.70	0.08	0.00	1.00	0.15
F16	0.00	0.68	0.15	0.35	0.32	0.10	0.15	0.15	0.10	0.62	0.65	0.00	0.23	0.60	0.15	0.80

Based on ratio of relationships quantity to total quantity of elements in Tables 1 and 2 when calculating general measure of functions similarity the following coefficients of specific measure of similarity were used: SI_{BP} = 0.5; SI_{ER} = 0.3; SI_{EW} = 0.2.

When coefficients were chosen the differences of service oriented architecture were taken into account; the main task of such architecture is support of distributed processes that requires increasing influence of relationships among business processes on result of clustering. Role *WriteInfo,* on the other hand, uses limited set of functions and will influence clusters structure a lot under high values of coefficient.

Results of calculating measure of semantic similarity of function SI in accordance with (1) are given in Table 3.

Obtained evaluation of semantic similarity was used for functions clustering. To perform clustering the algorithm FOREL was used. Experiment was done for search radius of neighbor elements 0.5 and 0.73. Radius 0.5 corresponds to half maximum distance between elements, 0.73 – to

average value of distances matrix composed from additions to one of elements from semantic similarity matrix.

The following clusters were created when search radius 0.5 was used:

- CompanyStatistics, PriceAnalisys, ProductPromotion, RiskAnalisys, MarketExplore;
- DealsAnalisys;
- LocalContentEstimation;
- Dealing, AnalyseOffer, FindBestOffer;
- InsertOffer, RequestInsert;
- ProductClassification, PaymentValidation;
- AccountRegistration, Subscribe.

The following clusters were created when search radius 0.73 was used:

- Dealing, DealsAnalisys, AnalyseOffer, InsertOffer, FindBestOffer, PaymentValidation, ProductPromotion, RequestInsert;
- CompanyStatistics, LocalContentEstimation, PriceAnalisys, RiskAnalisys, MarketExplore;
- ProductClassification;
- AccountRegistration, Subscribe.

There are no linear sequences of calls in obtained clusters therefore all obtained functions can be presented as service operations.

As results of experimental evaluation showed, set of formed services correlates with services, design of which was performed empirically without using means of automation.

Results showed:

1) functions close by semantic entered into one cluster, for example, functions of analysis and accumulation of statistics are distributed into two clusters separately from functions performing trade procedures and work with credentials;
2) functions with louse coupling not depending on used radius of neighbor elements search are chosen into separate cluster (service), for example, market research function;
3) distribution of functions with strong coupling depends on clustering parameters that allows identify such functions during further attempts varying radius of neighbor elements search;
4) when radius close to average of distance matrix was used, formed clusters correspond to higher degree to services obtained during empirical design without using automatic generation of system structure.

Based on clusters formed for search radius 0.73 the following services can be identified:

- performing trading procedures;
- statistics and analysis;
- support functions for content management;
- work with credentials of trading platform participants.

Identification of functions with higher degree of relationships allows designer to optimize process model with the purpose of reducing relationship of system elements.

Conclusions

Use of ontology reflecting system structure allows using it as a base for system design. Suggested ontology allows executing requests reflecting relationship of domain area with elements of system structure. Roles determined at ontology are used to evaluate functions similarity in service oriented architecture of system that allows performing automatic reflection of conceptual description of system to logical level.

The results of experimental evaluation showed:

- obtained clusters integrate into functions close by semantic;
- reduction of radius of neighbor elements search allows choosing special functions (such evaluation of local content of a deal) into separate services that can be available from different modules of system;
- suggested method can be well used to choose functions with louse coupling by context and semantic;
- method allows determining elements with high degree of relationship to which designer shall pay attention.

In conclusion, use of suggested method allows modeling structure of service oriented system with the purpose of evaluating relationship and, therefore, scalability of system. Obtained model can be used as system prototype.

REFERENCES

[1] Bass L., Clements P., Katsman R., Software architecture in practice, *2nd ed. St. Petersburg: Peter*, 2006

[2] Zachman J.A., A framework for information systems architecture, *IBM Systems Journal*, 38 (1999), no. 2.3, 454-470

[3] Wangler B., Åhlfeldt R.-M., Perjons E., Process Oriented Information Systems Architectures in Healthcare, 1998

[4] Duarte F.J., Machado R.J., Fernandes J.M., Automated Information Systems Generation for Process-Oriented Organizations, *Proc. 6th International Conference on the Quality of Information and Communications Technology, QUATIC 2007*, Lisbon, Portugal, 2007, 223-227

[5] Kremen P., Kouba Z., Ontology-Driven Information System Design, *IEEE transactions on systems, man, and cybernetics—part c: applications and reviews,* 42 (2012), no. 3, 334-344

[6] Strasunskas D., Hakkarainen S.E., Domain model-driven software engineering: A method for discovery of dependency links, *Information and Software Technology*, 54 (2012), no. 11, 1239-1249

[7] Bolloju N., Sugumaran V., A knowledge-based object modeling advisor for developing quality object models, *Expert Systems With Applications*, vol. 39 (2012), no. 3, 2893-2906

[8] Davies J., Faitelson D., Welch J., Domain-specific Semantics and Data Refinement of Object Models, *Electronic Notes in Theoretical Computer Science*, vol. 195 (2008), 151-170

[9] Orlova Yu.A., Analysis of models and methods for increasing efficiency of software designing process, *Izvestiya VolgGTU*, vol. 11 (2010), no. 9, 137-141

[10] Russell M., Using MBSE to Enhance System Design Decision Making, *Procedia Computer Science*, (2012) no. 8, 188-193

[11] Fogli D., Provenza L.P., A meta-design approach to the development of e-government services, *Journal of Visual Language and Computing*, 23 (2012), no. 2, 47-62

[12] Anaya V., Berio G., Harzallah M., Heymans P., Opdahl A.L., Jose M., The Unified Enterprise Modelling Language — Overview and further work, *Computers in Industry*, 61 (2010), 99-111

[13] Object Managment Group, OMG Unified Modeling LanguageTM (OMG UML), Superstructure (2011)

[14] Object Managment Group, *Business Process Model and Notation (BPMN)* (2011)

[15] Integration definition for function modeling, *National Institute of Standards and Technology* (Maryland, USA) (1993)

[16] Robles K., Fraga A., Morato J., Llorens J., Towards an ontology-based retrieval of UML Class Diagrams, *Information and Software Technology*, vol. 54 (2012), no. 1, 72-86

[17] Henderson-Seller B., Bridging metamodels and ontologies in software engineering, *The Journal of Systems & Software*, vol. 84 (2011), no. 2, 301-313

***Authors**: Viktor Mokerov, D. Serikbayev East Kazakhstan state technical university, 69 A.K. Protozanova Street, 070004, Ust-Kamenogorsk, Kazakhstan, E-mail: viktor.o.mokerov@ieee.com; prof. Waldemar Wójcik, Lublin University of Technology, Institute of Electronics and Information Technology, 38a Nadbystrzycka Str., 20-618 Lublin, E-mail: waldemar.wojcik@pollub.pl; Ph.D. Tatyana Balova, D. Serikbayev East Kazakhstan state technical university, 69 A.K. Protozanova Street, 070004, Ust-Kamenogorsk, E-mail: tbalova@ektu.kz*