INFORMATION SCIENCE & COMPUTING

International Book Series Number 4

Advanced Studies in Software and Knowledge Engineering

Supplement to
International Journal "Information Technologies and Knowledge" Volume 2 / 2008

I T H E A
SOFIA, 2008

Krassimir Markov, Krassimira Ivanova, Ilia Mitov (ed.)

Advanced Studies in Software and Knowledge Engineering

International Book Series "INFORMATION SCIENCE & COMPUTING", Number 4

Supplement to the International Journal "INFORMATION TECHNOLOGIES & KNOWLEDGE" Volume 2 / 2008

Institute of Information Theories and Applications FOI ITHEA

Sofia, Bulgaria, 2008

This issue contains a collection of papers concerning themes in the field of Software and Knowledge Engineering. Papers are selected from the International Conferences of the Joint International Events of Informatics "ITA 2008", Varna, Bulgaria.

International Book Series "INFORMATION SCIENCE & COMPUTING", Number 4

Supplement to the International Journal "INFORMATION TECHNOLOGIES & KNOWLEDGE" Volume 2, 2008

Edited by **Institute of Information Theories and Applications FOI ITHEA**, Bulgaria,
in collaboration with
- **V.M.Glushkov Institute of Cybernetics of NAS**, Ukraine,
- **Institute of Mathematics and Informatics, BAS**, Bulgaria,
- **Institute of Information Technologies, BAS**, Bulgaria.

Publisher: Institute of Information Theories and Applications FOI ITHEA, Sofia, 1000, P.O.B. 775, Bulgaria.
Издател: Институт по информационни теории и приложения ФОИ ИТЕА, София, 1000, п.к. 775, България
www.ithea.org, www.foibg.com, e-mail: info@foibg.com

General Sponsor: **Consortium FOI Bulgaria** (www.foibg.com).

Printed in Bulgaria

ISSN 1313-0455 (printed) **ISSN 1313-048X (online)** **ISSN 1313-0501 (CD/DVD)**

PREFACE

The scope of the International Book Series “Information Science and Computing” (**IBS ISC**) covers the area of Informatics and Computer Science. It is aimed to support growing collaboration between scientists from all over the world. IBS ISC is official publisher of the works of the members of the ITHEA International Scientific Society.

The official languages of the IBS ISC are English and Russian.

IBS ISC welcomes scientific papers and books connected with any information theory or its application.

IBS ISC rules for preparing the manuscripts are compulsory. The rules for the papers and books for IBS ISC are given on www.foibg.com/ibsisc. The camera-ready copy of the papers and books should be received by e-mail: info@foibg.com.

Responsibility for papers and books published in IBS ISC belongs to authors.

The Number 4 of the IBS ISC contains collection of papers concerning themes in the field of Software and Knowledge Engineering. Papers are peer reviewed and are selected from the several International Conferences, which were part of the Joint International Events of Informatics “ITA 2008”, Varna, Bulgaria.

ITA 2008 has been organized by

Institute of Information Theories and Applications FOI ITHEA

in collaboration with:

- ITHEA International Scientific Society
- International Journal “Information Theories and Applications”
- International Journal "Information Technologies and Knowledge"
- Association of Developers and Users of Intelligent Systems (Ukraine)
- Association for Development of the Information Society (Bulgaria)
- V.M.Glushkov Institute of Cybernetics of National Academy of Sciences of Ukraine
- Institute of Mathematics and Informatics, BAS (Bulgaria)
- Institute of Information Technologies, BAS (Bulgaria)
- Institute of Mathematics of SD RAN (Russia)
- Taras Shevchenko National University of Kiev (Ukraine)
- Universidad Politecnica de Madrid (Spain)
- BenGurion University (Israel)
- Rzeszow University of Technology (Poland)
- University of Calgary (Canada)
- University of Hasselt (Belgium)
- Kharkiv National University of Radio Electronics (Ukraine)
- Astrakhan State Technical University (Russia)
- Varna Free University “Chernorizets Hrabar” (Bulgaria)
- National Laboratory of Computer Virology, BAS (Bulgaria)
- Uzhgorod National University (Ukraine)
- Sofia University “Saint Kliment Ohridski” (Bulgaria)
- Technical University – Sofia (Bulgaria)
- New Bulgarian University (Bulgaria)

The main ITA 2008 events were:

KDS	XIVth International Conference "Knowledge - Dialogue – Solution"
i.Tech	Sixth International Conference "Information Research and Applications"
MeL	Third International Conference "Modern (e-) Learning"
ISK	Second International Scientific Conference "Informatics in the Scientific Knowledge"
INFOS	International Conference "Intelligent Information and Engineering Systems"
GIT	Sixth International Workshop on General Information Theory
CS	Third International Workshop "Cyber Security"
eM&BI	Second International Workshop "e-Management & Business Intelligence"
IMU ICT	International Seminar "Information Models' Utility in Information and Communication Technologies"
ISSI	Second International Summer School on Informatics

More information about ITA 2008 International Conferences is given at the www.foibg.com.

The great success of ITHEA International Journals, International Book Series and International Conferences belongs to the whole of the ITHEA International Scientific Society.

We express our thanks to all authors, editors and collaborators who had developed and supported the International Book Series "Information Science and Computing".

General Sponsor of IBS ISC is the **Consortium FOI Bulgaria** (www.foibg.com).

Sofia, June 2008 *Kr. Markov, Kr. Ivanova, I. Mitov*

TABLE OF CONTENTS

Papers in Russian

INDEX OF AUTHORS

INTELLIGENT SYSTEM FOR COMPUTER AIDED ASSEMBLY PROCESS PLANNING

Galina Setlak

***Abstract**: This paper presents the concepts of the intelligent system for aiding of the module assembly technology. The first part of this paper presents a project of intelligent support system for computer aided assembly process planning. The second part includes a coincidence description of the chosen aspects of implementation of this intelligent system using technologies of artificial intelligence (artificial neural networks, fuzzy logic, expert systems and genetic algorithms).*

***Keywords**: Artificial intelligence, flexible assembly systems, neural networks, fuzzy logic, fuzzy neural networks, group technology formatting rules.*

***ACM Classification Keywords**: I. Computing Methodologies, I.2 Artificial Intelligence, J. Computer Applications, J.6.Computer Aided Engineering*

***Conference**: The paper is selected from International Conference "Intelligent Information and Engineering Systems" INFOS 2008, Varna, Bulgaria, June-July 2008*

Introduction

The planning of the modern flexible manufacturing systems including the Flexible Assembly Systems (FAS) is a very complicated and responsible task. It assumed that the FAS are universal enough to be able to connect a high production capacity with the small quantities of production lots and short cycle time. It should ensure a production under the conditions of dynamical and sudden changes of the product range, the planed fixed dates for order realization and also the possibility of fast introducing of product design change into production [Boothroyd, Knight W., 1994]. According to the opinions of many assembling specialists the module assembly engineering is the fundamental and most promising direction of the development of the modern assembly technology [Grabmeier J., A. Rudolph, 2002], [Szabajkowicz, 1998]. The module technology is based on rules of the group production technology, which dominated in last dozen or so years, it improves and develops it [Szabajkowicz, 1998]. In addition, the module assembly technology enables a production adjusting according to market requirements, an easy adjusting oft the assembly system to every change of the product design, adding new engineering assembly modules. The planning of the flexible assembly modules in accordance with the modular engineering is the all-important problem at the centers for research and science as well as in design offices of the leading production companies in the last years.

This work presents the concepts of the intelligent system Computer Aided Assembly Process Planning (CAAPP) [Setlak,1999] for aiding of the module assembly technology planning. The CAAPP was developed in order to aid the decision making in the designing and functioning of the flexible assembly systems. In order to fulfill the identification and clustering tasks for product, parts and assembly unit groups, additional program modules are used, which include neural networks (Kohonen and Fuzzy Kohonen clustering network). In this paper, we handle decision making as a classification problem where an input pattern is classified as one of given classes. In this paper, neural networks are used as classification systems, which eventually could be implemented as a part of decision making systems. The second part includes a coincidence description of the chosen aspects of implementation of this intelligent system using technologies of artificial intelligence (artificial neural networks, fuzzy logic, expert systems and genetic algorithms).

The approach a problem of the modules assembly technologies

The module assembling engineering consists in presenting of the production process as a set of technological modules. The technological module is considered as a structurally closed part of the processing, which conforms

to the functionality, integrity and universality requirements. The module assembly means, that the assembly system has a modular structure and each module realizes a defined function or a limited function range, which are part of a general assembly process. According to the definition [Szabajkowicz, 1998] a technological assembly model composes "an integral set of the main and auxiliary activities of assembling, which are realized in a defined sequence at one station and uses a defined tool set for connecting of surfaces, parts, subassemblies, assemblies". The connection of the elementary technological modules lies in a proper development and selection of technological modules. Each of them realizes a proper design module of construction.

During the planning of the flexible assembly systems with the modular assembly engineering the following stages can be selected:

- Analysis of the construction of the assembled product and the assembling technologies.
- Identification and classification of objects into groups and subgroups of the processed parts and (technological similar) assembly sets. The working out of a typical flow chart (based on common assembly sequences, similar to the manipulation activities, duration, etc.).
- Separation of autonomic, integrated assembly activities from the flow charts, then assembling the separated assembly units into groups depending on equipment with instrumentation to carry out these operations.
- Planning of structures and functions of the constructional modules.
- Preliminary planning of elementary technological modules.
- Assembling of the elementary modules and selection of proper, possible variants of the technological and constructional modules.
- Optimization of the technological module structure and the structure of the constructional module realized.
- Clustering of the elementary technological assembly modules.
- Final planning of the technological assembly modules, the modular technological complexes and of the corresponding constructional modules.

The analyses of the construction of the assembled product concerns first off all the analysis of a producibility for the product construction, which is in the present generally made using the DFA methodology (Design For Assembly). The analysis of the producibility for a construction must be carried out in order to simplify the product constructions, reducing the part forms and subassemblies number. The questions concerning the producibility of product constructions assembled automatically were investigated among other in [Boothroyd, Knight W., 1994], [Łunarski J.,1993]. In these works the fundamental quality and quantity characteristics for producibility of product constructions for automatically assembling are presented (these are such features, as: interchangeability, regulation possibility, easy controlling and tool accessibility etc.). Planning products for assembly using the modular technology the constructional product modularization principle is to be kept. That means that by planning of units, subassemblies and parts following steps must be taken:

- Identification, separation of parts and basic surfaces;
- Use of typical assembly diagrams and methods;
- Aspiration to adjust a new product to such a construction, that the existing constructional modules and technological modules can be used.

By working out the expert system for modular assembly aiding system planning the necessity of integration of the constructional planning process with the processing planning was taken into consideration in order to utilize better the existing production equipment and eventually expansion or modernizing of it.

Conception structure of expert system for aiding of planning of the flexible assembly modules

The intelligent system Computer Aided Assembly Process Planning (CAAPP) outlined in this paper uses PC-Shell 4.0. – domain independent expert system shell, having strong hybrid properties. The PC-Shell has been implemented in Artificial Intelligence Laboratory (AITECH, Katowice) [Michalik, 1996]. The PC-Shell 4.0 system

integrates the expert systems shell using blackboard architecture elements and the simulator of the neural network. It assures the knowledge representation as declarative expressed rules, facts and distributing knowledge in the neural network. The expert knowledge can contain in some knowledge sources. A conception model of the expert system for aiding of the flexible assembly module planning is shown in the Figure 1.

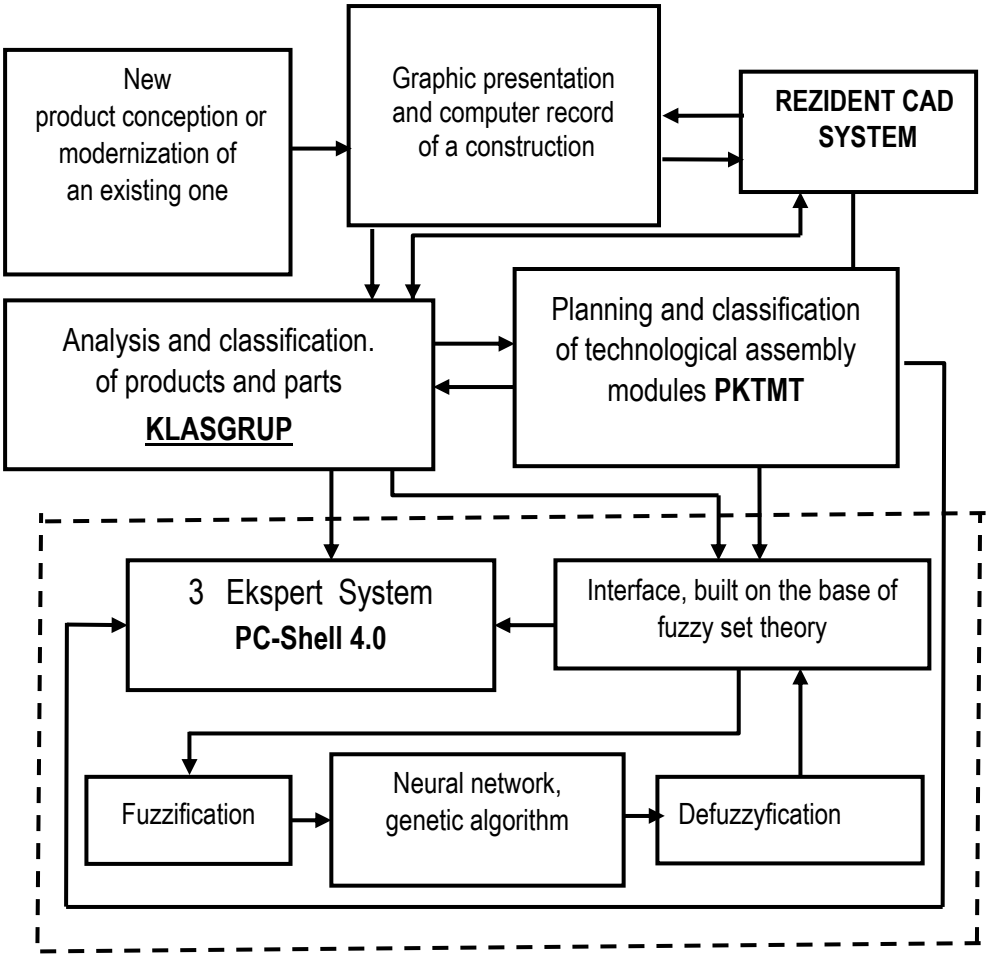


Fig.1. Conception model of a hybrid expert system.

Hybrid expert systems PC-Shell 4.0, as others classical expert systems, are built upon fundamental components: ♦ a knowledge base, ♦ an inference engine (interpreting knowledge stored in the knowledge base and making deductions), ♦ knowledge engineering system, ♦ automatic knowledge acquisition, ♦ explanation subsystem, ♦ user interface – one for accessing the knowledge base through the knowledge acquisition module, and another one for system users accessing the system in the consultation mode or in the explanation (tutor) mode and additional component part – the neural network.

The PC-Shell 4.0 system integrates the expert systems shell using blackboard architecture elements and the simulator of the neural network. It assures the knowledge representation as declarative expressed rules, facts and distributing knowledge in the neural network. The expert knowledge can contain in some knowledge sources. This system enables procedural knowledge representation too. The PC-Shell 4.0 system makes possible an integration of the declarative knowledge representation with procedural knowledge representation language, which enables programming of the system activity. The knowledge representation language SPHINX is a mean for building intelligent applications. It is the way of integration of particular artificial intelligence systems. The program in the PC-Shell system consists of instruction set included in the control block. The subset of language instruction enables the integration of neural network and the expert system.

The realization of application in the PC-Shell system is following:

- Creating by the NEURONIX subsystem one or some neural application.
- Elaborating knowledge base in form of knowledge sources.
- Integration of the elaborated knowledge sources on the level of knowledge representation language.

In this context the cooperation between systems usually follows by data interchange. Each of the subsystems realizes specifying purposes. It works by autonomous way and transmits results of its activity to the other system. Especially spectacular and also practically useful are results of expert system and neural network integration. We can describe following examples of their cooperation [Michalik, 1996]:

- the neural network realizes numeric data processing for the expert system ;
- expert system controls the learning process of neural network;
- the neural network is made for building knowledge base of the expert system;
- the expert system transforms the output neural network data in order to show other suitable for people interpretation.

A hybrid intelligent system has four major parts: PC-Shell 4.0 and Fuzzy Neural system (Neural networks, the Fuzzifier, the inference engine, the rule base, and the defuzzifier.

For aiding of the planning of the modular assembly technologies and to solve the problems of identification and classification of products groups, parts and units, the program modules have been developed, which complete the expert system CAAPP. These are the program module KLASGRUP and the module PKTMT. The program module KLASGRUP includes all procedures, which are necessary to carry out the constructional analysis of the planned or modernized product, and procedures to clustering the processed parts and (technological similar) assembly units of the mounted parts in order to separate and work out the constructional modules. The module PKTMT contains procedures for classification and grouping of the technological assembly modules. The details of the working out and structure of the expert system CAAPP, which technologically aids the assembly production preparation, are shown in the work [Setlak,1999].

To realize and test the program modules form aiding of the planning of the flexible assembly systems using the modular assembly engineering the knowledge base must be completed with following data:

- typical constructions;
- constructional features of the product parts;
- typical assembling flow charts and assembly methods;
- machinery data, technical equipment of the production system data;
- production costs for representatives of products from technologically similar groups.

In form of algorithms the constructional product analysis methods and assembly technology are formalized. The expert system CAAPP has been expanded by two additional modules; in addition a user interface has been introduced, which enables a presentation of a quality, verbal information in form of referring to adequate primary fuzzy sets. It enables a use of fuzzy inference engine in the program modules KLASGRUP and PKTMT the neural networks are used to classify the assembly parts and group the products.

Application of neural networks and hybrid intelligent systems for classification of assembly parts families

In the literature various classification methods have been proposed (see e.g. [Grabmeier J., A. Rudolph, 2002],). Some of them are based on neural networks, fuzzy systems and genetic algorithms (see e.g. [Zolghadri Jahromi, M. Taheri, 2008]). Neural networks are widely used as classifiers [Malave,1992], [Zurada,1992]. Classification and clustering problems has been addressed in many problems and by researchers in many disciplines like statistics, machine learning, data bases. The basic algorithms of the classification methods of machine elements

are presented in [Ed. by R. Knosala, 2002], [Ramachandran S., 1991], [Zolghadri Jahromi, M. Taheri, 2008]. The application of the clustering procedure can be classified into one of the following techniques [Grabmeier J., A. Rudolph, 2002]:

- graph -theoretic clustering,
- partitional in which a set is divided into *m* subsets, where *m* is the input parameter. These algorithms minimize the criterion function (*K*-means and *K*-medias);
- hierarchical form trees in which the leaves represent particular objects, and the nodes represent their groups. In terms of hierarchical methods, depending on the technique of creating hierarchy classes (agglomerative methods and divisive methods).
- fuzzy clustering,
- methods based on evolutionary methods,
- methods based on artificial neural networks.

In this work two approaches have been applied to clustering of parts and assembly units. The Kohonen self organizing map and fuzzy Kohonen neural network we have used in this work.

As basic method it was used Self Organizing Map (SOM), a class of unsupervised learning neural networks, to perform direct clustering of parts families and assembly units. SOM is an unsupervised neural network proposed by Kohonen [37] which consists of only two layers of neurons. Kohonen neural networks are unsupervised schemes which find the “best” set of weights for hard clusters in an iterative, sequential manner. This type of neural network is usually a two-dimensional lattice of neurons all of which have a reference model weight vector. SOM are very well suited to organize and visualize complex data in a two dimensional display, and by the same effect, to create abstractions or clusters of that data. The SOM can learn to recognize clusters of data, and can also relate similar classes to each other. SOM networks can also be used for classification when output classes are immediately available - the advantage in this case is their ability to highlight similarities between classes. SOM of Kohonen has been applied to classification of machine elements in group technology [Ed. by R. Knosala, 2002], [Setlak, 2003].

The other approach applies fuzzy logic and fuzzy neural systems for classification problems. Fuzzy and fuzzy neural systems can be employed in order to solve classification problems [Lin, 1996], [Nauck D.. R. Kruse, 1995].

In this work an application of the fuzzy version of the Kohonen self-organizing map network has been considered.

The fuzzy Kohonen clustering network integrates the concept of fuzzy c-means clustering technique into the learning rate and updating strategies of the Kohonen network [James C., Bezdek I., 1994]. It can be viewed as a Kohonen self-organizing type of fuzzy c-means, where the ”size” of the update neighborhood and learning rate in the competitive layer are automatically determined from training data. The cluster membership values of the input patterns are computed as a function of the distance between that pattern and the different cluster centers. These membership functions are used to determine the learning rates for the network weights updating.

The fuzzy Kohonen clustering network uses fuzzy membership values as learning rates, automatically extracted during learning from the training data. Also the adjustment of the update neighborhood is embedded in the learning procedure. Moreover, an increased number of fuzzy Kohonen network output nodes produce a better generalization performance of the modular classification system. The fuzzy Kohonen clustering network is shown in Figure 2. Combination of fuzzy c-mean and the Kohonen self-organizing feature maps was first considered by [Bezdek I., 1994] to make the Kohonen algorithm an optimization procedure. The algorithm combines the fuzzy membership values of the fuzzy c-mean for the learning and neighborhood size parameters and the update rules of the Kohonen feature maps.

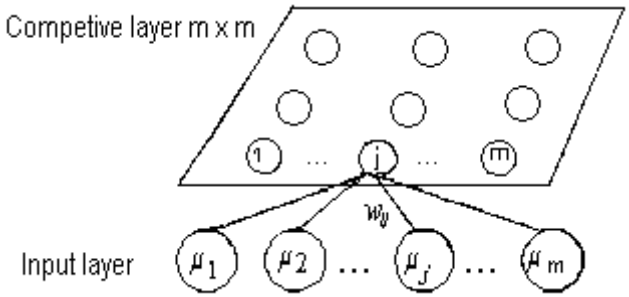


Figure 2. The structure of the fuzzy Kohonen clustering network

The update rule for the fuzzy Kohonen algorithm can be given as [Bezdek I., 1994]:

$$w_{i,t} = w_{i,t-1} + a_{ik,t}(x_k - w_{i,t-1}, \quad i =,12,...,c\,; k = 1,2,...,m \tag{1}$$

where $W_{i,t}$ is the centroid of the i^{th} cluster at iteration t , x_k is the k^{th} data point (or compound feature set) and α_{ik} is the only parameter of the algorithm and according to Huntsberger

$$a_{ik,t} = \left(U_{ik,t}\right)^{m(t)}, \tag{2}$$

where m(t) is an exponent like the fuzzification index in fuzzy c-mean and $U_{ik,t}$ is the membership value of the compound x_k to be part of cluster i. Both of these constant varies with each iteration t and given as:

$$U_{ik} = \frac{1}{\sum_{j=1}^{c} (D_{ik} / D_{jk})^{2/(n-1)}}, \quad 1 \le k \le n, \quad 1 \le i \le c \tag{3}$$

$$D_{ik} = (x_k - w_i)^T (x_k - w_i) \tag{4}$$

$$m(t) = m0 - \Delta m * t \tag{5}$$

$$\Delta m = (m0 - m_j)/max\ iter \tag{6}$$

where D_{ik} is the distance and m0 is some constant greater than the final value (m_f) of the fuzzification parameter m. The final value m_f should not be less than 1.1, in order to avoid the divide by zero error in equation 3.

In the examples below the presented algorithm have been used the method of geometrical description of the units of machine engines described in [Knosala,2002].

Geometrical features of structural elements were presented in the form of the matrix of properties. This method consists in exploiting geometrical primitive conditions which basic geometric features of similar are describing. Next made coding of geometrical features which consists in using wood is B-Rep method in order to receive the structure of the model in the three-dimensional space (3D). As a result of the division of the model of the element in three dimensions with the determined resolution to layers a matrix image of the element is received. The method of geometrical description of the units of machine is shown in figure 3.

The format of input data is being presented as follows:

<x> <y> <z>
<nr element> <nr layer > < the number of layers>
$<x_{11}>$ $<x_{12}>$ … $<x_{1n}>$
….
$<x_{n1}>$ $<x_{n2}>$ … $<x_{nn}>$

Where three first values means the resolution of the division of the 3D element into classes.

Grouped elements were written in the digital form at the 16x16x16 division in harmony with the accepted accuracy of the description of elements. The training data set includes 16x16X16 data items. The Fuzzy Kohonen neural networks composed of 16 neurons.

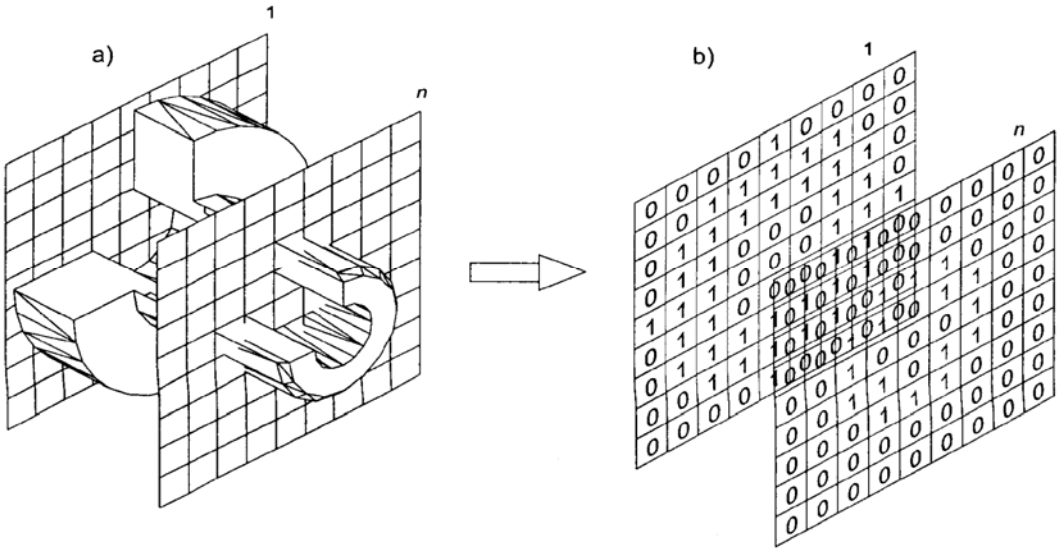


Figure 3. The method of geometrical description of the units of machine

Conclusions

The approach to the aiding of production systems planning based on the modular technology, proposed in this work, is a very promising direction for research on the field of the new production technologies. In the present the base problem at a practical realization of the presented expert system is lack of an access to data and work immensity, necessary to the pre-processing of the input data and to enter them into the knowledge base.

The way of integration of the neural network and the classical expert system presented in this paper is an interesting solution on the level knowledge representation language. Preliminary tests indicate that the approach employed in the PC-Shell system is capable of supporting problem formulation in ill-structured problem domains.

Future research in this work will be using the description of properties received as output date in program to the design CATIA. The research conducted proves that fuzzy Kohonen Neural Networks are a very effective and useful instrument of classification of the elementary assembly modules and can be employed in order to solve direct clustering of parts families.

Bibliography

[James C., Bezdek I., 1994] James C. Bezdek I, Eric Chen-Kuo Tsao and Nikhil R. Pal: Fuzzy Kohonen clustering networks, Pattern Recognition. Vol. 27, no. 5, pp. 757-764, 1994.

[Boothroyd, Knight W., 1994] Boothroyd G., Dewhurst P., Knight W.: Product design for manufacture and assembly, Marcel Dekker Inc., USA, 1994.

[Grabmeier J., A. Rudolph, 2002] Grabmeier J., A. Rudolph: Techniques of cluster algorithms in data mining, Data Mining and Knowledge Discovery, 2002, nr. 4, 303–360.

[Ed. by R. Knosala, 2002] Ed. by R. Knosala: Zastosowania metod sztucznej inteligencji w inżynierii produkcji, WNT, 2002, 456p.

[Kohonen,1990] Kohonen T.: Self-organizing Maps, Proc. IEEE, 1990, 78, NR.9, pp. 1464-1480.

[Lin, 1996] Lin, Chin-Teng and Lee, C.S. George. Neural Fuzzy Systems: A neural-fuzzy synergism to intelligent systems, New Jersey, Prentice-Hall, 1996.

[Łunarski J.,1993] Łunarski J., Szabajkowicz W.: Automatyzacja procesów technologicznych montażu maszyn, WNT, Warszawa,1993.

[Malave,1992] Malave C.O., Ramachandran S, Lee H: A self-organizing neural network approach for the design of cellular manufacturing systems //Journal of Intelligent Manufacturing, V.3, 1992, pp. 325–333.

[Michalik, 1996] K Michalik K.: PS-Shell dla Windows v.2.2- Przewodnik użytkownika, AITECH, Katowice, 1996.

[D. Nauck and R. Kruse,1995] D. Nauck and R. Kruse, NEFCLASS: A Neuro-Fuzzy Approach for the Classification of Data, In Proceedings of ACM Symposium on Applied Computing, George K et al (Eds.), Nashville, ACM Press, 1995, pp.-461-465.

[Ramachandran S., 1991] Ramachandran S, Malave C.: A neural network-based design of cellular manufacturing systems // Journal of Intelligent Manufacturing, V.2, 1991, pp. 305–314.

[Setlak,1999] Setlak G.: Hybrydowy system ekspertowy do projektowania procesów montażowych //Technologia I automatyzacja montażu, Warszawa, Nr.3, 1999, str. 23-27.

[Setlak,2003] G.Setlak: "Zastosowanie sieci neuronowych Kohonena w modułowej technologii montażu" // „Technologia i automatyzacja montażu", № 4, 2003, str.2-6.

[Szabajkowicz, 1998] Szabajkowicz W.: Modułowe technologie montażu // Technologia i automatyzacja montażu, Warszawa, Nr.4, 1998, str. 9-11.

[Zolghadri Jahromi, M. Taheri, 2008] M. Zolghadri Jahromi, M. Taheri, A proposed method for learning rule weights in fuzzy rule-based classification systems, Fuzzy Sets and Systems, V. 159 (2008), pp. 449 – 459

Authors' Information

Galina Setlak, *Ph.D., D.Sc, Eng., Associate Professor, Rzeszow University of Technology, Department Of Computer Science , Str. W. Pola 2 Rzeszow 35-959, Poland, Phone: (48-17)- 86-51-433, gsetlak@prz.edu.pl*

TRACEABILITY MANAGEMENT ARCHITECTURES SUPPORTING TOTAL TRACEABILITY IN THE CONTEXT OF SOFTWARE ENGINEERING

Héctor García, Eugenio Santos, Bruno Windels

***Abstract**: In the area of Software Engineering, traceability is defined as the capability to track requirements, their evolution and transformation in different components related to engineering process, as well as the management of the relationships between those components. However the current state of the art in traceability does not keep in mind many of the elements that compose a product, specially those created before requirements arise, nor the appropriated use of traceability to manage the knowledge underlying in order to be handled by other organizational or engineering processes. In this work we describe the architecture of a reference model that establishes a set of definitions, processes and models which allow a proper management of traceability and further uses of it, in a wider context than the one related to software development.*

***Keywords**: Traceability, Software Architectures, Configuration Management, Software Maintenance.*

***ACM Classification Keywords**: D.2.7: Distribution, Maintenance and Enhancement – Documentation; D.2.9: Management - Software configuration management*

***Conference**: The paper is selected from Sixth International Conference on Information Research and Applications – i.Tech 2008, Varna, Bulgaria, June-July 2008*

Introduction

The main goal in software traceability is to trace all the elements that can be considered relevant enough for the organization within a particular project or software product. Some classical examples of these elements are requirements, designs, source code or tests. However there is a number of information that is not considered carefully in current literature. Emails sent by stakeholders, minutes of meetings, project proposals or cost benefit analyses are documents that are also an essential part of a software product, retaining a great amount of knowledge that is required by organizations (e.g. in order to manage process improvement and capability determination [16]).

The capability to establish and maintain relationships between elements contained in these and other documents is essential, no matter the typology or stage during the product life cycle in which they arise. Managing all sort of relationships between whatever elements is what we define as total traceability. Some authors have previously discussed this term [14], although they have covered only aspects related to the engineering process.

In figure 1 below we illustrate the concept of total traceability that we try in our work. A software project does not start in the requirements engineering stage, as pointed by many authors [24],[29],[8],[20],[14]. Moreover, we should not consider requirements as the core in traceability, even when a wider perspective is considered, including information not directly related to engineering processes as described in [25].

We claim that software traceability should be focused in establishing relationships between the elements that compose the product, regardless of their type or stage in which they first appear. None of those elements should be considered the core or the start point in traceability. Only the specific purpose of the traceability itself, in a particular scenario, is to determine it.

Traceability should conform to an open and decentralized network in which we could include any element of particular interest in a given scenario.

Figure 1 shows a simplified hypothetic case, that we can easily find in the real world. The product life cycle starts when a user sends an email asking for some kind of software meeting some high level user requirements in order

to carry out with some tasks. After this email a project proposal or tender is prepared, covering those requirements, after a first approach to the problem. Later, a cost benefit analysis is developed, determining if it is viable to tackle the software. The solution to the problem of the user consists of developing specific software, carrying out a typical life cycle generating requirements specification, use cases definition, class diagrams and state charts, sequence diagrams and source code. In order to simplify the concept we avoided including a number of models and documents that can be considered, but in essence the main idea remains, all possible information, documents or models are candidates for tracing purpose.

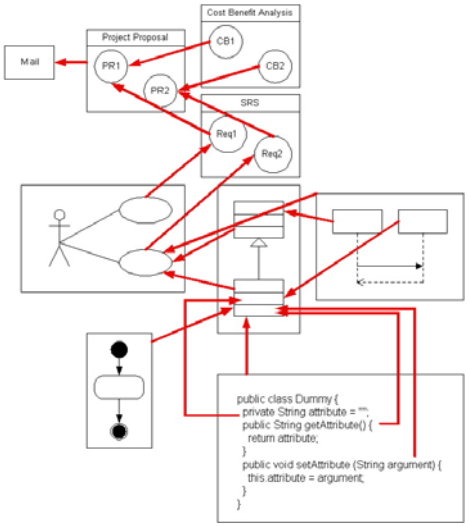


Figure 1 – Basic example on total traceability

Let us now imagine that, after carrying out the cost benefit analysis, the project is not developed because of the lack of resources. Then, considering a historical approach to traceability, all those emails and documents are not traced, so it could be potentially impossible to retrieve all knowledge underlying together with their relationships. What if we tackle the problem, again, after some time? Or if, also later, we decide to take it up again, using COTS? As long as the organization did not get to requirements engineering stage, no information on the project is available. How could we evaluate a number of commercial products or components? We should start everything from a scratch.

On the other hand, if we consider all these information, previous to requirements, we could trace our project proposal to the specifications given by vendors and take the project up again from the same point in which it was stopped.

Some of the documentation we are talking about is considered in [15] as essential to software life cycle, however nor this organization has treated these aspects with such a purpose, even in the new [18] or previously deployed [17], in which traceability is not considered as the essential part it is to software development.

May the reader appreciate that in some sense our work is not only related to software traceability, but also to knowledge management in the context of software development and maintenance, focusing traceability in the knowledge that documents, models or their relationships may provide in terms of the knowledge life cycle described by Birkinshaw and Sheenan [4].

To avoid misunderstandings, in this paper we will describe element as any item, under configuration management within an organization, and which may be related, or even not, to a software product in any sense. Also we will describe trace as any kind of traceability link in the widest semantics of the term.

In our work we considered the following hypotheses, which determine the context in which TRAMA is to be understood and that we discussed in [11]:

H1: The lowest level of granularity in traceability shall agree to the granularity established in Configuration Management.

H2: It shall be established a common framework regarding Configuration Management and Traceability from an organizational point of view, considering needs and goals of the organization.

H3: The products generated during the software life cycle can be modeled as structured and processable documents.

H4: Those tools used during all product life cycle and related tasks allow to make information persistent, in processable and structured formats, such as XML technology. In other case, such formats can be obtained through exports or digital processing.

Traceability Management Architecture

The basic components of the proposed architecture for traceability management systems are to support a set of operations that we shall enumerate below. We distinguish four main components: documentation and traceability data model support, traceability management, information retrieval and interfaces to external tools.

The data models define structures that allow making persistent all that information required to trace elements and managing traceability.

The traceability management component supports the basic operations allowed within traceability, using the data in the data models.

Information retrieval provides the capability to search and locate the information, as well as to infer new relationships and generate candidate lists.

Interfaces to external tools allow integrating the traceability management system to other applications used within the organization in order to automate the different tasks related, closely or not, to software projects.

In figure 2 below the different components of which TRAMA is composed, and some relationships to software life cycle processes [15], are shown.

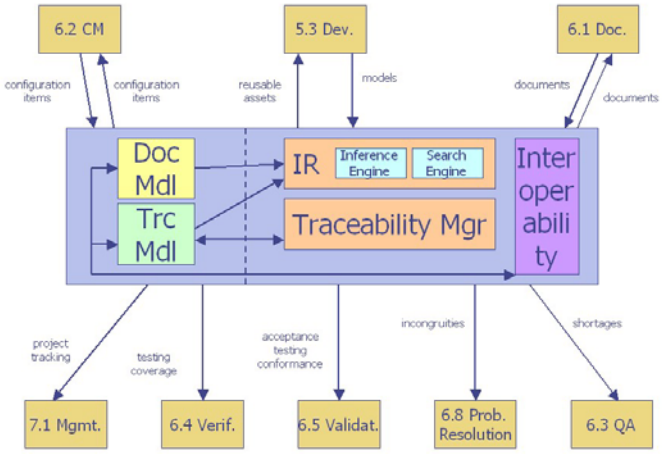


Figure 2 – Basic architecture for a traceability management system

Regarding data models we suggest to implement two different components, establishing proper relationships between them.

Documentation model is in charge of storing all information related to configuration items (e.g. projects, documents, sections of documents). Traceability model supports the storage of the traces established between configuration items.

Separating each model provides the capability of integrating configuration management systems from third parties. The purpose is to manage separately the information managed by external tools from the information managed by the traceability management system, which is the system our work is focused on.

In our implementation documentation model takes into account, amongst others, the following configuration items:

- Projects and project stages, as well as staff assigned to the different tasks. These data retain information about planning, allowing to generate project tracking reports.
- Documents and sections of documents represent a model of the documentation of a project and the structure of documents.
- Software artifacts, such as requirements and design, coding and testing artifacts represent the deliverables from engineering tasks in the software life cycle.
- Deliverable templates related to each task in a project support quality assurance and project tracking reports.
- Stakeholders involved in tasks and documents that shall receive the deliverables.

Meanwhile, the traceability model is in charge of storing the traces that establish the relationships between configuration items. A partial list of traces is described below.

- Dependency traces model the relationship between configuration items, where a configuration item is dependent on other. We provide a number of semantics for dependency (e.g. parent to child, aggregation, complement) as well as degrees for the dependency (e.g. full, partial).
- Document and section traces establish the contents of documents and their structure. Contents are always configuration items (e.g. software artifacts, text, figures).
- Rationale traces allow attaching an explanation to a configuration item.
- Output traces establish relationships between tasks and document templates.
- Drive, Implementation, Test and Verification traces allow attaching different software artifacts to other. Usually these traces are useful to generate quality assurance reports, as well as to support Change Management.

The information retrieval component is responsible of locating configuration items that match the criteria given by a user search. User search are defined as patterns that depend on the search type. We provide the following search types to support different goals:

- Locate individual configuration items matching a pattern, such a rationale, content or name.
- Locate configuration items involved in traces to a given configuration item.
- Locate configuration items involved in a specific kind of trace to a given configuration item. This is particularly useful to calculate impact trees derived from a change (i.e. to support change management).
- Locate software artifacts that satisfy a set of requirements. This feature has been introduced to support software reuse.

It is also feasible to infer knowledge from the information provided by traces. The main goal is to find new traces between configuration items, in order to reduce the effort of software engineers while carrying out traceability tasks. This feature is also important to find incongruence in the models of a given project, and to locate traceability lacks, as well as to support quality assurance tasks. These are the reasons why we defined two different components, Inference Engine and Search Engine.

Traceability Management component is in charge of carrying out the functionalities defined in the reference model, excepting information retrieval and interoperability. Some of these functionalities are:

- Create and delete repositories
- Create and delete projects
- Replicate elements and reuse assets
- Create, alter and delete traces
- Create, alter and delete documents and sections

Interoperability component supports the operations which allow integrating the traceability management system to external tools (e.g. CASE, configuration management or documentation tools). Information exchange is based on import and export features through XML and XMI files. An improvement of the features in which we are working on is to create web services. Web services could make feasible to share information in different tools in soft real time.

Related Work

Traceability usefulness, in the area of Software Engineering, has been demonstrated by most related literature. Processes such as change management obtain substantial benefits from traceability [10],[6]. Traceability links allow establishing relationships between different items, or knowledge assets, and they are of interest for the organizations [26]. The capability to reuse software assets, such as documents, models or code, is also a question closely related to traceability [7], as well as domain and product analysis [8].

However a high effort and cost is inherent to traceability management, as described in [9], and new perspectives on which elements are more relevant or require more detailed attention, in order to lower the effective costs are arising, such as the Value Based Software Engineering [5].

The most popular choice when automating traceability consists of the development of systems and frameworks stating clearly the information related to traceability links, and the way to implement them in a standardized manner, instead of depending on specific features provided by vendors.

Sherba et al. [28] provide a traceability management system consisting of the integration, through parsers, of different tools, sharing the project components and models in a common repository. The main problem is to maintain this duplicated and coupled structure. In [3] we can find some suggestions on how to avoid the problem, together with a compromise on the granularity or detail level required in this kind of systems.

More efforts in this sense have conducted to establish standardized formats for documents and models, most of them based in the eXtensible Markup Language, such as XML Metadata Interchange [23].

In [19] a metamodel for traceability management and a set of processes related to software traceability, based in patterns, can be found. Alarcón et al. [1] describe a software engineering environment, considering an integrated traceability system, in which documents generated within the environment are stored in an XML compatible format. Also many efforts in introducing tags within the source code, to provide traceability, have been described in earlier years [12].

Alves-Foss et al. [2] suggest the use of XMI to represent UML designs, and JavaML for the source code. A set of DTDs and transformations make it feasible to translate the models into source code and vice versa.

Another significant problem in traceability systems is to determine the proper information retrieval and processing. Huffman et al. [13] applied information retrieval techniques to create automated candidate traceability link lists. Marcus et al. [22] used latent semantic indexing to detect links between product documentation and source code, and Spanoudakis [29] established a set of heuristic rules to analyze links between different elements that resulted in patterns to determine which candidates where valid.

Regarding elements and relationships, the best analysis we can find is that one by Ramesh and Jarke [25], in which we can find a complete classification on traceability links and the data that should be considered in reference models. Different classifications, which provide more information and types of links can be found in [21], with the goal of supporting conformance analysis and inconsistency finding. In [20] we can find a model to support traceability management for UML projects, including rationales and stakeholders, as well as many software artifacts.

Tryggeseth and Nitro [30] classify the relationships in different categories keeping in mind a double structure related to application structure and documentation, while Riebisch [27] takes into account the link types depending on the structure of requirements documents. Von Knethen [31] establishes a difference between traces linking elements in the same abstraction level and those between elements in different abstraction level. Sherba et al. [28] describe some examples of links that are useful to determine some of the types of relationships between elements.

Relating the knowledge that should underlie to each link, Ramesh and Jarke [25] suggest to consider six dimensions: *What?, Who?, Where?, How?, Why?* y *When?*.

Conclusions

Introducing traceability as a part of the methodological definition of the development process could help in avoiding the historical problems of traceability application, marginally considered in the Software Engineering processes. The support to other organizational processes could result in decreasing the effective costs on applying traceability. The specification of ISO 24744 [18] could have been a nice chance to introduce traceability as an essential part during the definition of software development methodologies. Integrating traceability in such a standard, which considers all software process aspects, from documentation to tasks, as well as human resources involved, should have supposed a great step.

If we expect to reach such an ambitious goal, it is necessary to track any knowledge asset. It supposes to include, under configuration management, any element that shall persist in time. Then, it becomes necessary to extend configuration management to all areas in the organization, not only to those related to engineering processes. In this sense it is essential to establish a common framework in the organization regarding configuration management and traceability, considering also the proper minimum granularity required.

We also discussed how information contained in a traceability management system is not only useful for software processes, but also to other processes. Multiple uses of this information, especially through information retrieval and data mining, will result in long and short term benefits in organizations. In such case scenario it could be worthwhile to implement and introduce traceability management in the industry, as well as lowering the effort required to manage and maintain these kinds of repositories.

Bibliography

[1] Pedro P. Alarcon et al. Automated Integrated Support for Requirements-Area and Validation Processes Related to System Development. Proceedings of the 2nd International Conference on Industrial Informatics, IEEE. 2004, pp. 287-292.

[2] Jim Alves-Foss, Daniel Conte de Leon y Paul Oman. Experiments in the Use of XML to Enhance Traceability between Object-Oriented Design Specifications and Source Code. Proceedings of the 35th International Conference on System Sciences, pp. 3959-3966. IEEE. 2002.

[3] P. Arkley, P. Mason, S. Riddle. Position paper: Enabling Traceability. Proceedings of the 1st International Workshop on Traceability in Emerging Forms of Software Engineering, pp. 61–65. ACM. 2001.

[4] J. Birkinshaw, T. Sheenan. Managing the Knowledge Life Cycle. Engineering Management Review, volume 31, issue 3, p. 19. 2003.

[5] Barry Boehm. Value-Based Software Engineering. Software Engineering Notes, vol. 28, issue 2, section Article abstracts with full text on line, pp. 3. ACM. 2003.

[6] L.C. Briand, Y. Labiche, L. O'Sullivan. Impact Analysis and Change Management of UML Models. Proceedings of the 19th International Conference on Software Maintenance. IEEE. 2003.

[7] Andrea de Lucia et al. Enhancing an Artefact Management System with Traceability Recovery Features. Proceedings of the 20th International Conference on Software Maintenance, pp. 306-315. IEEE. 2004.

[8] Alexander Egyed, Paul Grünbacher. Automating Requirements Traceability: Beyond the Record & Replay Paradigm. Proceedings of the 17th International Conference on Automated Software Engineering, pp. 163-171. IEEE. 2002.

[9] Alexander Egyed, et al. A Value-Based Approach for Understanding Cost-Benefit Trade-Offs During Automated Software Traceability. Proceedings of the 3rd International Workshop Traceability in Emerging Forms of Software Engineering, pp. 2-7. ACM. 2005.

[10] Stephen G. Eick et al. Does code decay? Assesing the Evidence from change management data. IEEE Transactions on Software Engineering, vol. 27, no. 1. IEEE. 2001.

[11] Hector Garcia. Documentation and Traceability in Software Projects (in spanish). Research Work. Carlos III University of Madrid. 2007.

[12] Ernesto Guerrieri. Software Document Reuse with XML. Proceedings of the 5th International Conference on Software Reuse, pp. 246-254. IEEE. 1998.

[13] Jane Huffman Hayes, Alex Dekhtyar, James Osborne. Improving requirements tracing via information retrieval. Proceedings of the 11th International Conference on Requirements Engineering Conference, pp. 138-147.IEEE. 2003.

[14] Suhaimi Ibrahim et al. Implementing a Document-Based Requirements Traceability: A Case Study. Proceedings of the 9th International Conference on Software Engineering and Applications, pp. 124-131. IASTED. 2005.

[15] International Organization for Standardization. Information technology – Software life cycle processes. ISO/IEC 12207:1995. ISO. 1995.

[16] International Organization for Standardization. Software Process Improvement and Capability Determination. Juego de normas ISO/IEC 15504. ISO. 2003 and 2004.

[17] International Organization for Standardization. Information technology – Software Engineering Environment Services. ISO/IEC 15940:2006. ISO. 2006.

[18] International Organization for Standardization. Software Engineering – Metamodel for Development Methodologies. ISO/IEC 24744:2007. ISO. 2007.

[19] Justin Kelleher. A Reusable Traceability Framework using Patterns. Proceedings of the 3rd International Workshop in Emerging Forms of Software Engineering, pp. 50-55. ACM. 2005.

[20] Patricio Letelier. A Framework for Requirements Traceability in UML-based Projects. Proceedings of the1st International Workshop on Traceability in Emerging Forms of Software Engineering, pp. 32-41. ACM. 2001.

[21] Jonathan I. Maletic et al. Using a Hypertext Model for Traceability Link Conformace Analysis. Proceedings of the 2nd International Workshop Traceability in Emerging Forms of Software Engineering, pp. 47-54. ACM. 2003.

[22] Andrian Marcus, Jonathan I. Maletic. Recovering documentation-to-source-code traceability links using latent semantic indexing. Proceedings of the 25th International Conference on Software Engineering, pp. 125-135. IEEE. 2003.

[23] Object Management Group. MOF 2.0 – XMI Mapping Specification, v. 2.1. OMG. 2005. Available at <http://www.omg.org/technology/documents/formal/xmi.htm> [ref. february 11th, 2007].

[24] Balasubramaniam Ramesh. Factors Influencing Requirements Traceability Practice. Communications of the ACM, vol. 41, issue 12, pp. 37-44. ACM. 1998.

[25] Balasubramaniam Ramesh, Matthias Jarke. Towards Reference Models for Requirements Traceability. IEEE Transactions on Software Engineering, vol. 27, issue 1, pp. 58-93. IEEE. 2001.

[26] Balasubramaniam Ramesh. Process Knowledge Management with Traceability. IEEE Software, vol. 19, issue 3, pp. 50-52. IEEE. 2002.

[27] Matthias Riebisch. Supporting Evolutionary Development by Feature Models and Traceability Links. Proceedings of the 11th IEEE International Conference and Workshop on the Engineering of Computer-Based Systems, pp. 370-377. IEEE. 2004.

[28] Susanna E. Sherba, Kenneth M. Anderson, Maha Faisal. A Framework for Mapping Traceability Relationships. Proceedings of the 2nd International Workshop on Traceability in Emerging Forms of Software Engineering, pp. 32-39. ACM. 2003.

[29] George Spanoudakis. Plausible and Adaptative Requirements Traceability Structures. Proceedings of the 14th International Conference on Software Engineering and Knowledge Engineering, pp. 135-142. ACM. 2002.

[30] Eirik Tryggeseth, Oystein Nitro. Dynamic Traceability Links Supported by a System Architecture Definition. Proceedings of the International Conference on Software Maintenance, pp. 180-187. IEEE. 1997.

[31] Antje von Knethen. A Trace Model for System Requirements Changes on Embedded Systems. Proceedings of the 4th International Workshop on Principles of Software Evolution, pp. 17-26. ACM. 2001.

Authors' Information

Héctor García – *Adjunct Professor. Technical University of Madrid. E.U. Informática. Ctra. de Valencia Km. 7. E28031 Madrid. e-mail: hgarcia@eui.upm.es*

Eugenio Santos – *Professor. Technical University of Madrid. E.U. Informática. Ctra. de Valencia Km. 7. E28031 Madrid. e-mail: esantos@eui.upm.es*

Bruno Windels – *Researcher. Technical University of Madrid. E.U. Informática. Ctra. de Valencia Km. 7. E28031 Madrid. e-mail: bwindels@eui.upm.es*

INCREASING RELIABILITY AND IMPROVING THE PROCESS OF ACCUMULATOR CHARGING BASED ON THE DEVELOPMENT OF PCGRAPH APPLICATION

Andrzej Smykla

***Abstract**: The article presents the software written in Builder C++ that monitors the process of processor impulse charger. Protocol, interface, components used and the future research are presented..*

***Keywords**: PCGraph, developing software, charging process, C++ Builder*

***ACM Classification Keywords**: C.3 Special-Purpose and Application-based Systems*

***Conference**: The paper is selected from Sixth International Conference on Information Research and Applications – i.Tech 2008, Varna, Bulgaria, June-July 2008*

Introduction

The article presents the software written in Builder C++ language that monitors the process of charging through COM port. Pulsar is impulse charger made by Elprog. It is a professional fast impulse charger to charge all kinds of cells available on the market. The product won the prestigious prize "Polish Market" in 2004. Among its qualities are:

1. the speed of charging
2. the reliability of the charger process
3. the regeneration of old cells
4. the monitoring of charging process.

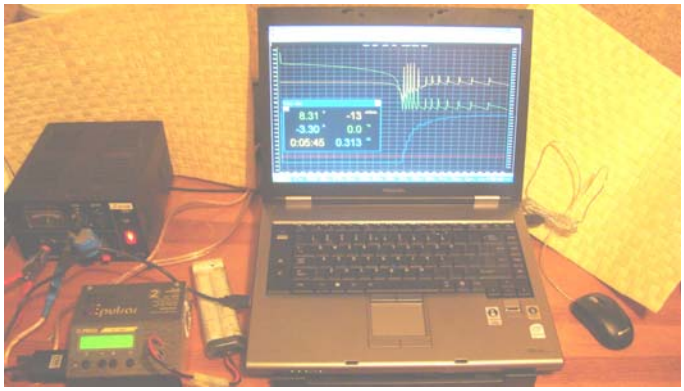


Fig. 1. The charger system during work

Computer program that read the data directly from charger is a useful tool to analyze the quality of aku pack.

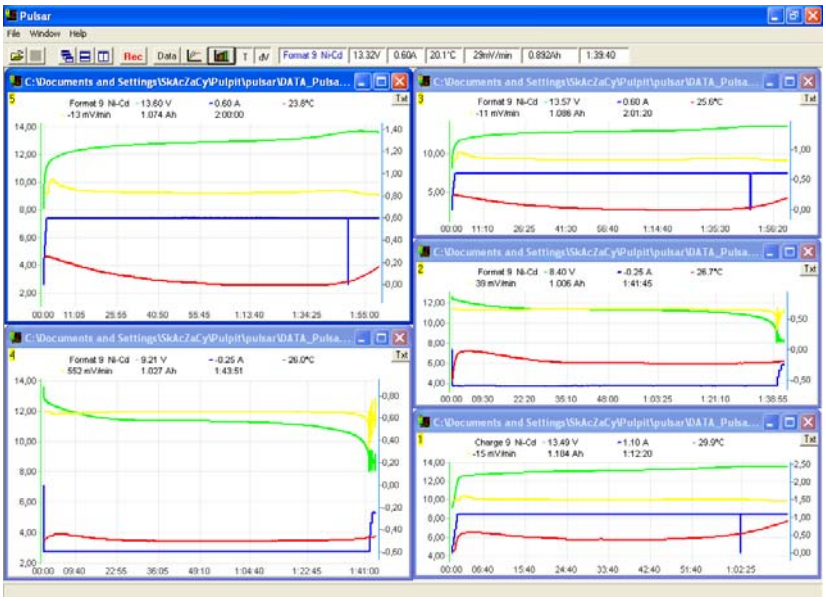


Fig. 2 The graphical interface of PCGraph program
(the colors determine: blue – amperes, red – temperature, green – volts, yellow – dV/dt)

The charger system and the graphical interface of PCGraph program is shown in Fig. 1. and Fig. 2.

The system is used especially when high reliability is required. The systems work for example in European Space Agency and Polish Polar Station (Spitsbergen).

The program is written in Builder C++ as an MDI application. Components used are presented in Figure 3.

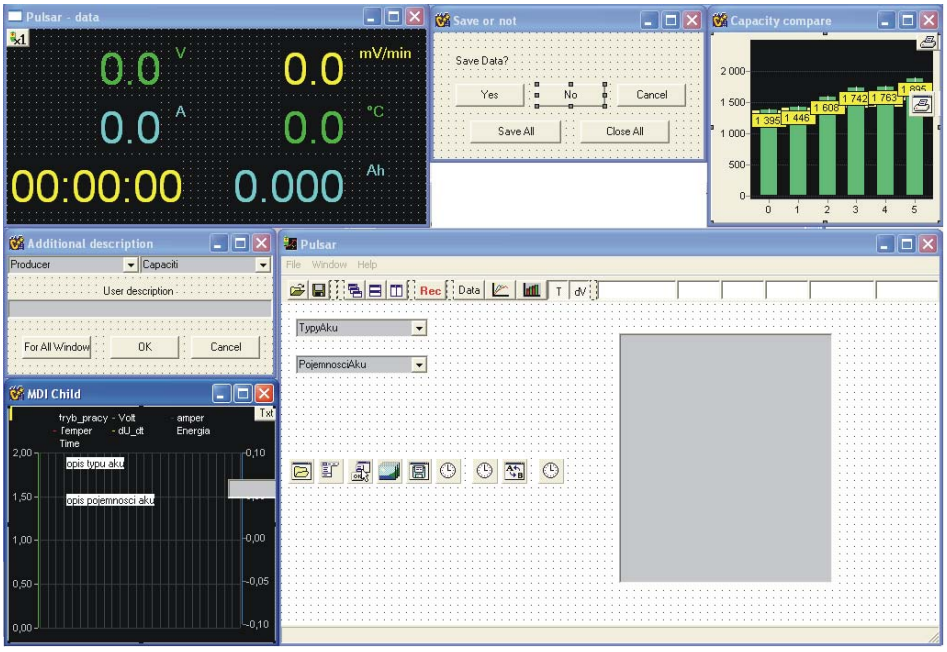


Fig. 3. The view of chosen forms with components used.

The data flows diagram between the most important components are presented in Figure 4.

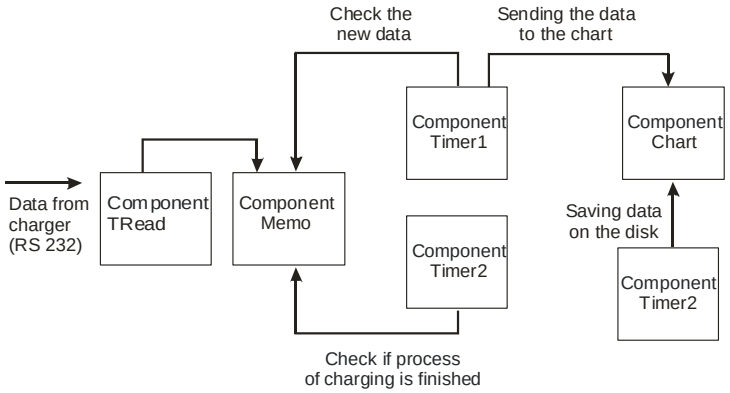


Fig. 4. Data flow diagram of applications.

A serial port is used for communication. The format of protocol (32 bytes long) is presented on Figure 5.

#C	3	5	D	00035,	12029,	+0199,	000,	00001
Begin frame	Number of cells	Cell type	Char Type	Time[s]	Volts[mV]	Curent[mV]	Temperature[°C]	Energy [mAh]

Fig. 5. The charger communications protocol data format

The type of calls determined the kind of process charging (Table 1)

The charge mode is described with letters – for their meaning see Table 2. The plus and minus sign before the ampere's value means charging and discharging.

Table 1

Type of calls	Kind of accumulator
1	Ni-Cd
2	Ni-MH
3	Pb-bat
4	RAM
5	Li-Ion
6	Li-Pol
7	Li-TA
8	Li-S

Table 2.

Charge/ Discharge mode	Charge mode
D	Discharge
S	CH. Simple
R	CH. Reflex
P	CH. Pb-bat
L	CH. Lith
C	Regen.
C	Charge
D	Disch
F	Format

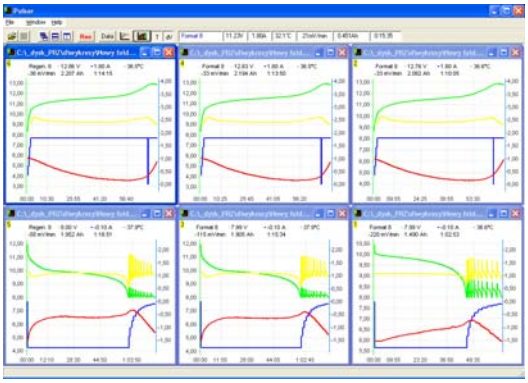
The reading and correct interpretation of charging and discharging process characteristics that are obtained enables us to determine the actual quality of accumulator. However, some experience in reading the characteristics obtained is required. A demonstration of characteristic received with one weaker cell is shown in Figure 6.

a) the characteristic of charging process in which after 22 minutes and 30 seconds there was a volt decrease. This situation means there is one weaker cell in the package

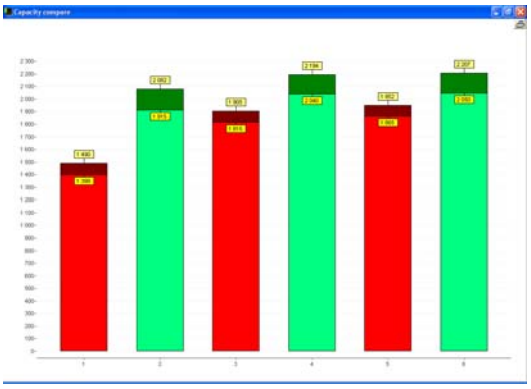
b) the example of charging a cell that was completely discharged

Fig. 6. The examples of packages of accumulators' defects characteristic

The program presented enables the recording of several discharging/recharging cycles and to present the data in graph form. The possibility of such analysis is valuable during the package's regeneration as well as during the determination of its consumption in time (see Figure 7).



a) the graph of separate cycles



b) the graph comparing capacity

Fig. 7. A three-cycle discharging/recharging process in which the capacity of an accumulator package was raised from 1400mAh to 1900 mAh

The monitoring program that collects data and presents them in the form of graphs is therefore extremely useful to all that increases the functionality of the charger. The reading and proper interpretation of graphs is possible only after gaining some essential skill by the user which often requires some time and effort.

The Current Research

Research on the process of creation the expert system that would make reading graphs not essential for the user is being conducted. Initial analyses using the artificial neural network were satisfactory. The network (for the scheme see Figure 8) was prepared as an expert system stating whether the discharging process of an accumulator package finished successfully.

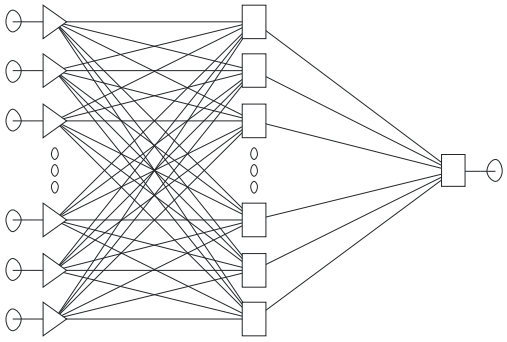


Fig. 8. The scheme of neural network used to identify the fault of accumulator packages.

The data received during the charging process was normalized in table of volting level $[U_1,U_2,..,U_n]$. The output vector consisted of two elements (the correct package and the wrong package). Coincidence of the network's learning process was observed during the experiment. The modeling of neural network was made in STATISTICA NEURAL NETWORKS system. At this stage collecting more actual data including information about the charging process is crucial. In order to achieve these purposes the current program version includes the record of extra information about the package.

The Future

It is probable that the next program version would enable sending the collected characteristics directly to the network server that contains a database. Building an internet server including the expert system is another aim that should be possible to achieve. It would be attainable for the user to check whether their package is correct using the current knowledge of server's expert system and the data could increase sources available. Several possible scenarios of development are shown in Figure 9.

The possibility of building evaluation system that would not only find the package faults but also indicate the type of damage, e.g. damage of a single cell (or a few cells), complete discharging of one cell in the package, wrong charging method used (e.g. Ni-Cd accumulators were charged as Ni-MH), wear–out of the package etc should be examined during future research. The significance of such research is emphasized by the fact that the increase of the number of mobile devices was observed in recent years and therefore there will be more demand for

evaluating and repair of accumulator. The charger is also currently utilized by certain services such as police, fire brigades where the reliability of technical devices is of great importance. The use of data-mining for huge amount of data might result in development of routine accumulator exchange or inspection standards. The experiments indicate that the chargers delivered with the equipment often cannot make full use of the device. Also they can damage the accumulator packages even though they were destined for the concrete type of equipment as it often happens in different kinds of inexpensive mobile tools. The user exchanges the devices or accumulator packed which causes the increase of their number on garbage dumps and causes the pollution of environment. Therefore, the conduction of this research and further development of software presented seems to be appropriate.

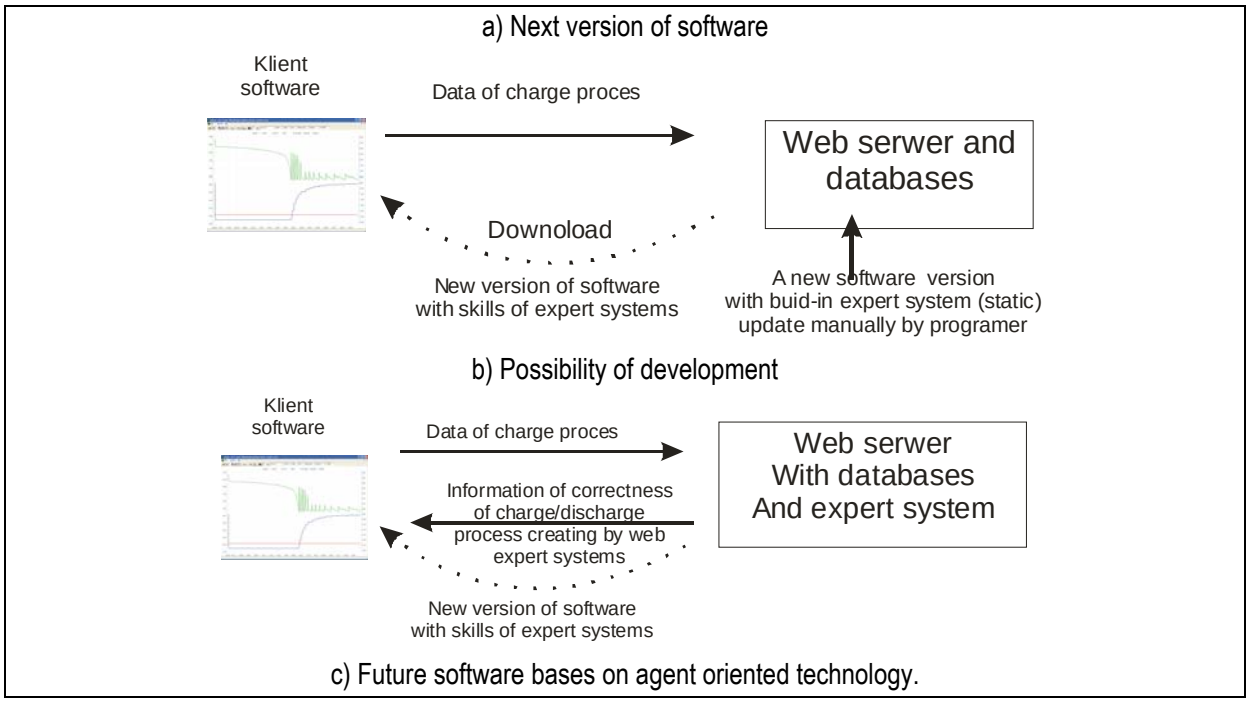


Fig. 9. The strategy of development of the software presented

Conclusion

1. The use of application to analyze the charger mode expands the device's possibilities.
2. The analysis of characteristic obtained requires experience in their interpretation.
3. The use of expert system to analyzing chosen signals will be possible in the future.
4. The creation of databases including the characteristics' history during the usage of separate models of accumulators might make it possible to forecast the wear–out of packages. In older to do that an analysis using data-mining should be performed. The current program version includes the mechanism of accumulator description suitable for such research.

The creation of internet service collecting data from the user that might be automatically integrated with the program for expert system learning and data-mining seems appropriate.

Author's Information

Andrzej Smykla – *Rzeszow Univesity of technology; Departure of Computer Science –,ul. W. Pola 2 Rzeszow 35-959, Poland; e-mail: asmykla@prz.edu.pl*

CORE DESIGN PATTERN FOR EFFICIENT MULTI-AGENT ARCHITECTURE

Kasper Hallenborg

***Abstract**: Interaction engineering is fundamental for agent based systems. In this paper we will present a design pattern for the core of a multi-agent platform - the message communication and behavior activation mechanisms - using language features of C#. An agent platform is developed based on the pattern structure, which is legitimated through experiences of using JADE in real applications. Results of the communication model are compared against the popular JADE platform.*

***Keywords**: multi-agents, design pattern, C# language features, message based architecture, behaviors.*

***ACM Classification Keywords**: D.2.11 Software Architectures, D.2.13 Reusable Software.*

***Conference**: The paper is selected from International Conference "Intelligent Information and Engineering Systems" INFOS 2008, Varna, Bulgaria, June-July 2008*

Introduction

Multi-agent systems, which also are referred to as *Distributed Artificial Intelligence* (DAI) [Weiss, 1999], are naturally expected to act and collaborate in a distributed environment and across different hosts. Thus from the beginning agent platforms have focused on supporting communication between agents across networks.

FIPA has proposed an abstract architecture for agent organization, and the dominating part focuses on message transport and agent communication. These specifications concern different levels of the communication, from the low-level message transport and communication protocols to the higher level abstract speech act theories for the message content. An increasing number of agent platforms try to comply with these specifications, and we find some of the most popular agent platforms, like JADE and FIPA-OS, in this category, given that they aim to support all kind of multi-agent systems.

Particularly in control systems, but for many other applications as well, the agents are just virtual representations of real entities in the application environment. The agents may have a simple bound communication channel to their physical entity, or the commands of the control unit are sent to PLCs or robots in the production environment, which performs the real actions. Thus in many situations, there is no need for advanced network communication support, as the agents are running in the same execution environment, often also in shared memory. Making just a few assumptions we can boost the performance of messages handling between agents by eliminating the overhead off network communication. Many existing multi-agent platforms try to avoid this overhead by grouping agents running on the same machine, but the abstract and general message envelopes still impact the scalability of the platform, when the communication increase.

In this paper we will present a design pattern and implementation details for a backbone to a multi-agent platform. The presented code listings for the implemented pattern is coded in C# taking advantage of language features that are not directly supported in Java. Most open-source platforms are Java based, but a few .Net based platforms are now available, such as MAPNET, EtherYatri.Net, Agent-Service and CAPNET. The latter two also being FIPA compliant, but Java is still being the dominating language for the open-source community.

In the next section we will elaborate on how JADE, as one of the most popular agent platforms, handles communication and messages.

Related Work and Motivation

Advanced interaction mechanisms are what really distinguish multi-agent platforms from general distributed architectures. Thus, a key feature of multi-agent systems has always been the mechanisms that allow the agents able to make intelligent decisions about their interactions and can participate in interactions that not necessarily were foreseen at design time [Jennings, 2000].

Many agent platforms implement this high level of abstraction by using an agent communication language (ACL) based on the speech act theory, originally introduced by Searle [Searle, 1969]. Thus most agent platforms are based on some kind of message-based interaction, where the message format allows abstract content to be encoded. FIPA-ACL specified by FIPA seems to be the most commonly used in new agent platforms [FIPA, 2002].

Aiming for more cognitive and deliberating agents more advanced interaction frameworks have been build on top of the message languages, such as goal-directed interactions [Cheong, 2006][Braubach, 2007] inspired by the BDI architecture, or role-based interactions [Cabri, 2006], but has not matured yet. Many agent platforms are mostly concerned about the basic message handling, message format, and encoding/decoding.

The JADE agent platform from Telecom Italia Lab [Bellifemine, 2005] is among the most popular agents platform available today, and it is rather generic in how it handles interactions, and we had extensive experience with it, from implementing control a baggage handling system in an airport in Asia [Hallenborg, 2006]. Thus, JADE is an appropriate candidate for explaining the message handling principle of agent frameworks. Our experience with JADE for real life applications was also that for many application domains, the rather abstract handling of message semantics was not required at all, even though you still want the agent mechanisms. This is specially the case for manufacturing systems and other agent-based systems, which are less spontaneous and more or less all participants in interactions are known at design time, at least from their characteristics and capabilities.

Similar to other agent platforms JADE has en rather simple and intuitive communication model based on asynchronous message passing. In JADE each agent lives inside a container, and a JADE platform refers to such a set of distributed containers, though several containers are allow on the same machine.

As illustrated in figure 1 each agent has a sort of mailbox – a message queue, where received messages are dropped and the agent is notified, but the receiver is in full control of if; when, and how to react to a new message. The common approach of using JADE specifies that each agent has a number of attached behaviors, which are activated if the message template matches an incoming message. After executing the action of the behavior, the behavior object will either be detached or start waiting for new incoming messages, depending on its configuration.

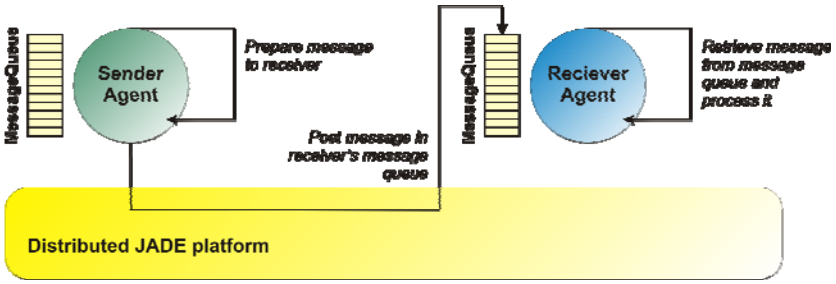


Figure 1: Communication model for the JADE platform

Our experience of applying JADE to this large real life application left no complaints about the architecture and communication model of JADE. The serious performance problems were due to the internal handling of messages inside JADE during the message passing process. Running a profiler on the implemented system everything pointed at the *send* method of the Agent class as the performance leak. Digging into the source of JADE the problem could be revealed from an extensive set of tasks being executed every time a message is sent.

In the short version the *send* method of *Agent* delegates the call to the *handleSend* method of *AgentContainer-Impl* class that always requires a clone of the message[1], not only for a message with multiple receivers, but for a single receiver as well. Besides making a deep clone of the message object, which include several byte array copies of the many string fields in a message, the message is wrapped in a generic command object that is processed after credentials of the sender is set in the message. The processing step includes a number of filtering steps of the message, which perform some extensive checking of the message before it is actually sent to the receiver. The extensive copying and processing is just one part of the problem, and usually irrelevant if the communication is low, but it scales very badly when the communication level increase. At the bottom this is due to numerous of synchronized methods in the filtering and processing steps, which slows the execution caused by massive scheduling.

All this is done even if both sender and receiver is within the same container in the same executive space, where message handling should be as simple as moving a memory reference, under just a few assumptions. And this performance flaw has nothing to do with the semantics or abstraction level of the content being encoded into the message, the content could be a simple as an integer and the procedure would still be the same.

For the real life applications we are working with, such as the baggage handling problem, the performance overhead in the communication model kill all advanced decision logic of the agents, and even with some modifications to the JADE source implementation, it has become evident that a much simpler approach should be pursued.

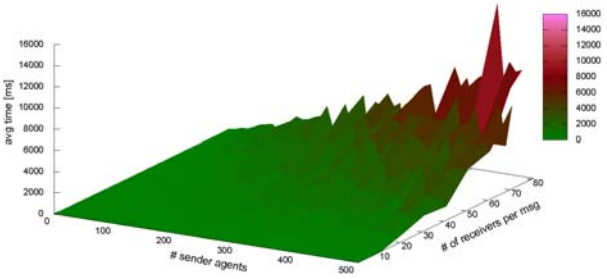


Figure 2: Average message sending time – many agents

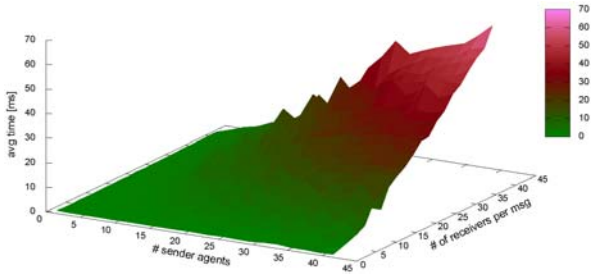


Figure 3: Average message sending time - few agents

Figure 2 and 3 show how JADE scales with the communication level in a very basic test application with the purpose of just sending and receiving simple messages. In large systems (figure 1) with up to 500 agents the average message sending time is kept under 22 ms if all 500 agents sends 1 message concurrently to a random receiver from a pool of 500 receivers, but it increases dramatically if we increase the number of receivers for the same message. If the 500 agents send the message to 80 receivers the average message time the time is approximately 10 seconds on average for just a single message. The measured time reflect solely the time an agent spends on executing the *send* method.

Behavior based Agent Architecture

We were quiet satisfied with the simple behavior and communication model of JADE, so similar features should be available in a new approach as well. Instead of doing serious hacking in the JADE source, we decided to create a simple agent platform from scratch, which were implemented under two important assumptions, at least for the first version.

- All agents were running in the same execution space with shared memory
- Receivers would not modify received message objects

[1] One of the parameters to *handleSend*. The *send* method of Agent is final, so it cannot be overridden.

With all agents in shared memory on the same machine, remoting could be avoided and message transfers could be handled by moving a simple memory reference. Especially due to the second assumption, because there would be no need to clone message objects if receivers do not modify the content.

The choice of using C# over Java was based both on ease of system integration with the rest of our application setup, but mainly due to the advantageous language features about events and delegations. With only a few modifications the pattern can also be implemented in Java.

The architecture is basically the same, as illustrated in figure 4. Agents are still connected to a container, but at the moment, there is no support for distributed containers. The container manages all the agents and put messages into the message queue of a receiver, and the same message object goes into all receivers.

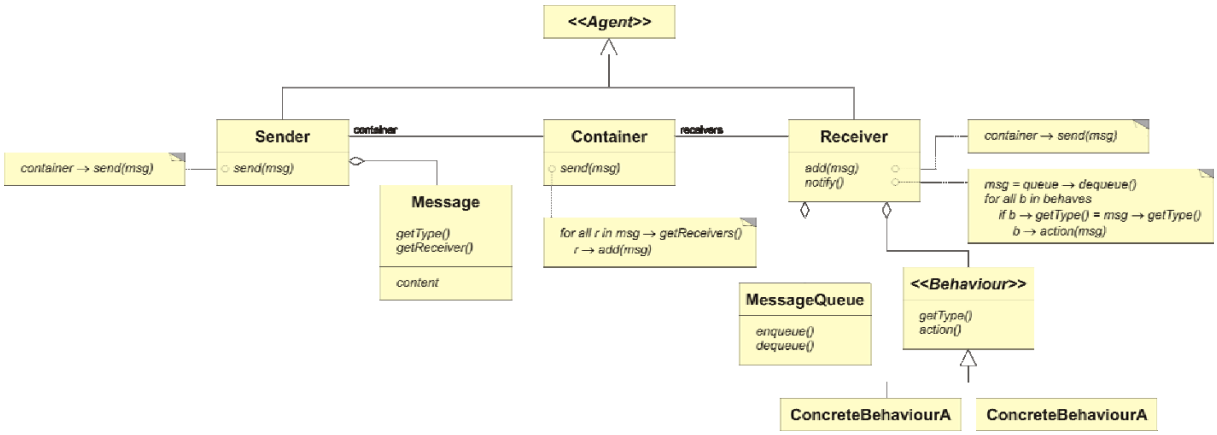


Figure 4: Pattern Structure

Message passing is in focus in figure 4, where both sender and receivers are instances of an agent class. Sender passes a message object to the container, which look up all the receivers and add the message object to their message queue. All receivers are then awaked by a notification and messages from the message queue will be processed, which means that for all behaviors attached to the current receiver the action method of the behavior will be invoked if the message type matches the type of the message.

Thus the communication principle is rather straight forward and is based on a pure asynchronous communication model. The sequence diagram of figure 5 illustrate the message being sent through the container and end up in the message queue of the receiver, which notifies and wakes up the receiver agent. The receiver agent then extracts the message from the queue and activates the appropriate behaviors in its own thread, and the sender agent can continues its own task as soon as it has delivered the message.

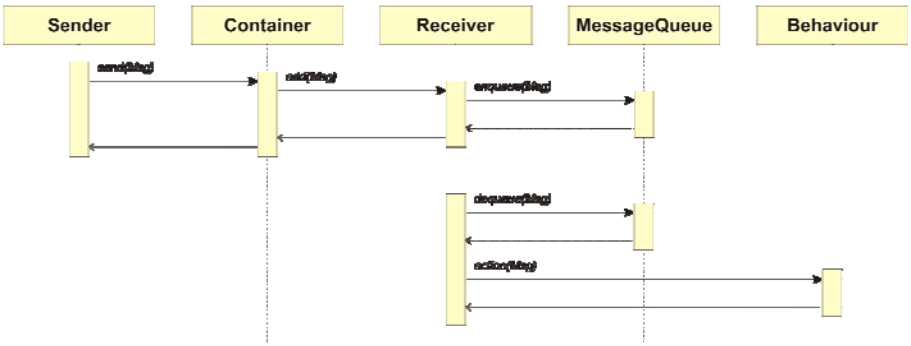


Figure 5: Sequence diagram for the communication model

Implementation

The pattern design presented in figure 4 represents the core architecture of the agent platform, which basically provides a communication platform for the agent-based systems. Therefore we also constrain the discussion of implementation details to this part of the agent platform.

In order to make the platform valuable the communication must be lightweight and the event mechanisms for both notifying receivers and activating behaviors must be very efficient.

Agent body: Sending message from the sender through the container management is rather straight forward. Instead it is not trivial to decide how receivers should activate their behaviors based on incoming messages.

All agents have their own thread and if not active performing some tasks, they will be sleeping, waiting for input, such as message. The main run loop of the thread body of a basic agent is given by the listing below

```
 1:        private void MainLoop()
 2:        {
 3:            m_Alive = true;
 4:            while (m_Alive)
 5:            {
 6:                m_SyncEvent.WaitOne();
 7:                while (m_MessageQueue.Count > 0)
 8:                {
 9:                    Message msg = null;
10:                    lock (m_MessageQueue)
11:                        msg = m_MessageQueue.Dequeue();
12:
13:                    if(m_Behaviours.ContainsKey(msg.GetType()))
14:                    {
15:                        BehaviourActionMsgDelegate behaviours = m_Behaviours[msg.GetType()];
16:                        if (IsBehavioursAsync)
17:                        {
18:                            foreach (BehaviourActionMsgDelegate bh in behaviours.GetInvocationList())
19:                                bh.BeginInvoke(msg, null, null);
20:                        }
21:                        else
22:                            behaviours.Invoke(msg);
23:                    }
24:                }
25:            }
26:        }
```

The activeness of the agent is controlled by a WaitHandle (*m_SyncEvent* in line 6), which can be signaled whenever new messages in the message queue are ready to be processed. When the agent is awakened it will process all messages in the message queue before it falls asleep again.

For each message the agent checks if it has behaviors that match the received message type (line 13). *m_Behavior* is a mapping of *BehaviourActionMsgDelegate* objects, which extends the *MultiCastDelegate* C# type that can chain delegates. So whenever a new behavior is added to an agent, it is just added the particular chain of the message types it matches.

This chain of delegates will be invoked with the current message object either sequentially or by asynchronous calls (line 16-22) based on the *IsBehaviourAsync* property of the Agent that can be modified by the programmer.

Behaviors: The consequence of invoking behaviors, using the simplified principle of delegates as presented above, is that behaviors must have a single invocation point, which can be activated after instantiation. JADE has a similar approach with the *action* method that runs the behavior, so the core task can be isolated from instantiation, recycling, etc. of the behavior object.

Delegates, which are method pointers in C#, are a perfect mechanism to achieve an even more flexible solution, which is not tied to a specific method name. For convenience programmers are still encouraged to implement invocation points as methods with the name *action*. Thus a simple behavior could look like

```
1:        public class ConcreteBehaviour : Behaviour
2:        {
3:            ConcreteBehaviour(...) { ... }  // Constructor
4:
5:            public void Action(Message msg)
6:            {
7:                // TODO : the tasks of the behavior
8:            }
9:        }
```

Remember that behaviors are matched against the message type, so an efficient way for a behavior to match a message is to provide action methods that take subtypes of *Message* as their single argument. This also overcomes another classic problem with JADE of having behaviors that should react on different messages. In JADE the problem can be solved by a more complex message template, or synchronizing behaviors with a shared data store to transfer data between the behaviors, which is rather non-intuitive for beginners.

With this approach the programmer simply add action methods to the behavior object for each of the message types to be matched, as exemplified below

```
1:        public class ConcreteBehaviour : Behaviour
2:        {
3:            ConcreteBehaviour(...) { ... }  // Constructor
4:
5:            public void Action(ConcreteMessageA msg)
6:            {
7:                // TODO : the tasks of the behavior when message type A is received
8:            }
9:
10:           public void Action(ConcreteMessageB msg)
11:           {
12:               // TODO : the tasks of the behavior when message type B is received
13:           }
14:       }
```

The final thing missing is how delegates for the invocation points of the behavior are coupled to the agent. In order to make it as simple as possible for the programmer, an *addBehaviour* method on an *Agent* provides a standard way of adding a behavior to the agent.

```
1:        internal protected void AddBehaviour(Behaviour behave)
2:        {
3:            IEnumerator<Type> enumerator = behave.GetEnumerator();
4:            while (enumerator.MoveNext())
5:            {
6:                if (!m_Behaviours.ContainsKey(enumerator.Current))
7:                    m_Behaviours[enumerator.Current] = behave[enumerator.Current];
8:                else
9:                    m_Behaviours[enumerator.Current] += behave[enumerator.Current];
10:           }
11:           behave.MyAgent = this;
12:       }
```

The behavior super class simply supports iteration by implementing the *IEnumerable*<*Type*> interface, so we can loop through all the message types that a behavior will respond to, and they are added to the mapping of behaviors in the agent (line 7 and 9).

The only part missing to be explained is how the delegates are created and how we can enumerate the invocation points for any sub types of the behavior class. As shown in line 7 and 9 above we use the special *indexer* language construct of C# to get a delegate of a certain type for a behavior. The indexer create the delegate for the right *action* method

```
1:        public BehaviourActionMsgDelegate this[Type msgType]
2:        {
3:            get
4:            {
5:                foreach (MethodInfo mthInfo in this.GetType().GetMethods())
6:                {
7:                    if (mthInfo.Name == ACTION_METHOD_NAME)
8:                        if(mthInfo.GetParameters()[0].ParameterType.Equals(msgType))
9:                            return delegate(Message msg) { mthInfo.Invoke(this, new object[] { msg }); };
10:               }
11:               return null;
12:           }
13:       }
```

We simply use reflection to find the right method that takes the correct message parameter, create a new delegate for this method, and return it. Finally, the enumeration implementation searches through the behavior object using reflection and create a list of supported types, but it skips super types whenever one matching method have been added.

```
1:        public IEnumerator<Type> GetEnumerator()
2:        {
3:            List<Type> tempList = new List<Type>();
4:            foreach (MethodInfo mthInfo in this.GetType().GetMethods())
5:            {
6:                if (mthInfo.Name == ACTION_METHOD_NAME)
7:                {
8:                    // Only add if a sub-type not allready has been added
9:                    Type mthType = mthInfo.GetParameters()[0].ParameterType;
10:                   foreach (Type type in tempList)
11:                       if (type.IsSubclassOf(mthType))
```

```
12:                         goto Found;
13:                     tempList.Add(mthType);
14:                 Found:
15:                     continue;
16:                 }
17:             }
18:             return tempList.GetEnumerator();
19:         }
```

One could claim that reflection and the rather general approach for adding and invoking behaviors presented above are not very efficient, but for the applications in mind the setup of agents and behaviors are done at initialization. Thus there is no performance overhead of reflection and iteration during runtime, where all activation is handled though lookups in mappings (constant time) and efficient delegates.

Results

As mentioned in the introduction our motivation was primarily based on bad experiences of performance when we implemented a baggage handling system using JADE [Hallenborg, 2006]; a large complex real life system with extensive communication for coordinating the activities. The system has now been re-implemented using the presented agent platform, which fully eliminated the performance problems.

Figure 2 and 3 showed the lack of responsiveness of agents in JADE as the communication increase. Corresponding results for our platform is showed in figure 6.

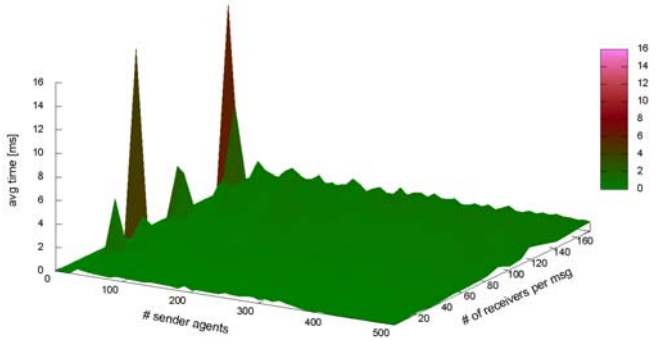


Figure 6: Average message sending time for this agent platform

Apart from a few non-consistent spikes in the case of just one sender agent, the graph clearly illustrates that the average message sending time is constant regardless of the number of sender agents and the number of messages they transmit. The average message time is between 0.5 to 1.0 ms, even for 500 agents each sending 160 messages. A factor 10,000 less than the result in figure 2 for 500 agents sending 80 messages (the highest number possible to generate on the test machine).

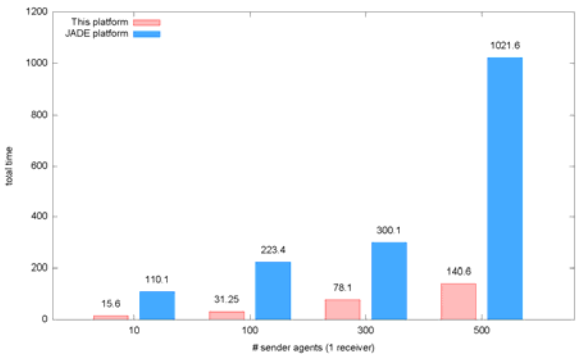


Figure 7: Total time for different number of agents, but only one message sent per agent

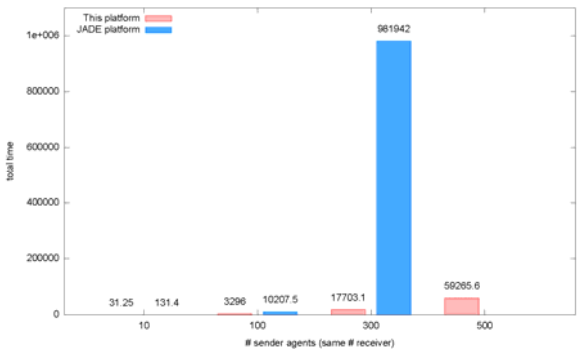


Figure 8: Total for different number of agents, sending the same number of messages per agent

Truthfully, the average message time and responsiveness is just one side of the story. The total computation time of the system is another important factor. Figure 7 and 8 illustrate how our agent platform outperforms JADE.

The results are very clear. Even with only one message sent per agent, where message duplication not should be a problem in JADE, our platform is still a factor 4-7 times faster than JADE. Keep in mind that these implementations are the simplest possible to test the communication model, no advanced encoding and decoding of message content are included, which would slow the JADE solution ever further. In figure 8, where the communication increases dramatically our platform could send and receive the 90,000 message within 18 seconds, but it took more than 16 minutes in JADE. The test machine could not complete the 500 times 500 example in JADE.

Conclusion

We have presented a pattern structure to implement the core communication model of an agent platform. Implementations details that take advantages of special language constructs in C# to efficiently activate and invoke the behaviors of an agent are outlined. The implemented agent platform is compared to agent implementations in JADE, and the presented platform clearly outperforms JADE in all situations, and especially when the communication increase. So for the applications domains in mind (large complex manufacturing and logistics) we have created a very efficient agent platform.

Bibliography

[Weiss, 1999] G.Weiss. Multiagent Systems - A modern approach to distributed artificial intelligence. MIT Press, 1999.

[Jennings, 2000] N.Jennings and M.Wooldridge. Agent-Oriented Software Engineering. In: Handbook of Agent Technology. Ed. J.Bradshaw. AAAI/MIT Press, 2000.

[Searle, 1969] J.R.Searle. Speech Acts. Cambridge University Press, 1969.

[FIPA, 2002] Foundation for Intelligent Physical Agents (FIPA), FIPA Communicative Act Library Specification, 2002.

[Cheong, 2006] C.Cheong and M.Winikoff. Hermes: Designing goal-oriented agent interactions. In: Agent-Oriented Software Engineering VI: 6th Int. Workshop. Ed. J.P.Müller and F.Zambonelli. Lecture Notes in Computer Science, Vol. 3950 Springer Verlag, page 16-27, 2006.

[Braubach, 2007] L.Braubach and A.Pokahr. Goal-Oriented Interaction Protocols. In: Fifth German conference on Multi-Agent System TEchnologieS (MATES-2007), 2007.

[Cabri, 2006] G.Cabri, L.Ferrari, and L.Leonardi. Supporting the Development of Multi-Agent Interactions via Roles. In: Agent-Oriented Software Engineering VI: 6th Int. Workshop. Ed. J.P.Müller and F.Zambonelli. Lecture Notes in Computer Science, Vol. 3950 Springer Verlag, page 154-166, 2006.

[Bellifemine, 2005] F.Bellifemine, F.Bergenti, G.Caire and A.Poggi. JADE - a java agent development framework. Multi-Agent Programming: Languages, Platforms and Applications. Ed. R.Bordini, M.Dastani, J.Dix and A.Seghrouchni. Number 15 in Multiagent Systems, Artificial Societies, and Simulated Organizations. Springer, page 125–148, 2005

[Hallenborg, 2006] K.Hallenborg and Y.Demazeau. Dynamical Control in Large-scale Material Handling Systems through Agent Technology. The 2006 IEEE/WIC/ACM International Conference on Intelligent Agent Technology (IAT-06), HongKong, China, December 18-22, 2006.

Authors' Information

Kasper Hallenborg *– Assistant Professor; Maersk McKinney Moller Institute, University of Southern Denmark, Campusvej 55, DK-5230 Odense M, Denmark; e-mail: hallenborg@mmmi.sdu.dk*

CASE-BASED REASONING TOOLS FROM SHELLS TO OBJECT-ORIENTED FRAMEWORKS

Essam Abdrabou, AbdEl-Badeeh Salem

***Abstract**: A Case-Based Reasoning (CBR) tool is software that can be used to develop several applications that require cased-based reasoning methodology. CBR shells are kind of application generators with graphical user interface. They can be used by non-programmer users but the extension or integration of new components in these tools is not possible. In this paper we analyzed three CBR object-oriented framework development environments CBR*Tools, CAT-CBR, and JColibri. These frameworks work as open software development environment and facilitate the reuse of their design as well as implementations.*

***Keywords**: Case-Based Reasoning, Case-Based Reasoning Shells, CBR, CBR Shells*

***ACM Classification Keywords**: I.2.5 Expert system tools and techniques - Conference proceedings*

***Conference**: The paper is selected from XIVth International Conference "Knowledge-Dialogue-Solution" KDS 2008, Varna, Bulgaria, June-July 2008*

Introduction

CBR is based on psychological theories of human cognition [Watson, 1997]. It rests on the intuition that human expertise does not depend on rules or other formalized structures but on experiences.

Since late 70s many CBR applications have been developed by research institutes or industrial companies in order for solving specific domain problems. In addition, there are several tools or shells have been built to facilitate the building of a CBR application by non-programmer users. Most of these tools aim to provide Application Programming Interfaces (APIs) which provide a set of functions that deal with CBR algorithms and methodologies. They intended to help programmers to embed these APIs in their application development. In order to access more complex problems the research goes to provide an open development environment that lead users to more uniform tool at the level of design [Jaczynski & Trousse, 1998].

The concept of object-oriented frameworks has been introduced in the late 80's and has been defined as "*a set of classes that embodies an abstract design for solutions to a family of related problems, and supports reuses at a larger granularity than classes* [Johnson & Foote, 1988]".

Frameworks allow the reuse of both code and design for a class of problems, giving the ability to non-expert to write complex applications efficiently. A framework can be considered as a semi-complete application than can be specialized to produce custom applications [Bello-Tomas et al. 2004]. A framework can be applied in a wide range of domain, and can be enhanced by the adding of new components.

This paper gives a survey on some of CBR shells, shows the need for the development of CBR tools based on open framework environment and discusses three object-oriented based CBR frameworks CBR*Tools, CAT-CBR, and JColibri. The paper shows the importance for developers of CBR applications to move from shells to object-oriented frameworks that facilitate the reuse of their design as well as the code to implement the intended CBR application. The paper is divided into five sections. Section one is this introduction. Section 2 gives a theoretical background on case-based reasoning. Section 3 discusses some of the CBR shells and developing environments. Section 4 discusses the importance of moving to object-oriented frameworks and studies the three proposed frameworks. Finally, section 5 concludes for the work.

Theoretical Background

In case-based reasoning (CBR) systems expertise is embodied in a library of past cases, rather than being encoded in classical rules. Each case typically contains a description of the problem, plus a solution and/or the outcome. The knowledge and reasoning process used by an expert to solve the problem is not recorded, but is implicit in the solution. To solve a current problem: the problem is matched against the cases in the case base, and similar cases are retrieved. The retrieved cases are used to suggest a solution that is reused and tested for success. If necessary, the solution is then revised. Finally the current problem and the final solution are retained as part of a new case.

CBR is a five-step problem solving process. Figure 1 shows the CBR cycle.

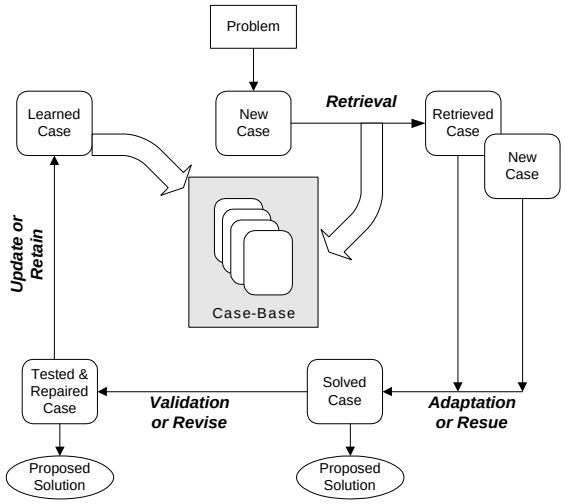


Figure 1: The CBR Cycle

Representation: Given a new situation, generate appropriate semantic indices that will allow its classification and categorization. This usually implies a standard indexing vocabulary that the CBR system uses to store historical information and problems. The vocabulary must be rich enough to be expressive, but limited enough to allow efficient recall [Kolodner, 1993].

Retrieval: Given a new, indexed problem, retrieve the best past cases from memory. This requires answering three questions: What constitute an appropriate case? What are the criteria of closeness or similarity between cases? How should cases be indexed? Indexing a case is essential in establishing similarity, because the indices help define the important elements of a problem, those that should be considered when studying the problem. Thus, part of the index must be a description of the problem that the case solved, at some level of abstraction. Part of the case is also the knowledge gained from solving the problem represented by the case [Kolodner, 1993].

Retrieving a case starts with a (possibly partial) problem description and ends when best matching cases found. The subtasks involve identifying a set of relevant problem descriptors, matching the case and returning a set of sufficiently similar cases; and selecting the best case from the set of cases returned.

Adaptation: Modify the old solutions to confirm to the new situation, resulting in a proposed solution. With the exception of trivial situations, the solution recalled will not immediately apply to the new problem, usually because the old and the new problem are slightly different [Kolodner, 1993].

Reusing the retrieved case solution in the context of the new case focuses on: identifying the differences between the retrieved and the current case; and identifying the part of a retrieved case that can be transferred to the new case. Generally the solution of the retrieved case is transferred to the new case directly as its solution case.

Validation: Determine whether the proposed solution is successful. Checking a solution can take many forms, depending on the domain. Whatever the means, after the system checks a solution, it must evaluate the results of

this check. If the solution is acceptable, based on domain criteria, the CBR system is done with reasoning. Otherwise, the case must be modified again, and this time the modifications will be guided by the results of the solution's evaluation [Kolodner, 1993]. Revising the case solution generated by the reuse process is necessary when the solution proves incorrect. This provides an opportunity to learn from failure.

Update: If the solution fails, explain the failure and learn it, to avoid repeating it. If the solution succeeds and warrants retention, incorporate it into the case memory as a successful solution and stop. The CBR system must decide if a successful new solution is sufficiently different from already-known solutions to warrant storage. If it does warrant storage, the system must decide how the new case will be indexed, on which level of abstraction it will be saved, and where it will be put inside the case-base organization [Kolodner, 1993].

Case-Based Reasoning Shells and Development Environments

CBR shells are kind of application generators with graphical user interface. They can be used by non-programmer users but the extension or integration of new components in these tools is not possible. There is a clear difference between a CBR application and a CBR shell. A CBR application is a direct implementation of CBR methodology to a specific domain problem in order to solve this problem. On the other hand, a CBR shell is an application that enables developers to develop a domain specific CBR application.

In the late 1980s, the U.S. DARPA program funded a series of workshops on CBR and the development of a CBR tools [DARPA, 1991]. This tool became Cognitive System's ReMind and marked the transition of CBR from purely academic research in cognitive science and artificial intelligence into the commercial area.

Many CBR shells have been developed to make the theory practically feasible [Watson, 1997]. Table 1 shows a summary of the key features of the major CBR shells. In addition to the previous tools there are three major CBR development environments CASPIAN, CASUEL, and CBR-Works.

Table 1: A Summary of the Major CBR Sheels [Watson, 1997]

Product	Platform	Representation	Retrieval	Interface
ART Enterprise	PC, Workstation	flat attribute:value pairs supporting a full range of variable types	Nearest-neighbor	Fully featured GUI builder
CaseAdvisor	PC Windows	Flat records supporting text and weighted questions	Nearest-neighbor and knowledge-guided	Use Netscape
CBR3	PC Windows	Flat records supporting text and weighted questions	Nearest-neighbor and knowledge-guided	CasePoint available as a DLL or API and CGI scripts
Eclipse	Any ANSI C environment	Flat attribute	Nearest-neighbor	No interface, only supply as a C library
ESTEEM	PC Windows	Case can be nested	Nearest-neighbor with inductive weight generation	Simple form-based GUI builder
KATE	PC Windows and UNIX	Hierachical cases	Nearest-neighbor and induction	ToolBook interface can be customized

CASPIAN [Pegler & Price, 1996] is CBR tool in the public domain developed at the University of Aberstwyth in Wales. It was used as the CBR component of the Wayland system. It has a simple command line interface, but can be integrated with a GUI front end if required. CASPIAN is written in C and can run on MS-DOS, MAC or UNIX but without the GUI. CASPICAN performs nearest-neighbor matching and used rules for case adaptation. It stores a case-base, including adaptation rules, in ASCII file. An individual case comprises a series of attributes and a solution. CASPIAN has an internal engine sophisticated enough to allow its use in industrial applications.

CASUEL [Manago et al., 1994], the Common Case Representation language developed by the European INRECA project (Integrated Reasoning from Cases), is the interface language between the INRECA component systems. It is also intended to serve as the interface language between the INRECA integrated system and the external world, and as a standard for exchanging information between classification and diagnostic systems that

use cases. CASUEL is a flexible, object-oriented, frame-like language for storing and exchanging descriptive models and case libraries as ASCII files. It is designed to model naturally the complexities of real cases. CASUEL represents domain objects in a class hierarchy using inheritance, slots being used to describe the objects, with typing constraints on slot values, as well as different kinds of relationships between objects.

CASUEL also supports rule formalism for exchanging case completion rules and case adaptation rules, as well as a mechanism for defining similarity measures. CASUEL is more concise than flat feature values vectors for representation of objects with a large number of potentially relevant attributes of different types, only a few of which are applicable to any given case. Its use reduces the number of information-gain calculations needed for induction systems or similarity computations required for case-based reasoning.

CASUEL does not require applications to use all of them. CASUEL is a keyword-driven language that allows different system components to ignore irrelevant definitions. CASUEL is also open in the sense that new features can be defined, if necessary for a particular kind of application of component [Watson, 1997].

CBR-Works can be seen as a CBR-Shell providing all necessary tools to model, maintain, and consult a case base [Schulz, 1999]. CBR-Works comes from the German company TECINNO, running on MS Windows, Mac, OS/2, and various UNIX platforms. Written in SMALLTALK, it supports an object-oriented model and flexible retrieval methods. It also supports the definition of concept and type hierarchies to help define similarity of symbolic concepts. CBR-Works includes an attribute editor, a rule editor, similarity criteria editor, distributed processing support and is easily integrated to existent applications. CBR-Works can import case-bases from Microsoft Excel and in the CASUEL case format.

Table 2 shows a summary of the three discussed CBR development environments.

Table 2: A Summary of the Major CBR Development Environments

Product	Platform	Representation	Retrieval	Interface
CASPIAN	DOS, MAC, or UNIX	Attribute-Value for feature representation	Nearest-neighbor	Can be integrated with a GUI
CASUEL	Portable	Frame-like language for storing and exchanging descriptive models as ASCII files	Not Applicable	Not Applicable
CBR-Works	MS Windows, MAC, or UNIX	Flat records supporting text and weighted questions	Nearest-neighbor with support of feature weights	Fully featured GUI

Case-Based Reasoning Object-Oriented Frameworks

Most of the CBR tools presented in scientific papers aim to provide Application Programming Interfaces (APIs) which provide a set of functions that deal with CBR algorithms and methodologies. They intended to help programmers to embed these APIs in their application development [Jaczynski & Trousse, 1998]. Usually these APIs can be extended by the programmer to modify the provided algorithms. However, none of these tools are designed to provide an open development environment that lead users to more uniform tool at the level of design [Jaczynski & Trousse, 1998].

The concept of object-oriented frameworks has been introduced in the late 80's and has been defined as "*a set of classes that embodies an abstract design for solutions to a family of related problems, and supports reuses at a larger granularity than classes*" [Johnson & Foote, 1988].

The goal of a framework is to capture a set of concepts related to a domain and the way they interact. In addition a framework is in control of a part of the program activity and calls specific application code by dynamic method binding. A framework can be viewed as an incomplete application where the user only has to specify some classes to build the complete application [Jaczynski & Trousse, 1998].

Frameworks allow the reuse of both code and design for a class of problems, giving the ability to non-expert to write complex applications quickly. Frameworks also allow the development of prototypes which could be extended further on by specialization or composition. A framework once understood, it can be applied in a wide range of domain, and can be enhanced by the adding of new components [Jaczynski & Trousse, 1998].

Before exploring the CBR frameworks, there are some points inside the framework that need to be addressed [Jimenez-Diaz & Gomez-Albarran, 2004]:

- Users must know the type of application which the framework can be used. Users should understand whether or not the application could be developed based on their choice of the framework.
- The mapping between application domain concepts and framework classes should be well studied to avoid the normal indirect mapping between domain entities and framework class.
- The framework users need to know behavior of elements within the framework in order to identify the hierarchy of classes that will be involved in the design of the application.
- Users need to study carefully the communication between the classes of the framework in order to avoid the integrity problem of the framework.
- Some problems like the duplication of functionality and extension of some parts of the framework can be avoided by the knowledge of framework architecture.

The following is a discussion of three object-oriented CBR frameworks CBR*Tools, CAT-CBR, and JColibri. We discuss their architecture and how CBR methodologies are applied in them.

CBR*Tools [Jaczynski, 1998] is an object-oriented framework for CBR which is specified with the Unified Modeling Language (UML) notation [Booch, 1994] and written in Java. It offers a set of abstract classes to model the main concepts necessary to develop applications integrating case-based reasoning techniques: case, case base, index, measurements of similarity, reasoning control. It also offers a set of concrete classes which implements many traditional methods (closest neighbors indexing, Kd-tree indexing, neuronal approach based indexing, standards similarities measurements). CBR*Tools contains more than 220 classes divided in two main categories: the core package for basic functionality and the time package for the specific management of the behavioral situations. The programming of a new application is done by specialization of existing classes, objects aggregation or by using the parameters of the existing classes.

CBR*Tools delegates each CBR step retrieve, reuse, revise or retain to a different object. Each class defines an abstract interface to a step of the reasoning while the Reasoner class defines how to control the reasoning. The step classes must be specialized to implement a specific reasoning. The Reasoner class allows the implementation of different reasoning control methods. In order to ensure that the reasoning step implementations and the reasoning object are consistent, the ReasonerFactory class is provided. Figure 2 shows the class diagram of CBR*Tools object model.

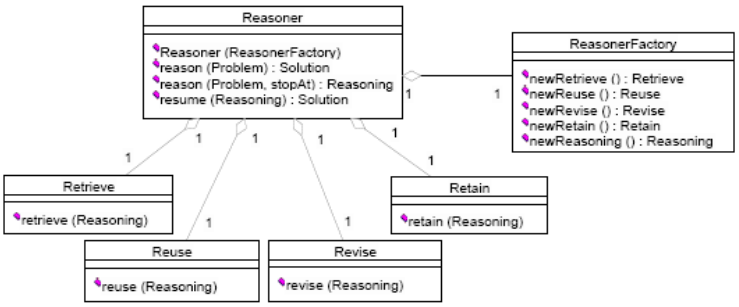


Figure 2: CBR*Tools Object Model

CAT-CBR platform uses a library of CBR components to guide the user in the development of a CBR application [Abasolo et al., 2002]. These components describe the different tasks that can appear in a CBR system and also the problem solving methods that can be applied to these tasks. The CAT-CBR platform has been developed on Noos platform [Arcos, 1997]. Noos uses feature terms as representation language.

Universal Problem-solving Methods Language (UPML) has been used to describe the CBR components used inside the framework [Abasolo et al., 2002]. Two levels can be differentiated in a component description: a specification level in which UPML is used and an operational level in which the Noos is used.

CAT-CBR uses two processes to enable users to develop a CBR application the configuration process and the operationalization process. The configuration process focuses on selecting different components and connecting them in order to specify an application. CAT-CBR has an interactive tool where users choose the components that need to be included in an application. This tool is built over a CBR system that guides and gives support to users during the configuration process. The operationalization process takes an application specification and generates an executable application. The platform generates a file that links with Noos methods following the structure of the configuration of components. Figure 3 shows the process of developing a CBR system. It is done in three steps: Configure, Enable, and Enact.

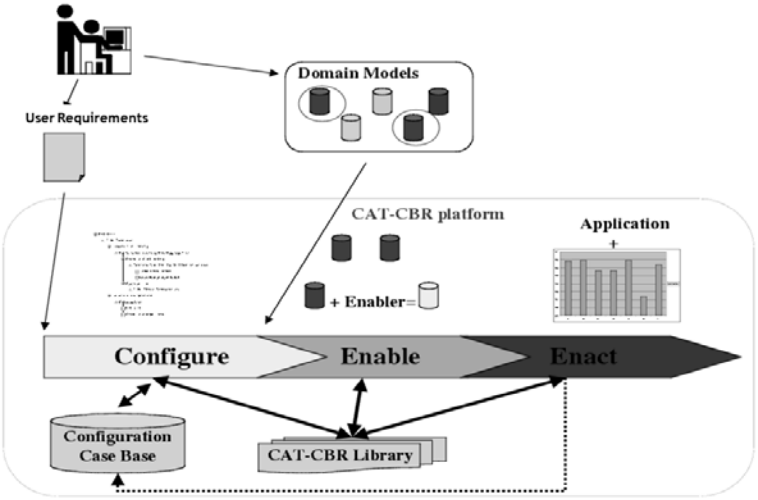


Figure 3: CAT-CBR Process of Developing a CBR System [Abasolo et al., 2002]

The goal of the Configure step is to decide which technique will be used in the CBR system. Only general information about the desired CBR system is required; this information is about general objectives (i.e. classify), or performance characteristics (i.e. noise tolerance). As result of the configure step, users get a configured CBR system; this configured CBR system is a task-method decomposition of components from the CAT-CBR library. This configured system specifies also which models will be used by each method.

The goal of the Enable step is to link the configured system with the concrete domain. In this step user have two options, first, they can assign the concrete models that the configuration needs to be carried out; second, they can use methods to acquire these models that the configuration needs and they are not currently available.

Enact: Finally in the Enacting step, the configuration and models will be translated into an executable code. As the platform is developed over Noos framework the resultant code will be Lisp functions. Once the configuration is operationalize, the application can run to solve new problems.

JColibri: The application framework of JColibri [Bello-Tomas et al., 2004] comprises a hierarchy of JAVA classes plus a number of XML files. The framework is organized around four main elements: tasks and methods, case-base, cases, and problem solving methods.

Tasks and Methods: XML files explain tasks supported by the framework and the methods to solve these tasks. Tasks are the key elements that represent the method goal and can identify it by name and description in an XML file. Users can add task to the framework at anytime.

Case Base: jColibri has a memory organization interface that assumes that whole case-base can be read into memory for the CBR to work with it. It is not feasible for big size. JColibri implemented a new interface who allows retrieving cases enough to satisfy a SQL query. A second layer of case base is a data structure which will organize cases after they loads into memory. The two layer approach is efficient enough to allow different strategies for retrieving cases.

Cases: jColibri represent cases in a very simple way. A case is individual which has number of relationships with other individuals. Framework is supported by different data types which define any simple case.

Problem Solving Methods: JColibri deals with the CBR methodology as follows:

- Retrieval: Main focus of methods in this category is to find similarity between cases. Similarity function can be parameterized through system configuration.
- Reuse: a complete design where case-based and slot-based adaptation can be hooked is provided.
- Revise: It is not supported by JColibri framework.
- Retain: Process of updating the case base is totally based on implementation of the case-base.

Conclusion

In this paper three object-oriented CBR frameworks have been studied CBR*Tools, CAT-CBR and JColibri. CBR*Tools is an object oriented framework implemented in JAVA. The framework identifies the delegation of reasoning steps, the separation of case storage and case indexing, the design of indexes as reusable components, and the design of adaptation patterns. CAT-CBR uses UPML for specifying CBR components and provides: a library of CBR components, a language and a graphical editor to specify new components and syntax to implement their operational code, and a broker service that allows specifying the requirements of a target CBR application and the available models in a domain. JColibri is an object-oriented framework implemented in Java. It uses XML to configure its data which make it interchangeable between computers. The development framework of JColibri supports the simple to most complex knowledge support.

The move to object-oriented CBR frameworks has many advantages including modularity which helps improve software quality, reusability that leverages the domain knowledge and prior effort of experienced developers in order to avoid recreating and revalidating common solutions, and extensibility which enhanced by providing explicit methods that allow applications to extend its stable interfaces. So, it is recommended for researchers or for industrial applications that need to solve their domain problems to use the available CBR frameworks.

Bibliography

[Aamodt & Plaza, 1994] A.Aamodt, and E.Plaza. Case-Based Reasoning: Foundational Issues, Methodological Variation and System Approaches. AICOM, Vol.7, No.1, pp. 39-58.

[Abasolo et al., 2002] C.Abasolo, E.Plaza, J.L.Arcos. Components for Case-based Reasoning Systems, In Topics in Artificial Intelligence, Lecture Notes in Artificial Intelligence, Vol. 2504, p. 1-12.

[Althoff et al., 1995] K-D.Althoff, E.Auriol, R.Barletta, M.Manago. A Review of Industrial Case-Based Reasoning Tools. In Goodall, A. (Ed.), An AI Perspectives Report. Oxford: AI Intelligence.

[Arcos, 1997] J.L.Arcos. The Noos representation language. PhD thesis, Universitat Politecnica de Catalunya.

[Bello-Tomás et al., 2004] J.J.Bello-Tomás, P.A.González-Calero, B.Díaz-Agudo. JColibri: An Object-Oriented Framework for Building CBR Systems. In Advances in Case-Based Reasoning, Lecture Notes in Computer Science. Springer Berlin/ Heidelberg, Vol. 3155/2004, p. 32-46.

[Booch, 1994] G.Booch. Object-oriented analysis and design. Redwood City, CA: The Benjamin/ Cummings Publishing Company, Inc.

[DARPA, 1991] Proceedings from the Case-Based Reasoning Workshop, Washington D.C., May 8-10, 1991. Sponsored by DARPA. Morgan Kaufmann, 1989.

[Jaczynski & Trousse, 1998] M.Jaczynski, B.Trousse. An Object-Oriented Framework for the Design and the Implementation of Case-Based Reasoners. In Proceedings of the 6th German Workshop on Case-Based Reasoning, Berlin.

[Jimenez-Diaz & Gomez-Albarran, 2004] G.Jimenez-Diaz, M.Gomez-Albarran. A Case-Based Approach for Teaching Frameworks: Universidad Computense de Madrid, Juan del Rosal 8. 28040 Madrid Spain.

[Johnson & Foote, 1988] R.E.Johnson, B.Foote. Designing Reusable Classes. Journal of Object-Oriented Programming, 1(2), 22–35.

[Kolodner, 1993] J.L.Kolodner. Case-Based Reasoning, California: Morgan Kaufmann Publishers.

[Manago et al., 1994] M.Manago, R.Bergmann, N.Conruyt, R.Traph ner, J.Pasley, J.Le Renard, F.Maurer, S.Wes, K.D.Althoff, S.Dumont. CASUEL: a common case representation language. ESPRIT project 6322,Task 1.1, Deliverable D1.

[Pegler & Price, 1996] I.Pegler, C.J.Price. CASPIAN: A Freeware Case-Based Reasoning Shell. In proceedings of the Second U.K. Workshop on Case-Based Reasoning, edited by I. Watson. Salford, UK: Salford niversity.

[Plaza & Arcos, 2000] E.Plaza, J.L.Arcos. "Towards a software architecture for case-based reasoning systems", Foundations of Intelligent Systems, 12th International Symposium, ISMIS 2000. Ras, Z. W. and Ohsuga, S., (Eds.), Lecture Notes in Computer Science 1932.

[Schulz, 1999] S.Schulz. CBR-Works- A State-of-the-Art Shell for Case-Based Application Building. Proceedings of the CWCBR-1999, Wurzburg.

[Watson, 1994] I.Watson. The Case for Case-Based Reasoning, Proceedings of EPSRC/DRAL, November 1994, pp.55-64.

[Watson, 1997] I.Watson. Applying Case-Based Reasoning: Techniques for Enterprise Systems. California: Morgan Kaufmann Publishers.

Authors' Information

Essam Abdrabou *– General-manager, Cairo Engineering Support Labs CESLabs, 36 Al-Imam Ali St., Heliopolis-11351, Cairo, Egypt; e-mail: gm@ceslabs.com*

AbdEl-Badeeh Salem *– Professor; Faculty of Computer and Information Sciences, Ain-Shams University, Abbassia-11566, Cairo, Egypt; e-mail: absalem@asunet.shams.edu.eg*

KNOWLEDGE CONSTRUCTION TECHNOLOGY THROUGH HYPERMEDIA-BASED INTELLIGENT CONVERSATIONAL CHANNEL

S.M.F.D Syed Mustapha

Abstract: *There have been multifarious approaches in building expert knowledge in medical or engineering field through expert system, case-based reasoning, model-based reasoning and also a large-scale knowledge-based system. The intriguing factors with these approaches are mainly the choices of reasoning mechanism, ontology, knowledge representation, elicitation and modeling. In our study, we argue that the knowledge construction through hypermedia-based community channel is an effective approach in constructing expert's knowledge. We define that the knowledge can be represented as in the simplest form such as stories to the most complex ones such as on-the-job type of experiences. The current approaches of encoding experiences require expert's knowledge to be acquired and represented in rules, cases or causal model. We differentiate the two types of knowledge which are the content knowledge and socially-derivable knowledge. The latter is described as knowledge that is earned through social interaction. Intelligent Conversational Channel is the system that supports the building and sharing on this type of knowledge.*

Keywords: *Knowledge Building, Community Channel, Community Learning, Virtual Agents*

ACM Classification Keywords: *K.3.1 Computer Uses in Education (Collaborative Learning)*

Conference: *The paper is selected from Sixth International Conference on Information Research and Applications – i.Tech 2008, Varna, Bulgaria, June-July 2008*

Introduction

There have been multifarious approaches in building expert knowledge in medical or engineering field through expert system, case-based reasoning, model-based reasoning and also a large-scale knowledge-based system. The intriguing factors with these approaches are mainly the choices of reasoning mechanism, ontology, knowledge representation, knowledge elicitation and knowledge modeling. In our study, we argue that the knowledge construction through hypermedia-based community channel is an effective approach in constructing expert's knowledge. We define that the knowledge can be represented as in the simplest form such as stories to the most complex ones such as on-the-job types of experiences. The current approaches of encoding experiences require expert's knowledge to be acquired and represented in rules, cases or causal model. The development time and cost upsurge with the amount of knowledge encoded in the system. Our approach emphasizes on collaborative knowledge construction where experts such as engineers can post their basic knowledge such as simple facts or a more complicated one in the form of stories in the interactive conversational channel. There are experiences that cannot be modeled, as they need to be visually observed in order to grasp the understanding. Our Intelligent Conversational Channel supports other hypermedia such as video, graphics, audio, video, animation, or images as part of knowledge entity in order to handle these types of knowledge. Each member of the community can build his/her own personal knowledge unit or ontology that will then be integrated as a larger form of expert knowledge, so-called expert community knowledge. We introduce an intelligent technique for knowledge integration such that this knowledge can be organized and retrieved by other community members for sharing purpose. We also introduce the knowledge sharing protocol that imposes certain restrictions for security and to prevent repudiation, spoofing of the knowledge being posted. Members are registered for identification but their personal information is not reflected to public as part of the knowledge content. This suppresses the chances of using personal status as a way to force one's understanding to the others. The personal knowledge unit can be protected by the owner at the same time it allows other members to patch their

knowledge to the existing knowledge unit. We believe our approach offers a low construction cost with minimal development time. The knowledge is more naturally encoded and represented and it is continuously maintained by the community for validity and update.

The paper is organized by firstly differentiating theoretically between the content knowledge and socially-derivable knowledge. This is essential as the system we developed; so-called Intelligent Conversational Channel (thereafter, ICC) emphasizes learning on socially-derivable knowledge. Secondly, we describe the process of knowledge construction technology through the Intelligent Conversational Channel. Thirdly, a demonstration of simulated scenario of ICC usage is given for illustration purpose. The paper concludes the discussion in the final section with the future work.

What is content knowledge and socially- derivable knowledge

If one asked a group of people to draw three symbols with different shapes such as rectangle, circle and triangle, the results for each of the participant will differ by the interpretation of the size, position on the paper of each symbol, relative position of each symbol to each other and also other additional imaginative features such as color or dimensions. In different example, a group of people witnessing a car accident is questioned by a police officer. Each witness contributes few pieces of the jigsaw puzzle that eventually shape up an overall understanding of the scene. In the above examples, there are two types of knowledge, which are the content knowledge and socially-derivable knowledge. Content knowledge is the fundamental theory about the knowledge that every learner has to know. Socially-derivable knowledge is about learning the knowledge produced by other learners in terms of its interpretation, analysis, ways of applying the concepts etc. In the former example, the three basic symbols (triangle, circle and triangle) are the content knowledge, which becomes the basic theory for everyone. The varieties of the construction and perhaps the explanation given by the individual who draws it are the socially-derivable knowledge. In the perspective of science and engineering, the formulas, physical laws, rules are examples of content knowledge, which everyone agrees and globally accepted. The approach and methods of applying the content knowledge differs among engineers especially how they interpret the problem, choose suitable techniques and analyze the solution.

Knowledge-shaping

In this paper knowledge-shaping is referred to how knowledge is formed using certain approach. There are many approaches to knowledge-shaping. Among them are knowledge acquisition, building expert system, building qualitative model and constructing case-based system. Knowledge acquisition tool is designed to acquire knowledge from a single expert. In some applications, multiple experts are used in encoding the knowledge into the knowledge base. However, the knowledge framework in the knowledge acquisition tool is rather rigid and designed by taking the perspective of few experts. An expert system is known to be a knowledge acquired from a single human expert. It contains the experience in the form of rules or frames of a single expert. Reasoner system, which is built on the qualitative model, describes the content knowledge that is modeled using qualitative terms and relationships. The qualitative model represents a single model for reasoning on multiple aspects of a problem. Case-based system contains a library of past case collections by an expert that is stored in the system and retrieved by the best-match method. We argue that these knowledge-shaping approaches are best applied in developing content knowledge. Expert system such as MYCIN or case-based learning tool can be used to train professional doctor or engineer into a more specialized skill based on the experience of an expert built on the system. However, the learner does not have the opportunity to learn skills from other experts, which are not encoded in the system he/she is using. Similarly, system that performs reasoning and explanation that is built on qualitative model, is mould to cater for a specific type of learning. The learning method, which is adopted from the current pedagogy theory, is tuned into the system. We claim that these technologies offer know-how knowledge in performing a specific task while in the real world, the process of acquiring knowledge and managing problems are rather ill-structured and difficult to be modeled.

The characteristics of socially-derivable knowledge towards knowledge-shaping are spelled out as follows:

- Multiplicity in learning objects – knowledge in the real world is delivered or obtained in different forms. The objects, which are used as part of the learning whether directly or indirectly is called learning, object as described by Community of Practice [Wenger, 1997]. Radio, television or LCD screen used for advertising are examples of broadcasting system that contribute to one's knowledge. Newspaper, magazines, leaflets or brochures are pieces of information, which transform into one's knowledge when he/she reads them. Other forms of learning objects are the working colleagues, animated or unanimated artifacts such as the copier machine, pets at home, video movies and neighbors whom one socialize with. In this respect, the expert knowledge does not come from a single source as well as the multiplicity in methodology for delivering the knowledge. Expert's talk in the open seminars or television is examples of learning objects.
- Open-world assumptions – assumption is needed when one designs a system to be used as problem-solver. The assumptions are perspective that draws the boundary of the intended world in order for the system to work successfully within the specified limit. In modeling the content-knowledge, close-world assumption is always used. Unlike the content knowledge, socially-derivable knowledge does not specify the assumption as the knowledge is not modeled but shared in its original form. The knowledge contains the description about the real world problems and solution rather than the hypothesized.
- Rapid knowledge-building – content knowledge requires a system builder to analyze and study, to model the solution, to build the system and test its performance. These processes are rather time-consuming and costly. On the other hand, the socially-derivable knowledge is built by the community in a progress manner and can be learned immediately without the need of highly mechanistic and sophisticated process. Knowledge is presented in a human-readable format rather than machine-readable format.

Unorganized, ubiquitous but retrievable – content knowledge built in an expert system is meant to be organized and frequently validated by the truth maintenance technology. The purpose is to avoid conflict of facts and retain consistencies in delivering solution. The retrieval of the solution depends on the reasoning technique employed in the system. Socially-derivable knowledge is rather unstructured and ubiquitous. The knowledge allows conflict solutions to a single problem as it can be treated as having choices of different perspectives. Learners are not confined to solution of a single expert in this case as knowledge is contributed by several experts or non-experts who is involved in the knowledge construction process. The socially-derivable knowledge is retrieved through social interactions and dialogues with the communities.

Knowledge construction technology through ICC

There are many related literature concerning learning through social interaction [Wenger, 1997;Thomas, 2004,Thomas, et al, 2001]. Knowledge building developers have realized the inclusion of community in the process of knowledge building is an essence to the social knowledge building [Stahl, 2000].

In this section we describe the three components of ICC that supports knowledge construction process. This is followed by the three fundamental knowledge construction process which are the knowledge-building process, the knowledge-sharing process and knowledge conversational process.

Components of ICC

We describe the community as the group of people participating in the knowledge-building process. In the actual application, the community could be a group of experts, engineers, scientists or layman such as managers or consumers. The three major components of ICC are the hypermedia-learning space, the discourse communicator, and the discourse analyzer in which all these are connected to the community channel as shown In Figure 1.

The community channel consists of several channels of knowledge units. Each channel is set up for a different set of community, which discusses separate issues. Each channel has a series of knowledge unit as shown in Figure 2.

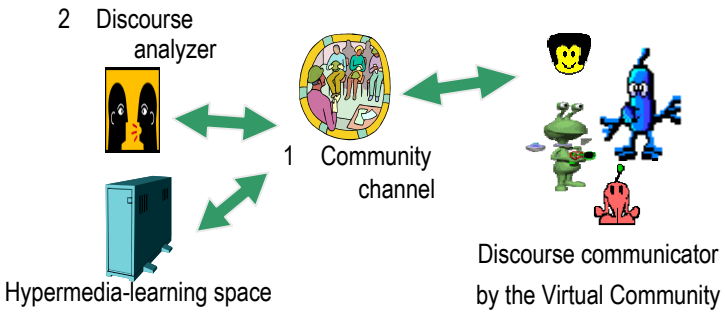


Figure 1- Components of Interactive Conversational Channel

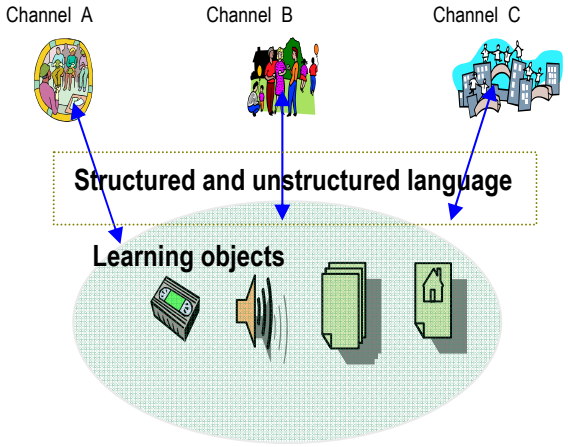


Figure 2- Community Channel

The three channels A, B and C are exclusive as they may be used by different organizations that do not demand the knowledge to be shared. Nevertheless, in some cases the channel can be made available with some sharable property such as viewing. Knowledge can be expressed in a simple natural language using structured and unstructured language. Structured language labels the written text so that the semantic meaning is identifiable by the system without the need to perform micro processing of the sentence. The unstructured language will be treated as a block of sentence in which the semantic is unknown. The learning objects are composed of any multimedia objects such as video clips, audio files, electronic documents or web pages. They are posted to the community channel together with the text describing or summarizing their contents. The hypermedia-learning space is a collection of learning objects that can be accessed by members in the community channel. Discourse communicators are collections of software agents that form the virtual community. They perform two tasks; firstly to simulate interactions and dialogues which have been transpired in the community channel on behalf of the community; secondly, to interact with individual member in the community when query is posted to the virtual community. The discourse analyzer functions as a social network analyzer that determines the density of participation in terms of the members a well as the popularity of the discussion topic posted by the members. In building expert community knowledge, we assume that the density of participation is not a critical factor as the community of experts is people of high skills and knowledge. The discourse analyzer is built to analyze if there exist spiral of silence of some individuals or unpopular topics, which are overlooked by the community.

Knowledge building process

The two main properties in knowledge building when the community manages it, is dynamism and multiplicity. Dynamism emphasizes on the speed of knowledge creation, knowledge update and knowledge maintenance. Knowledge can be created in a fast mode as it can be represented with a simple knowledge piece such as written text. For a more complex knowledge piece, such as a scene of oil drilling or injection molding process, the

knowledge can be presented using video clips or images. The knowledge in a lengthy document with a complex concept can be easily extracted and shared with a simple annotated text attached to it. A collection of several annotated texts (posted by different members) for a document helps a new reader to understand the content without the need to spend time reading its detail. In a collaborative learning strategy, different main points highlighted by different members form a uniform understanding about the document. Knowledge is updated communally such that not only by the knowledge creator but also other members. The members can update knowledge by issuing argumentative statements such as support, disagreement, suggestion or just a general remark. Knowledge is maintained by installing security feature such as password to the knowledge unit so that vital knowledge will not be deleted. The security feature can be set by the creator of the knowledge unit or otherwise will be set by other members if the creator did not set it initially.

Multiplicity describes the varieties of knowledge sources. Gordon Bell predicts that in year 2047, all physical objects such as buildings, cars, home appliances and also humans will be online [Bell, 2004]. Taking this speculative statement into account, the learning objects are not confined to the present application software or multimedia formatted files, which are accessible by computer system but also any objects surrounding a person's life that are directly connected. In the previous paper, social values can be part of learning objects which can be extracted the evidences in the community of practice [Syed Mustapha, 2004].

Knowledge sharing process

Surrogating strategy is used for knowledge sharing where each member may ride on someone's knowledge in order to develop his/her own understanding. In order to enable the knowledge sharing process, the learning objects have to be made sharable in a learning space. Few collections of learning objects, which are attached to the text object is called knowledge unit. Each knowledge unit may consist more than one representative term, which are contributed by other members in the channel. This idea is depicted in Figure 3.

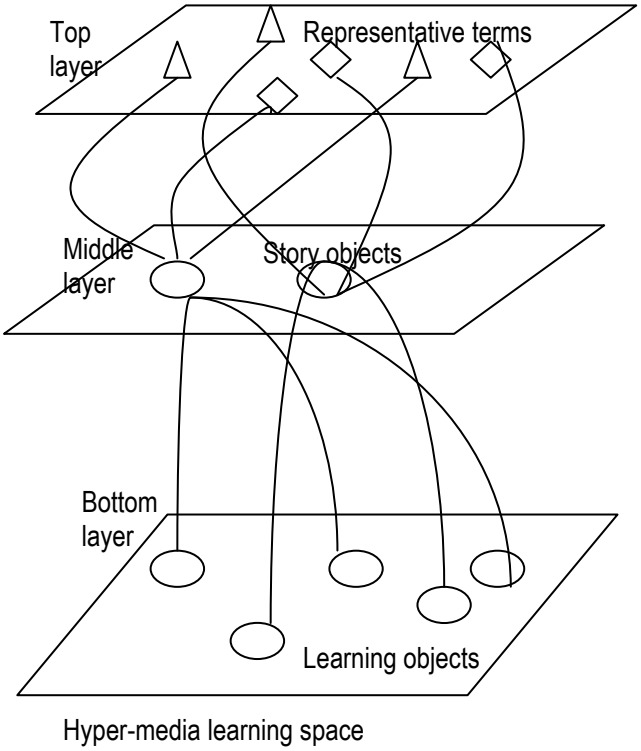


Figure 3 Schematic diagram of knowledge sharing process

The top layer consists of representative terms, which are submitted by the community members. The representative terms are the argumentative terms determined by the community such as “support”, “argue” or “suggest. They are structured language, which are machine-readable. In Figure 3, there are two examples of representative terms that are symbolized by the triangle and diamond shapes. Each representative term is associated to each story object as shown in the middle layer. Story objects have to be created before the representative terms can be linked to it. Story objects can be contributed by one or many members. In response to the submission, other members will create the representative terms (structured language) and tag it to a written text (unstructured language), this couple is called as argumentative bead that hangs to the story object. A series of the argumentative bead will continue to regenerate the knowledge cycle of a given concept in the story.

The bottom layer is the hyper-media learning space, which contains learning objects. At present, the community channel supports the multimedia objects such as video clips, audio, web documents, Microsoft office files that are commonly used by the community in sharing resources. Each story object can be associated to more than one the learning object. The learning objects are the artifacts, which act as the medium of knowledge sharing and knowledge exchange among the communities.

Knowledge conversational process

ICC creates the knowledge conversational environment through virtual community that represents the expert community. The virtual community consists of animated agents who speak on behalf of the expert community. They can be activated when query is entered into the system. The system will search for stories, which are relevant to the query and organize several agents to simulate the discussion. This session is called “interact with agent”. This approach is taken to enable knowledge to be converse in a more natural way i.e. through dialogue or question and answer. The knowledge unit in the community channel can grow very large that it is difficult to be read and searching certain entry is like searching needle in a haystack.

As the system is running on the web platform, the virtual agents that form the community and the type of discussion are personally tailored to the query being entered by the member. That means the conversational environment is different from user to user.

The conversational knowledge process is demonstrated in Figure 4. Each agent is assigned to response to a set of story objects. The sequence of the story object follows the entry order in the community channel. This replicates the actual dialogue order that is transpired in the previous session. Each agent will take turn in communicating the knowledge in the story object. The member can listen to the dialogue just like the actual conversation taken place by the real human. The conversation can be intercepted with other query and a new knowledge conversational environment will take place.

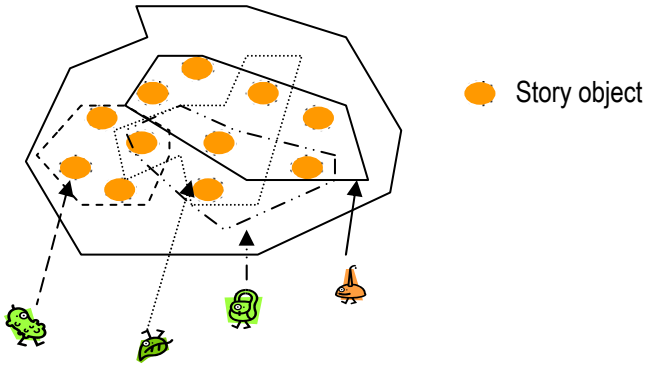


Figure 4. Virtual community in the conversational knowledge process

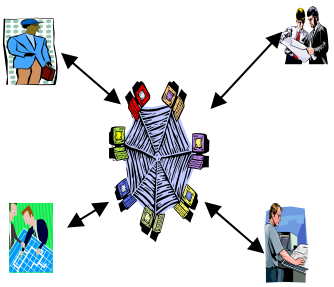
A simulated scenario of building expert knowledge community through ICC

Story-telling has been accepted as a form of knowledge exchange and sharing for scientist and engineers. Steve Denning [Denning, 2000] described an incident when Pakistan government wanted to new technology for to fix their widespread failure of pavement on the highway. The cost to maintain them is no longer affordable and new solution is needed. Since, the world bank team in Pakistan has no experience on that, they have to make a global contact with their partners. Colleagues in New Zealand reported that they had encountered similar problem in South Africa and they had developed a technology for that problem. At the same time, partners from Jordan shared their views after managed to solve the problem alike in Jordan with a promising result.

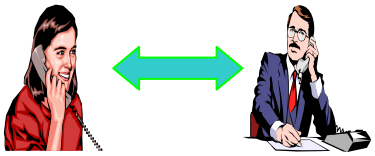
What can be learned from the above true story is that, the engineers do not share the content knowledge as all engineers are well-equipped with that knowledge through formal education and experience. Instead, they share the socially-derivable knowledge which are simply exhibited through simple discussion or video movie.

The scenario given above, the conversation can take place on the phone or video conferencing. However, the knowledge generated disappeared without being recorded. In our approach, the problems and solutions have to be stored for future retrieval. Similar to building expert system that is to retain human expert knowledge for reuse, the expert community knowledge is accumulative, retainable and referable.

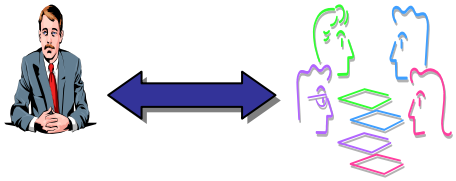
The example of how expert community knowledge can be built is given in the following three scenarios.



Scenario 1. Expert externalizing their experience, knowledge or ideas into the community channel



Scenario 2.An expert was contacted and consulted for an advice in solving a problem



Scenario 3.Expert consults the expert community knowledge through the virtual community

Scenario 1 shows the knowledge building process where experts encode their experience, knowledge or ideas into the community channel. Some may create story objects for new ideas or response to someone's idea using representative terms. Scenario 2 shows an expert was contacted by his colleague about a problem. Being a member to the ICC, he decided to consult the system. In Scenario 3, the expert acquires socially-derivable knowledge from the expert community knowledge which had been incorporated earlier.

Conclusion and Future Works

In this paper we emphasize the two different types of knowledge, which are content knowledge and socially-derivable knowledge. Many computer-based learning system such as intelligent tutoring system, computer aided learning, educational courseware are mainly to train learners to acquire content knowledge. The socially-derivable knowledge is not obtainable from formal sources such as books or formal training but rather by talking to colleagues or counterparts. Without making a claim that ICC is a substitute to other computer-based training system, ICC is rather a complement to the other training system including the traditional learning methods.

The strength of ICC in supporting knowledge construction is to allow complex knowledge that is implicit to be presented in a simple form through story-telling. Consequently, the process of knowledge extraction is made by the human effort collaboratively which adds up to the speed of system development. At the same time, the community is given freedom to design the knowledge and shape its content according to the interest of the

community. The traditional approach in expert system or case-based system only allows the knowledge to be designed according to a specific human expert when the system is developed. However, the knowledge in the ICC depends on the community who builds them continuously and the structure is fluid rather than rigid.

The system can be extended in many ways. Among them, the immediate ones are to enable agents to access the learning space where the multimedia objects can be processed and the knowledge can be extracted. The agents can assist the community by not relying totally on the knowledge extraction done manually by the community through story telling.

Bibliography

[Wenger, 1997] E. Wenger. *Community of Practice*. New York: Cambridge University Press, 1997.

[Thomas, 2004] J.C. Thomas. Fostering the Collaborative Creation of Knowledge A White Paper. Available at http://www.research.ibm.com/knowsoc/project_paper.html as of February 6, 2004.

[Thomas, et al, 2001] J.C. Thomas, W.A. Kelogg, T. Erickson. The knowledge management puzzle: Humans and social factors in knowledge management. *IBM Systems Journal, Vol 14, No 4,* pp 863-884, 2001.

[Stahl, 2000] G. Stahl. A Model of Collaborative Knowledge-building. In B. Fishman & S. O'Connor-Divelbiss (Eds). *Proceedings of the Fourth International Conference of the Learning Sciences,* pp 70-77. Mahwah, NJ:Erlbaum, 2000.

[Bell, 2004] G. Bell. The Revolution Yet to Happen, ACM2047.doc is an invited book chapter commemorating the 50th anniversary of the Association for Computing Machinery by Jim Gray and Bell Gordon. Available http://www.research.microsoft.com/~gbell/CyberMuseumPubs.htm [February 19, 2004]

[Syed Mustapha, 2004] S.M.F.D Syed Mustapha. Communicative Social Intelligence: An Analysis in Community of Practice. *Knowledge Management International Conference and Exhibition. KMICE 2004*, February 13-15, Penang, Malaysia, pp.42-52.

[Denning, 2000] S. Denning. Storytelling: The art of springboard story. Available at http://www.creatingthe21stcentury.org/Steve.html

Authors' Information

S.M.F.D Syed Mustapha *– Associate Professor in the Faculty of Information Technology, University Tun Abdul Razak, 16-5 Jalan SS6/12 Kelana Jaya, Selangor Darul Ehsan, MALAYSIA; e-mail: syedmalek@unitar.edu.my*

A "CROSS-TECHNOLOGY" SOFTWARE DEVELOPMENT APPROACH

Stefan Palanchov, Alexander Simeonov, Krassimir Manev

***Abstract**: Contemporary web-based software solutions are usually composed of many interoperating applications. Classical approach is the different applications of the solution to be created inside one technology/platform, e.g. Java-technology, .NET-technology, etc. Wide spread technologies/platforms practically discourage (and sometime consciously make impossible) the cooperation with elements of the concurrent technologies/platforms. To make possible the usage of attractive features of one technology/platform in another technology/platform some "cross-technology" approach is necessary. In the paper is discussed the possibility to combine two existing instruments – interoperability protocols and "lifting" of procedures – in order to obtain such cross-technology approach.*

***Keywords**: web-solution, interoperability of applications, interoperability protocols, lifted procedures*

***ACM Classification Keywords**: D.2 Software Engineering, D.2.11 Languages (interconnection), D.2.12 Interoperability (interface definition)*

***Conference**: The paper is selected from Sixth International Conference on Information Research and Applications – i.Tech 2008, Varna, Bulgaria, June-July 2008*

Introduction

In [Maneva, Manev, 2008] different models for development and distribution of software (MDDS) and their role for the efficiency of the developed software products were discussed. Especially it was stressed the *status quo* of the contemporary models for development and distribution of web-based business-oriented software solutions. Many negatives of the existing models were outlined that lead to high costs or to low quality of the implemented web-based solution in medium and small companies, as well as in the state administration. As a result, the necessity of a new model concept was formulated. In conclusion, the following features of such model were identified:

- It has to guarantee the **independence of the user** from the technologies, i.e. the user has to be free to chose for each component of the solution the existing technology that is the best for this component;
- It has to guarantee that the user will obtain a service with a **quality, which is relevant to the paid cost**;
- It will be very good if the model has the **tolerance for the qualification** of the users and to allow them to extend and update solution, etc.

Dependence of the users from the technologies was identified as a crucial element of the existing models. The notion *independence* is not new in the domain of development of software products for the business. For example, one of the main goals of the very popular approach Model Driven Architecture of OMG [OMG, 2008] is to liberate the process of conceptual design of the solution from the technologies of the implementation. It is used by many developers. Following the MDA concept they first design the solution in conceptual (or technology independent) level and than, automatically, semi-automatically or manually, *map* the elements of the solution in a chosen technology/platform.

It is true that MDA gives **some independence** from the technologies. More precisely MDA gives **full independence, but only in the stage of conceptual design**. In the stage of implementation of the conceptual design there are two possibilities. The fist is to implement all applications with one "clean" technology and to make the solution totally dependent from this technology. The second is to use different technologies in the different applications of the solution. If the second possibility is chosen, then some additional efforts will be necessary for homogenization of the interfaces between interoperating applications. The developers rarely do the efforts to develop the homogenization from scratch and usually rely it to corresponding software (*middleware*), making the product dependent of the middleware.

In this paper we will consider the **dependences** from the technologies, which are **generated on the second stage** of applying some concepts, similar to MDA. We will try to investigate the possibility to give to the users more independence from the technologies or middleware. The main objective is that each technology has its own positive elements as well as its shortcomings – a **single technology**, due to different reasons, **could not be ideal**. We will try to estimate the possibilities to integrate some of the best features of different technologies on the base of **homogenization of the interfaces** among languages, proposed by these technologies. We will call this a *cross-technology* software development approach.

In the second section of the paper some terminology and necessary basic knowledge are introduced. In the third and fourth section two instruments are considered that are necessary for implementation of our idea. Some advantages and shortcomings of these instruments are stressed. The idea itself is presented in the fifth section. In the last section some conclusions are given.

Basic notions

In this paper we will call *technology* (or *platform*) some general *concept* or *approach* for development of software. Part of the technology or platform are also some preliminary tailored *components* (classes of objects, small program modules – applets, servlets, etc., and even not very large "stand alone" applications) that implement the concept or approach, as well as the corresponding *tools* (programming languages, IDE, API, DB-interfaces, etc.) dedicated to support creation/integration of the software solutions (i.e. Java-technology of SUN, or .NET-technology of Microsoft).

With the term *dependence on the technology* we will denote different kind of limitations that the users have to obey if choose specific technology/platform for creating/integrating the necessary solution. For example, any attempt of the user for appending new functionality to the solution, developed with a specific technology, has to be implemented "within" the technology. Issuing of a new generation of the elements of the technology could lead to necessity of total upgrading of all bought to the moment elements and, probably, reintegrating of the solution. And more, when an element of the technology is of low quality, comparing with the concurrent products with the same purpose, it is very difficult to eliminate this element from the solution and to replace it with a better one.

Following [Ousterhout, 1998] we have to agree that the contemporary web-programming is really *gluing components* (GUI-components, small applications providing content or services, etc.) in a *solution*. The components are usually written in some *system programming language* (like C, C++ or Java) and as a gluing instrument different *scripting languages* (like Unix-shells, Java Script, PHP, Perl, Tcl, Python etc.) are used. It is possible that some components are also written in a scripting language. The opposite, gluing of the solution with a program written in system programming language is rather nonrealistic – these languages are not dedicated to such purpose.

One of the main objectives for the promoted in this paper cross-technology approach is that the components written in different languages (both system and scripting) are not always able to *cooperate in run time*. Developed technologies/platforms resolve this problem with some "**inner",** built-in, **interoperability** of one or more system programming languages with one or more scripting languages (C – Unix-shells, Java – Java Script, etc.). We are proposing a way to achieve such **interoperability** in a cross-technology level, i.e. **among languages for which built-in co-operability mechanism does not exist**.

Interoperability protocols

One of the possibilities for achieving run-time interoperability of components is to use some *interoperability protocol.* Examples of such mechanism for achieving interoperability of different processes written in C/C++ that even could work on different computers, is Remote Procedure Calls (RPC), developed for Unix-like operating systems by Sun [Marshal, 1999]. For applying RPC a unique number is assigned to each interoperating program

and, for a given program, a unique number is assigned to each procedure inside the program. Call of a remote procedure is made trough a corresponding *RPC-client* (executing function `rpc_call()` in the calling process). Beside the traditional list of parameters of the called procedure, many other parameters have to be provided also – the identification of the host, the unique numbers of the program and of the procedure that is called, etc. The calling program is waiting as usual to obtain the result or some indication that the remote call failed (timed out, for example) and then continues the work.

The RPC-client is responsible for serialization of the given arguments of standard types, i.e. translating them to an inner RPC form, which is suitable for transportation in the net. Serialization of user defined types is responsibility of the user. For this purpose RPC provide a set of standard procedures and the corresponding procedure has to be called for each included in the user defined type variable of standard type. The request is composed following the rules of the protocol, sent to the host and processed by *RPC-server*. The server performs deserialization of arguments, identifying and calling the procedure and serialization of the obtained result, which is sent back to the RPC-client. Finally, the RPC-client performs deserialization of the result and returns it to the calling process.

The system RPC is developed to support communication among processes, the code of which is written in C/C++. There are samples of systems for generating interoperability protocols for other languages, well checked and proved as mentioned RPC – RMI for Java [Java RMI, 2007], RPyC for Python [RPyC, 2007], CORBA, SOAP, etc.) – that could also be used as a model. There is no popular sample of protocol generating system that is able to provide interoperability of components, written in different languages. For achieving interoperability of processes, the code of which is written in different languages, additional efforts will be necessary and we will discuss them below. We will take some existing systems as models and will try to extend the idea to a system, which is able to provide interoperability protocols for processes written in different languages – *homogenization protocols*.

The advantages of such homogenization protocols are obvious. They could be a first step toward obtaining a total independence of the developer from the technologies or platforms. In such way each component of the solution could be created in the most appropriate language, within most favorable technology or platform. Different components could be executed on different machines and even under different operating systems.

Obvious shortcoming of the homogenization protocols will be the significant amount of time, necessary for the execution of the procedure call. Each call is passing through a complex process of structuring and parsing of the request, serialization and deserialization of arguments and results, etc. As a result the additional time could be many times bigger than the time necessary for the local call. That is why such kind of homogenization is inappropriate for relatively small and simple tasks. There is no sense to organize a remote call (especially to a procedure written in another language) for finding the sum of two integers, for example. Homogenization protocols have to be used only for requesting services that could not be obtained locally or could not be obtained in reasonable price.

Some negatives of the approach could be observed on the example of RPC also. The function `rpc_call()` that performs remote call has 8 arguments due to the rules of he protocol – coding of such call could take time and the probability for giving a wrong argument is significant. One of the arguments is the identification number of the program, which contains the called procedure, and, if the system is implemented as in RPC, the developer has to keep in mind large amount of such identification numbers (in RPC the numbers reserved for programmers are from 0x20000000 to 0x3FFFFFFF). And finally, we described above a very simple scheme of RPC. Really, the approach is much more complex and could be used only by very experienced programmers. All these negatives are not generic and could be surmounted in one well planned implementation.

"Lifted" procedures

On the road for achieving cross-technological interoperability of components written in different languages, we will use one other approach too that we will call *lifting of procedures*. As it was mentioned above, for creation and

integration of software solution, languages of different level of abstraction are used. The idea is to choose one language as a basic, to create a library of necessary procedures in this language and than to "lift" each procedure to each of the other languages being in use. With "lifting" of the procedure we will denote the process of making basic procedures accessible from programs written in languages different from the basic language. The assumption is that the basic language is of the lowest level among all used languages. That makes the candidate for a basic language almost unique – the C language. The level of C is low enough. C++ and Java, as well as many of the most used scripting languages inherit the syntax of C and are appropriate for the "lifting" process.

Example of a tool for lifting of procedures is the system SWIG [SWIG, 2007]. It is a typical "open source" project developed and maintained by some enthusiasts on voluntary principle. As mentioned on the official page of the system: "SWIG is a software development tool that connects programs written in C and C++ with a variety of high-level programming languages." Nowadays the most popular scripting languages as Perl, PHP, Python, Tcl and Rubby, as well as non-scripting languages as Java, C#, Common Lisp, etc. are supported by SWIG.

The system SWIG was written initially (by Dave Baezley in 1995) for C as a basic language. In 1996 the system was rewritten for C++ as a basic language. Obviously a bit more high level of C++ was not suitable for the goals of the system because since version 3.1 the software migrated back to C. This confirms our observations that the basic language has to be of as low level as possible.

The idea of lifted procedures originally was to ameliorate the performance of programs written in a specific scripting language, providing a mechanism for calling procedures written in the language C, through which the scripting language was interpreted. The idea happens to be very helpful and soon some versions for other scripting languages interpreted trough C or C++ were created. Finally, the idea happens to be universal and soon versions for non-scripting languages and, which is more important, genetically not connected to C and C++ appeared too.

The lifting of procedures practically has no shortcomings beside the fact that for each new language a specific module of the system has to be created. Fortunately the most of used in the contemporary web-solutions programming languages are supported by the current version of SWIG; the software is relatively wide spread and well tested. A few small shortcomings could appear from the peculiarities of some language that could make one universal lifted procedure inefficient in this language. In such case a specific version of the procedure has to be written for each such language.

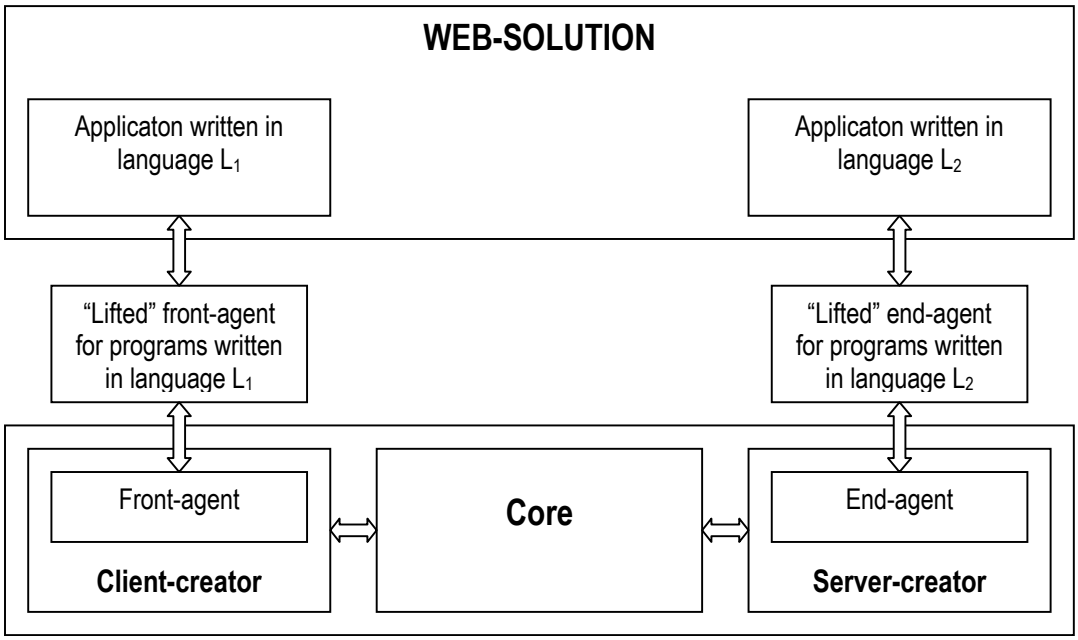
There is an objective that we have to keep in mind when plan to use "automatic" tools like SWIG. It is quite possible that such automatic tool does not support all constructions of the basic language. For SWIG and C language this seems not to be true, but for SWIG and C++ such problems exist. Fortunately, the systems like SWIG are with open code. This gives a possibility to qualified users to re-develop some specific modules in order to solve some specific problem and, as a result, to contribute to extension and amelioration of the tool.

"Lifted" interoperability protocols

The general idea of this work is to mix the two approaches – interoperability protocols and "lifted" procedures – in order to obtain interoperability of applications written in different languages. As a beginning, let us take some tool for achieving interoperability through interoperability protocols. It could be RPC or some modification of RPC, if some features of RPC are not suitable for implementation of the idea. It could be some other tool with the same functions also. Finally it is possible, following the model of RPC and another existing system, to create a new tool for implementation of interoperability protocols. Let us call this tool *Basic Interoperability Protocols* (BIP). The language of creating of BIP has to be as low as possible – most probably it will be the C language.

Schematically BIP could be split in three parts (see the Figure) – *BIP-client creator*, *Core* and *BIP-server creator*. The Core part is independent of any used language and is dedicated to provide a transportation mechanism between applications (including applications that are working on different computers). The BIP-client creator and the BIP-server creator are dedicated to enable communication of the Core with calling procedure and called procedure, respectively. Both creators could have a part, which is independent of the used languages. But the

essential for them is the part that depends of the language of calling/called procedure. It is very probable that these parts will have more than one version (because of the mentioned above particularities of the used language) but we will prefer to refer them as integral elements of the system – the *front-agent* and the *end-agent*.



Figure

The front-agent and the end-agent of the BIP system (really they are set of written in C functions) are lifted to the level of all used languages. Initially the lifting could be maid by existing SWIG processors. Some additional *lifters* written in SWIG-style could be created if necessary (for example, when some of the existing SWIG processors are not appropriate for BIP, or when a language not supported by SWIG is used).

Really, lifting of front-agent and end-agent are different kind of processes. It is possible to say that lifting of the front-agent is a classical use of SWIG-like mechanism and lifting of end-agent is an attempt to extend the possibilities of the tool with a new functionality that was not initially presumed. Lifted front-agent has to provide to procedures written in high-level language a possibility to call procedures written in low-level language. In our case these are the procedures of the interoperability protocol that have to transport the remote call to the called procedure. Lifted end-agent practically makes the opposite – provides to procedures written in low-level language a possibility to call procedures written in high-level language. In our case these are the called procedures. It will be more correct to say that the end-agent is "taking down" the level of these procedures.

Because initially SWIG-like mechanisms were not designed for taking down the level of procedures, this process will not be as easy as lifting of the level. Anyway, there are mechanisms for solving the problem and one of them is the "callback" mechanism. For some reasons, different of discussed in this paper, the callback of procedures was implemented in SWIG and, even if it is not working very smoothly (see for example [SWIG, 2002]), it could be used for our purposes.

Conclusions

The discussions and the innovative ideas presented in this paper are results of some experiments. The current versions of RPC, as a generator of interoperability protocols, and SWIG, as an instrument for lifting of procedures, were used. First, some experiments with SWIG system were made. Procedures (containing relatively heavy calculations) were written in C/C++ and were lifted to some of the most popular scripting languages (PHP,

Perl and Phyton). The lifting process passed smoothly and the results were encouraging – using of lifted procedures led to significant decreasing of execution time, compared with the time necessary for same calculations but written in the corresponding scripting language.

The second experiment was dedicated to execution of call (local, not remote) of procedure written in C from procedure written in scripting language. One of the standard interoperability protocols, created with RPC, was extended to a front-agent and lifted to Python-level. Then a procedure written in Python called successfully a procedure written in C. Some experiments for remote call of procedure written in C from procedures written in scripting language are in progress. It is clear, that more efforts will be necessary for implementing of the end-agent.

As a result of experiments we could make the following conclusions:

- Proposed in the paper approach is **quite realistic** and could be used **for achieving cross-technological interoperability** of the applications in a web-based software solution;
- The proposed approach is **very promising in sense of time consuming** and could be much more appropriate than some other approaches – for example, using XML as an interoperability mediator;
- The proposed approach **could be implemented** with minor extensions of the existing tools for creation of interoperability protocols and lifting of procedures. RPC and SWIG are very good base for start of the implementation;
- Both discussed mechanisms are **not easy** and their usage will be a true challenge for some developers. That is why a corresponding interface (shell) for ordinary users has to be provided too.

Bibliography

[Maneva, Manev, 2008] N. Maneva, Kr. Manev. On the models of development and distribution of Software, Interntional Journal of Information Theories and Applications, No. 15, 2008.

[OMG, 2008] Model Driven Architecture, http://www.omg.org/mda

[Ousterhout, 1998] Scripting: Higher Level Programming for the 21st Century. IEEEComputer, March 1998.

[Marshal, 1999] D. Marshal. A tutorial on ONC RPC, http://www.sc.cf.ac.uk/Dave/C/node33.html, May 1999.

[Java RMI, 2007], The Java Tutorials. RMI. http://java.sun.com/docs/books/tutorial/rml, 2007.

[RPyC, 2007] Remote Phyton Call (RPyC). http://rpyc.wikispaces.com, 2007.

[SWIG, 2007] Welcome to SWIG. http://www.swig.org, 2007.

[SWIG, 2002] SWIG :Pointers to functions and callbacks, http://www.garyfeng.com/wordpress/2002/11/27/swig-pointers-to-functions-and-callbacks/, 2002.

Authors' Information

Stefan Palanchov *– Manager; STEMA-SOFT, St. Ivan Rilski, No. 27, vh. A, Varna-9009, Bulgaria; e-mail: s.palanchov@stemasoft.com*

Alexander Simeonov *– Manager; Inovative Web Solutions, Mladost 4, 448, vh. 3, Sofia-1715, Bulgaria; e-mail: simeonov@stemasoft.com*

Krassimir Manev *– Associated Professor, Faculty of Mathematics and Informatics, Sofia University, 5 J. Bourchier str, Sofia-1164, Bulgaria; e-mail: manev@fmi.uni-sofia.bg*

POSITION PAPER: IMPROVING SOURCE CODE REUSE THROUGH DOCUMENTATION STANDARDIZATION

Rubén Álvarez-González, Sonia Sanchez-Cuadrado, Héctor García

***Abstract**: In the context of Software Reuse providing techniques to support source code retrieval has been widely experimented. However, much effort is required in order to find how to match classical Information Retrieval and source code characteristics and implicit information. Introducing linguistic theories in the software development process, in terms of documentation standardization may produce significant benefits when applying Information Retrieval techniques. The goal of our research is to provide a tool to improve source code search and retrieval In order to achieve this goal we apply some linguistic rules to the development process.*

***Keywords**: Software Reuse, Information Retrieval, source code documentation.*

***ACM Classification Keywords**: Reusable software, Information theory, Documentation.*

***Conference**: The paper is selected from Sixth International Conference on Information Research and Applications – i.Tech 2008, Varna, Bulgaria, June-July 2008*

Introduction

In the area of Software Reuse research projects have obtained promising results since the decade of 90's [Prieto-Díaz, 1991]. Some major concerns are reducing effort and cost, increasing competitiveness, through reusing documents, models or source code. The first steps in Software Reuse consisted of reusing source code. Since then, researchers have been increasing continuously the possibilities of reuse. However much work is still required to provide proper methods, techniques and repositories allowing source code reuse.

As long as source code is basically text, the application of Information Retrieval (IR) techniques is a common approach. These techniques have proven its usefulness for purposes such as document retrieval. The similarities between text documents and source code make IR a good candidate to support source code reuse.

However the application of IR techniques to source code retrieval is not exempt of troubles. Reserved words of languages are to be considered as empty words in IR, while some of them could be important in Natural Language Processing. Programmers may define meaningless identifiers that are considered correct in the context of compiler theory; however this does not help in applying IR techniques. The difficulties found while automating semantics processing of source code are also a matter of fact.

In our research we are proposing a tool to ease retrieving source code (e.g. classes) from a repository. As long as many management applications are developed under the Object Oriented paradigm we chose to focus on it, particularly the .NET framework.

The purpose was to define a repository to store the source code from developed applications, already tested, after checking that they are consistent enough to be reused. A search engine is to be deployed to allow searching the repository. IR is applied to comments, once they have been written compliant to the standard we have defined. Also we focus the search in source code (e.g. class name, method signature). The purpose is to provide the users the capability to find non exact matches, but to retrieve variations on the query terms (i.e. words with similar morphology). Finally, ranking techniques are applied to the results.

Related work

Some applications have been deployed in order to provide to users source code retrieval capabilities. Swik or Google Code Search are applications oriented to find full open source code to Internet users. As long as these applications are focused on the community it is difficult to use them within an organization where sharing code is not an option.

The work by Sindhgatta [Sindhgatta, 2006], named JSearch, provides a plug-in for Eclipse development environment. The goal is to provide programmers the location of source code that may be used as an example, useful while developing new applications. The source code assets are already developed libraries or classes, in the context of the same organization. However this may not be considered as Software Reuse.

JSearch applies IR techniques, modified for the specific purpose of source code retrieval. The tool provides a source code transformation module. This module extracts and modifies some features in order to properly model and index a class or library. It is focused on users that know exactly what they look for, but have little knowledge on how to use them. This is to say, it finds code assets that use a specific class or method. Then, if a user looks for functionality instead for a similar example the system returns unexpected results.

The same situation is found in the results obtained by [Krugle, 2008] and [Koders, 2008]. Both tools assume that the programmer have deep knowledge of the structure of the source code to be located.

In a more general context, when the exact name of a class or method to be found is not known, comments may be useful to override the problem. As long as comments should describe briefly the functionality provided by a source code asset they can be used to understand source code. The main concern is that it can not be assumed that comments are correct or represent properly the functionality. This fact affects significantly to the goodness of obtained results.

The work by Ying, Wright and Abrams [TT Ying, 2005] describes a feasible classification for the different types of comments in source code. In the case of source code search engines described above the comments such as “do not delete this line or execution shall produce errors” are considered as proper comments. It will not help in obtaining appropriated results. This is why good comments are a must if it is desired to provide support for Software Reuse.

Due to the limited IR capabilities of the tools available at Internet we described above we evaluated [Lucene, 2008] and [Google Desktop, 2008]. The purpose of the evaluation was to infer which third-party IR capabilities may be reused for the purpose of source code reuse. We chose these tools because they can be used in both local and centralized environments, and because we did not incurred in licence fees.

These tools provide the capability of finding documents containing syntactically close to those in query criteria. The test consisted on designing a corpus that consisted on a set of source code documents. After indexing the corpus three different query types were executed:

- Queries with only one search term that is located, at least, in an indexed document. The purpose is to check if the results contain documents with exact term matching.
- Queries with a term containing the same root of the first query (e.g. plurals, derived words). The purpose is to check if the results of the first query are included in the results of the second.
- Queries with shorter forms of those in the first query. The purpose is to check the deviation on the results derived from loosing specificity.

With Google Desktop we needed to use only two Spanish terms: “entrada” and “entradas” (in English “input” and “inputs”). With de first term the search engine found one document. But with the second term, it could not found it. This, drove us to conclude that Google Desktop looks for exact term matching.

The same results were obtained during the tests for Lucene. Also we used two terms, in this case “Developer” and “Developers”…

Software reuse, documentation and linguistic theories

In this section we describe the tool we are proposing in order to improve the results of IR applied to source code retrieval. First we will show the architecture of the tool, and then we are to analyze the characteristics of a high quality comment. Finally we suggest some techniques which application may lead to the improvement of the results.

The tool is intended to receive, from programmers, a text file containing the definition of one or more classes. Once the file has been received each class is separated from the rest in the same file. Each class is processed separately, obtaining a model of the class in the form of a digest document. Information retrieval techniques are applied to these documents. The information in documents contains:

- Class name
- Classes in import clauses
- Names of the inherited classes
- A digest from method signature
- Comments from source code

Comments to be considered have been reduced in order to improve the results for queries looking for functionality instead for specific resource usage. For such a purpose selected comments are those compliant to .NET style from Visual Studio 2005. Figure 1 shows the structure of a source code asset. Imports, identifiers and selected comments have been highlighted.

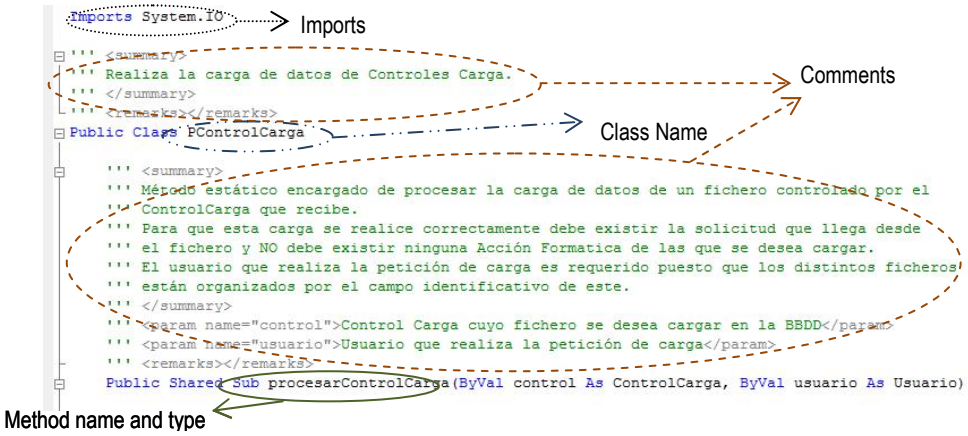


Figure 1. Commented Visual Basic .NET source code asset

In Figure 1 the structure of .NET comments is shown. In the particular case of method comments an extra advantage is obtained. The text is the one shown when debugging code in Visual Studio, so most programmers tend to describe widely the semantics of the functionality. Then it may be reduced the effort required to integrate the tool in the development process.

The architecture of the tool is divided in three main modules. Code Incorporation Module is in charge of uploading source code files, transforming them into documents and including documents in the repository. Documents Retrieval Module is in charge of executing queries over the corpus through IR techniques. Finally, Code Retrieval Module is in charge of processing the results obtained and get the source code related to each document and provide it to the user. In figure 2 a schema of the architecture is shown. User may upload source code assets to Code Incorporation Module. This module stores the code in the repository used by Document Retrieval Module, which sends the results of queries to Code Retrieval Module. Code Retrieval Module returns the code to the user.

In this process, as we mentioned, it is essential that comments available do not contain arbitrary semantics. Getting high quality comments depends on understanding the purpose of comments. Ambler [Ambler, 2000] establishes that the goal of source code documentation (i.e. comments) is to provide a clear idea on code functionality and design. Hence, a comment is of a better quality the better it eases code comprehension.

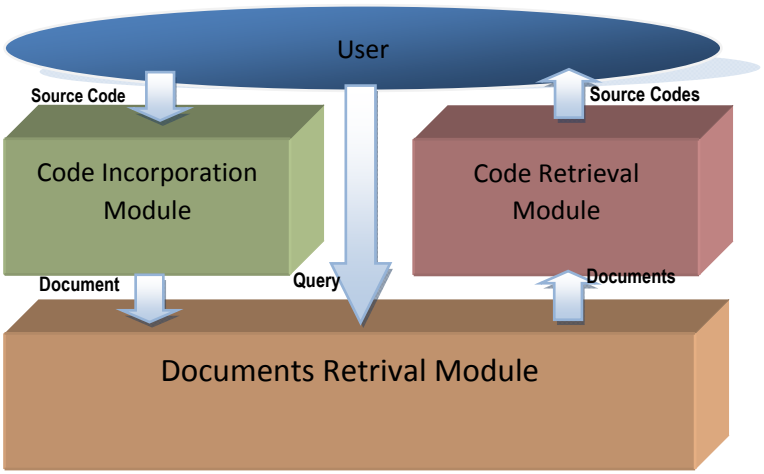


Figure 2. **Basic architecture**

Main aspects to consider when deciding if a comment is a quality one or not are:

- Comment contents depend on comment type
- The proper granularity level of comments is not related to comment length, but on the precision. Comments should include description of technology, functionality and design criteria.
- How comments are written. The better comments are expressed the better the quality of the comments. Incorrect or ambiguous expressions in comments may be worst than writing no comments, as long as they may affect to further maintenance.

Expressing correctly the comments is, then, crucial for further comprehension. Gutiérrez [Gutiérrez, 2007] provides some suggestions on how to write leaflets. These suggestions are of interest, as long as comments are close to leaflets in the sense of providing critical information in a concise manner. This means that quality comments are easy to read and understand, so they meet some requirements:

- They are written using simple terms
- Phrases are as short and clear as possible
- Phrases are written using active form, instead of passive form. Concatenation of sentences, pronouns, and similar linguistic resources are avoided. This is particularly important when writing in Spanish, because it is widely believed that long complex phrases indicate higher knowledge of the language.

Hence, correct writing implies ordered phrases subjected to the basic subject-verb-predicate form.

However some considerations are to be kept in mind regarding the difference from a leaflet to source code. Technical documentation shall be non ambiguous, while leaflets may not comply it because of their nature. Ambiguity is a marketing technique to attract customers asking for additional information. Attracting programmers asking about comment semantics is not the marketing we look for in Software Reuse. Only a meaning should be obtained from a given comment, this is the goal that leads to avoid the following elements from a language:

- Conditional forms of verbs. No speculation may arise, the subject of the description already exists and its behaviour is deterministic.

- Ambiguous terms, such as “further”, “some”, open lists, etc. These are the terms that we use to avoid while writing technical documents (e.g. software requirements specifications). We shall keep in mind that source code comments are a part of technical documentation.

TxReadability [University of Texas, 2007] evaluates the legibility of a given text in Spanish, English or Japanese. The feasibility on using this tool was previously shown by Gutiérrez [Gutiérrez, 2007]. Tools such as FLAVER [Santana, 1997] are capable to detect automatically the verbal form used in a phrase.

Finally it is required to decide which techniques are to be used by Documents Retrieval Module. To support expected features it is needed to provide a corpus containing grammatically categorized terms from Spanish languages [Sebastián-Gallés, 2000], using those tags from EAGLES [Monachini, 1996].

The Stopwords technique is to be used in order to optimize search. This reduces the size of the term index. An empty word does not provide useful information when selecting a document from the corpus. These techniques are implemented using EAGLES tags. Anyway we are to store the full text in order to provide exact matching capabilities.

In order to avoid restrictions derived from exact matching we decided to implement stemming. Stemming identifies the root of each term to search [Brants, 2004]. Roots are used to keep in mind, during search process, terms similar from those on the search criteria. The reason to avoid using directly the roots as search criteria is to distinguish two terms sharing the same root. This allows ranking on upper positions documents matching exactly search criteria.

Vectorial model [La Serna, 2004] has been selected to rank search results. Applying this technique requires correlation between terms and documents. On the one hand, we store the terms in a document, as well as the number of occurrences of the term. On the other hand, we need to store also the position of each occurrence. It is also required to keep in mind that those terms written in different forms, or belonging to different grammatical categories, are processed as different terms.

Conclusions

The tool described in this work intends to reach two goals. First goal consists of improving the quality of the comments within the source code. Second goal is to ease the introduction of the reuse culture in organizations, focussing on developers. In some cases programmers are who reject proper software reuse, while they receive the benefits of source code reuse.

Increasing comment quality relies on analyzing the characteristics that a good comment shall comply. The analysis we conducted allowed providing to programmers, at Technical University of Madrid, guidance on how to document the source code.

The proposal of constructing a tool meeting exposed requirements allows creating a repository containing source code. The purpose of the repository is properly reusing code safely, and locating source code through searching for functionality instead for specific and exact terms.

Further research, while prototypes are under development, is to solve the problems derived from supporting a wider spectrum of programming languages.

Bibliography

Ambler, S.W.: Writing Robust Java Code, White paper, 2000. Available at Internet < http://www.ambysoft.com/downloads/javaCodingStandards.pdf> [ref. march, 29th 2008]

Apache Software Foundation: Lucene library. Available at Internet <http://lucene.apache.org/> [ref. march, 29th 2008]

Brants, T.: Natural Language Processing in Information Retrieval. Proceedings of CLIN, 2004, pp. 1–13.

Damiani, E.: A descriptor-based approach to OO code reuse, IEEE Computer Society, 30, 1997

Google, Inc.: Google Desktop. Available at Internet <http://desktop.google.com/> [ref. march, 29th 2008]

Gutiérrez, U., Blanco, A., Casal, B., Calvo, A. y Ramos, A.: Information: new times, new media outlets, new staff (in Spanish). In Proceedings XII Jornadas Nacionales de Información y Documentación en Ciencias de la Salud. Zaragoza (Spain), 24, 2007, p. 26.

Koders: Open Source Code Search Engine. Available at Internet <www.koders.com> [ref. march, 29th 2008]

Krugle, Inc: Code search for developers. Available at Internet <www.krugle.org> [ref. march, 29th 2008]

La Serna, N., Román, U., Osorio, N., Benito, O., Espezúa, J. y Vega, H.: Estudio y Evaluación de los Sistemas de Recuperación de Información, RISI, 1(1), 2004, pp. 49-58.

Monachini, M. y Calzolari, N.: Synopsis and comparison of morphosyntactic phenomena encoded in lexicons and corpora a common proposal and applications to European languages, 1996

Prieto-Díaz, R.: Implementing faceted classification for software reuse, Association for Computing Machinery, Inc., 34, 1991

The University of Texas Accessibility Institute, 2007 June 29, 2007-last update, TxReadability. Available at Internet: <http://www.lib.utexas.edu:8080/TxReadability/app> [ref. march, 29th 2008]

TT Ying, A., L. Wright, J. y Abrams, S.: Source code that talks: an exploration of Eclipse task comments and their implication to repository mining. MSR '05: Proceedings of the 2005 international workshop on Mining software repositories. 2005, pp. 1-5.

Santana, O., Pérez, J., Hernández, Z., Carreras, F. y Rodríguez, G.: FLAVER: Automatic stemmer and inflectioner of verbal forms (in Spanish). Current Spanish Linguistic, 19(2), 1997, pp. 229-282.

Sebastián-Gallés, N.: LEXESP: Computerized Spanish lexicon (in Spanish). Edicions de la Universitat de Barcelona, Barcelona (España), 2000

Sindhgatta, R.: Using an information retrieval system to retrieve source code samples, ACM New York, NY, USA, 2006, pp. 905-908.

Authors' Information

Rubén Álvarez-González – *Researcher. Technical University of Madrid. E.U. Informática. Ctra. de Valencia Km. 7. E28031 Madrid. Spain. e-mail: ruben.alvarez.gonzalez@gmail.com*

Sonia Sanchez-Cuadrado – *Carlos III University of Madrid. Avda. de la Universidad 30, E-28911 Leganés, Madrid, Spain. e-mail: sonia.sanchez.cuadrado@uc3m.es*

Héctor García – *Adjunct Professor. Technical University of Madrid. E.U. Informática. Ctra. de Valencia Km. 7. E28031 Madrid. Spain. e-mail: hgarcia@eui.upm.es*

A LOG TOOL FOR SOFTWARE SYSTEMS ANALYSES

Igor Karelin, Boris Lyubimov, Tatyana Gavrilova

***Abstract**: The article presents a new type of logs merging tool for multiple blade telecommunication systems based on the development of a new approach. The introduction of the new logs merging tool (the Log Merger) can help engineers to build a processes behavior timeline with a flexible system of information structuring used to assess the changes in the analyzed system. This logs merging system based on the experts experience and their analytical skills generates a knowledge base which could be advantageous in further decision-making expert system development. This paper proposes and discusses the design and implementation of the Log Merger, its architecture, multi-board analysis of capability and application areas. The paper also presents possible ways of further tool improvement e.g. - to extend its functionality and cover additional system platforms. The possibility to add an analysis module for further expert system development is also considered.*

***Keywords**: Knowledge Base, Software Systems Analyses, Log Tool, Telecommunication System.*

***ACM Classification Keywords**: B.8.0 Hardware – Performance and Reliability - General, D.2.5 Software – Software Engineering - Testing and Debugging, H.3.4 Information Systems - Information Storage and retrieval - Systems and Software.*

***Conference**: The paper is selected from XIVth International Conference "Knowledge-Dialogue-Solution" KDS 2008, Varna, Bulgaria, June-July 2008*

Introduction

Nowadays Telecommunication Infrastructure feels growing consumers' demand for high performance systems due to the changed character of traffic and increasing number of subscribers. The situation requires the particular level of system reliability and availability - both are the essential characteristics of a modern telecommunication system.

Such strict requirements should be supported by appropriate architectural solutions, such as redundancy of all the elements in the platform and mechanisms (both in hardware and software parts of the platform) providing rapid automatic recovery from failures.

But the growth of a system leads to the increase of data level required for comprehensive systems state analysis and supervision and life cycle description. The majority of this information is represented as log files – text files consisting of time-stamped status and error messages detailing the operational history of a given piece of software.

As it was already mentioned system state and lifecycle are described by the huge amount of jumbled data produced and distributed by multiple software units. These data represent the behavior of each unit on a long time scale.

The problem is that during system analysis (failure investigation for ex.) the search for information is time-consuming. Maintenance engineer should manually filter and sort data from all the log files to assess system state and its behavior. The situation can be more complicated in case of log files allocation in different network nodes (multi-board systems). The developed software tool helps the raw data to be automatically collected, analyzed, reordered and filtered according to engineer's needs in each particular case.

Further along in this paper, we will consider the existing methods and tools and share a new approach that has been successfully used by the authors in their work.

Overview of the existing methods for log capturing and analysis

Nowadays there is a great amount of systems, which are focused on the log capturing and analysis. Generally these programs are meant for solving this problem in definite fields. For example there is a great deal of such systems oriented on Web Log File Analysis. One of them FlashStats 2006 [4] analyzes web site's log files and provides comprehensive information about the web traffic. Another one is widely known site optimizing system Semonitor, which includes a proxy server logs analysis application ProxyInspector [5].

We can name a lot of systems like these, but every system type has its own advantages and disadvantages, as well as limitations, such as ability to work with local database only, or not enough accuracy and work speed.

All in all there is no common approach to the problems of information capturing, filtering and analysis from log files. It can be partly explained by the fact that every field of science or production has its own system architecture. It is precisely this fact that explains the necessity of a new tool creation.

So the article deals with the tool, focused on gathering, filtering and useful presentation of information from log files, found in the complex telecommunication system.

Architectural Approach

1. Experimental Approach

It is true to say that there is no value in simply collecting raw log data, the value comes in how it is presented, structured and how clear the problem is from that data.

What is the approach?

The system consists of several boards of different types to provide functionality with the required level of reliability, which works under Linux system. Each board contains particular set of tasks depending on its type, which supports required functionality.

According to this information it was decided to take next fields as primary keys for log events:

- Board name (type + slot number)
- Task name
- Timestamp

Setting up the filter parameter board name and task name user is able to choose boards and tasks from which logs are likely to be gathered. It is extremely important option as there are a lot of boards and tasks in the system. Also the time interval can be set in order to collect only needed information as well as white and black lists are assigned for the same purpose.

White list can be used to gather only logs which contain particular word(s). For example, if you want to analyze logs where the word “error” appears you should write it down in the white list. On the other hand, if you do not want some words to be contained in logs you should write it down in the black list.

2. Development tools selection

From the very beginning it was decided that an execution should be supported not only on a target platform, but also on local machines with a provided log folder or an archival file (for more info see section 3). These platforms work under different Unix/Linux systems. So it was decided to take Perl which obvious advantage is that there is no need to compile different executables for various platforms. Perl script can be easily modified. Perl is good for fast development in this area and is obviously aimed at work with text structures. Also Perl Tk allows easily create required GUI.

3. Overview of Log Merger Architecture

The simplified system architecture is presented in Figure 1.

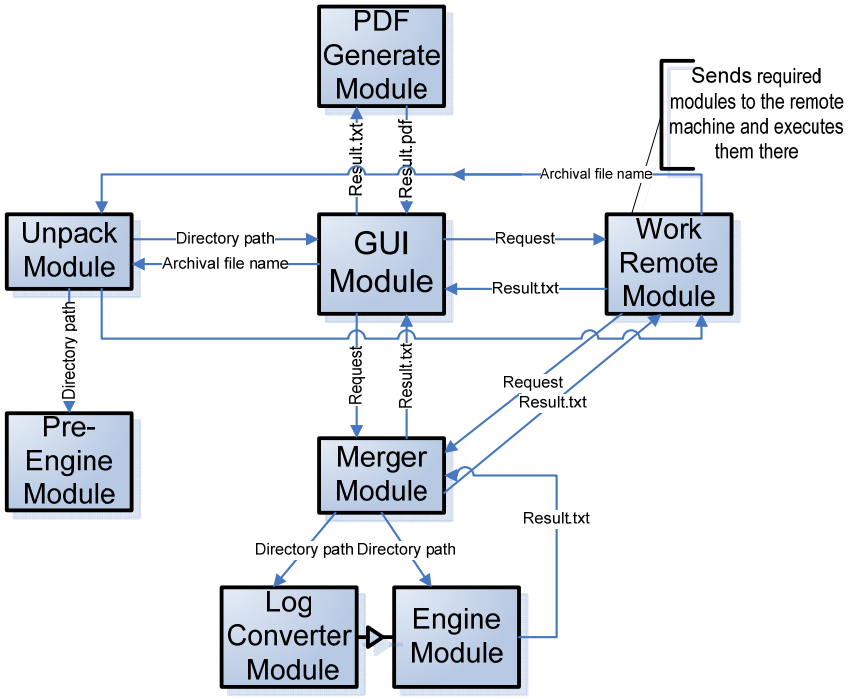


Figure 1. Log Merger architecture

1) GUI module

GUI collects the user information, creates filters and executes Pre-Engine and Merge Engine scripts to receive merge file. After this file is received it is represented in text window and user can browse and modify it. Also there is no need to run remerge process if user wants to reduce number of observed tasks and boards. GUI allows switching on/off events from tasks and boards of initial filter, which was used during merge procedure previously, without rerunning engine again. At the end user can save a final file in PDF format with different colors for various types of events (board + task) – this functionality is provided by the PDF Generate module.

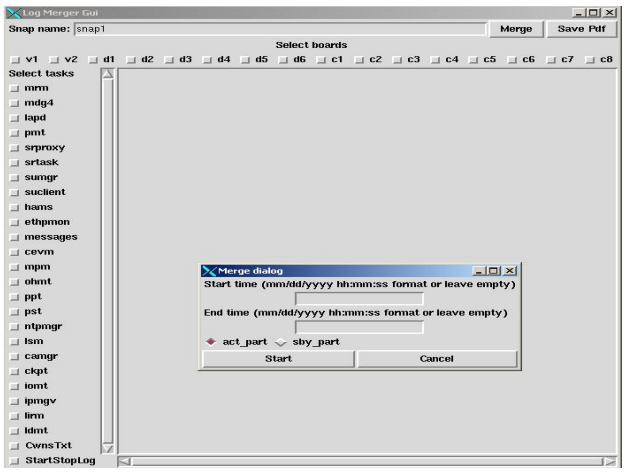


Figure 3: Graphic User Interface

2) Unpack, Pre-Engine and Log Converter modules

One of the most important problems for this type of tool is reusability. Different systems have different folders and logs structures. Moreover, in some systems log files are dumped into one archival file. So it was decided to separate out the following functionalities – Unpack module which unpacks archive file and returns the name of the directory with a specially organized file set and Pre-Engine and Log Converter modules which will convert different files to one input format for Engine module. It was also decided to implement a rule engine which will process files according to the rules. At the moment these rules are processed one after another, and the possibility to use Rete algorithm [1][2][3] is reasonable to be investigated in further studies.

3) Engine

There are two possible sorting modes:

- Ordering by timestamps
- Ordering by tasks, boards and then ordering by timestamps

For sorting by timestamps we used heuristic algorithm (See Figure 4) based on the statement that each log file is time ordered inside itself. So we receive the list of time ordered log files and should merge them according to the rules.

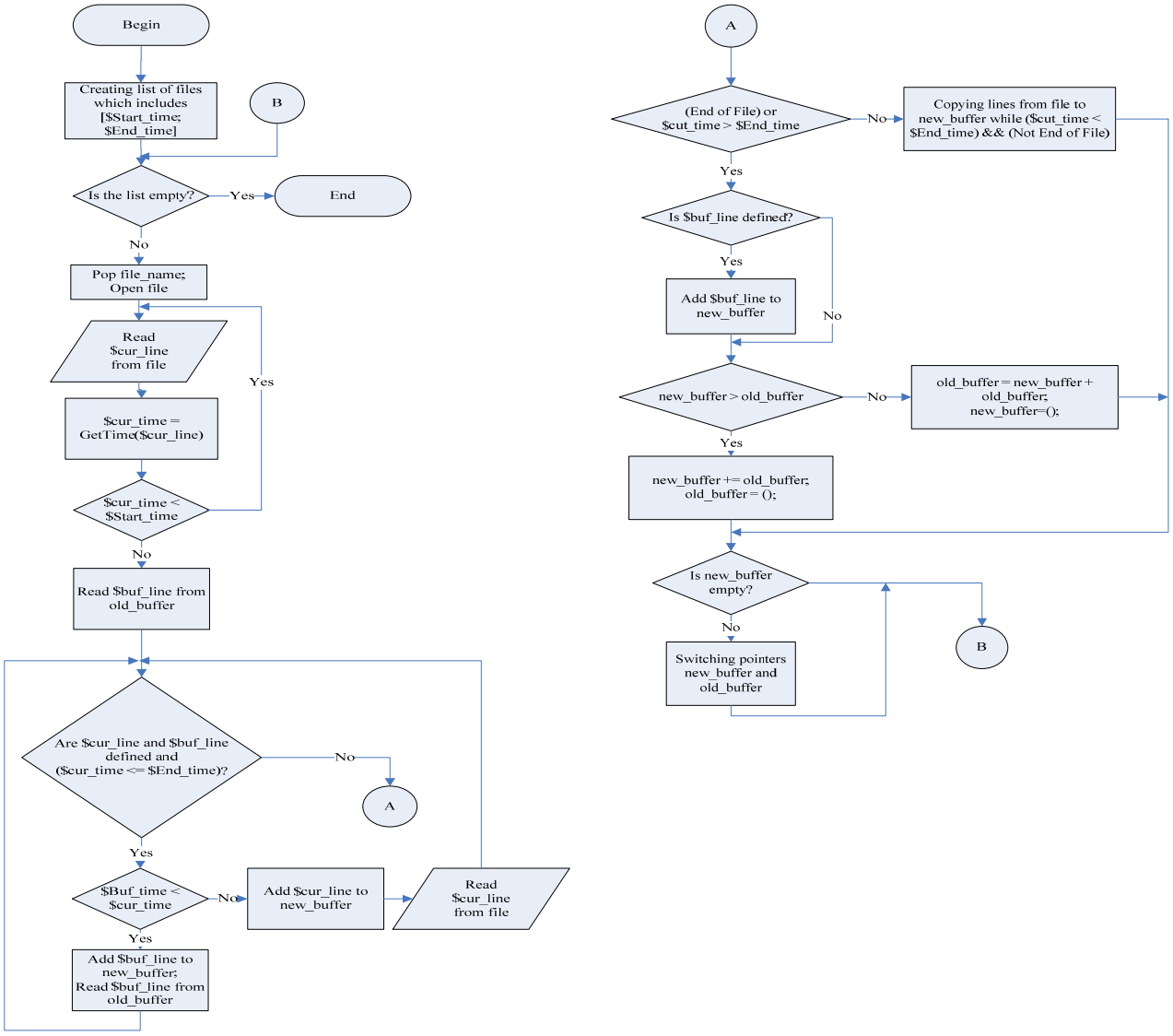


Figure 4: Timestamp sort

4) Work Remote module - Network Support

One of the most important features of the system is the network mode support which is provided by the Work Remote module.

Our application is working on the local machine connected to the target platform through several network gateways (Figure 5).

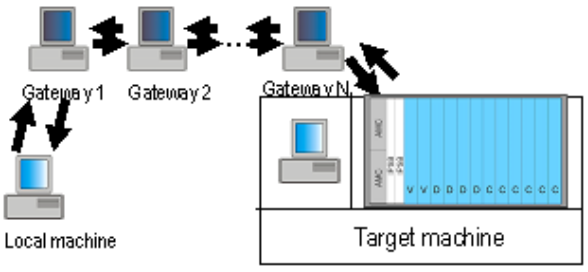


Figure 5: Work process in network mode

The module connects to the target machine and sends required scripts there. When it is necessary the application starts gathering of the required logs on the target machine and then starts merging process. After execution it removes script files and sends the result file to the local machine.

5) Introduction of analysis module as a further development

As a further development analysis module based on the knowledge and analytical skills of system experts should be added to the application to form an expert decision-making system together with the knowledge base generated by Log Merger. Rule engines (e.g. inference rule [6]) can be used for such purpose. Such system will automatically collect and investigate information from log files and then simply give the results freeing maintenance engineers of the necessity even to look through log files.

Conclusion

So we have developed a new type of system backed up by easy to perform common techniques which can be used on almost every platform both through the graphical user interface (GUI) and by the command line so that the user who does not possess libraries required for the GUI work can work with this system by calling scripts – the help file is given. Two run modes – local and network – enable user to gather required information from log files allocated either on the local machine or on remote machines without copying all log files to the local machine. As a result the structured information from different boards is collected in one file which could be defined as a knowledge base and can be used to develop an expert analyzing system.

Bibliography

[1] Charles Forgy, "A network match routine for production systems." Working Paper, 1974.

[2] Charles Forgy, "On the efficient implementation of production systems." Ph.D. Thesis, Carnegie-Mellon University, 1979.

[3] Charles Forgy, "Rete: A Fast Algorithm for the Many Pattern/Many Object Pattern Match Problem", *Artificial Intelligence*, 19, pp 17-37, 1982

[4] FlashStats - http://www.maximized.com/products/flashstats2006/

[5] LogInspector - http://www.semonitor.ru/log_analyzer.html

[6] Proceedings By Luc De Raedt And Peter Flach (Paperback - Oct 2 2001) «Machine Learning»: ECML 2001: 12th European Conference On Machine Learning, Freiburg, Germany, September 5-7, 2001.

Authors' Information

Igor Karelin *– Student of Saint Petersburg State Polytechnic University (fourth year), Software Engineer, Motorola Software Group – Russia, Saint-Petersburg Software center, T4 Business House, 12 Sedova str., 192019, St. Petersburg, Russia; e-mail: ngv743@motorola.com, ikarus47@mail.ru*

Tatyana Gavrilova - *Scientific adviser, Professor, Doctor of Science, Full professor in St. Petersburg State Technical University, department of Intelligent Computer Technologies, St. Petersburg, Russia; e-mail: tgavrilova@gmail.com*

Boris Lyubimov *– Graduate student of Saint Petersburg State Polytechnic University, Software Engineer, Motorola Software Group – Russia, Saint-Petersburg Software center, T4 Business House, 12 Sedova str., 192019, St. Petersburg, Russia; e-mail: abl100c@motorola.com*

ADAPTIVE SOA INFRASTRUCTURE BASED ON VARIABILITY MANAGEMENT

Peter Graubmann, Mikhail Roshchin

***Abstract**: In order to exploit the adaptability of a SOA infrastructure, it becomes necessary to provide platform mechanisms that support a mapping of the variability in the applications to the variability provided by the infrastructure. The approach focuses on the configuration of the needed infrastructure mechanisms including support for the derivation of the infrastructure variability model.*

***Keywords**: Service-oriented Architecture, Adaptability, Variability Model, Distributed Systems.*

***ACM Classification Keywords**: C.2.4 Distributed Systems*

***Conference**: The paper is selected from Sixth International Conference on Information Research and Applications – i.Tech 2008, Varna, Bulgaria, June-July 2008*

Introduction

Adaptability of IT infrastructures is one of the prerequisites that provides the necessary potential for allowing the realization of application variants (for instance in a software product line context). Usually, application configuration is a task that is driven by application feature selection. The corresponding selection of the appropriate infrastructure is a task derived thereof. It can be supported and automated to a great extent by exploiting dependencies and constraints between variation points and variations in variation models. This is obviously also the case in a SOA context where, as an additional requirement, configuration tasks are expected to be particularly flexible and supposed to be performed during all stages from design until run time.

The central idea of our approach is thus, to use a given configuration, that is, a particular resolution of the variability model of the given product line, and to derive the appropriate configuration of the infrastructure by exploiting dependencies, requirements and constraints described in the variation model.

Automated support of the derivation of the infrastructure configuration assists the application developers in selecting the best fitting infrastructure services and mechanism with appropriate quality of services and relieves them from the burden of investigating infrastructure properties again and again. It facilitates taking into account infrastructure usage patterns and best practices based on the knowledge of infrastructure experts (re-use of infrastructure knowledge). In the case of run time configuration, an automated support becomes inevitable.

Approach

Our approach particularly focuses on the derivation of the needed infrastructure configuration for a given application (that is, a given derivation in a product line). However, in order to provide support for this, an adequate description and a proper formalization of the available diversity of the infrastructure is a necessary prerequisite. Thus, we address the establishing of a variability model (VM) of the infrastructure and concentrate on the derivation of the infrastructure configuration from a given application configuration.

Identifying an appropriate infrastructure configuration is based upon the thorough understanding of the potential for adaptation of the available infrastructure services because they have to be mapped onto the requirements posed by the product variants derived from the given product line. To gain such an infrastructure variability model, we envisage three possibilities:

- The first step is to establish such an infrastructure variability model “manually”; that means that domain engineers knowledgeable of the infrastructure define it in the usual way by identifying the variation points in the infrastructure and the related available features (respective variants).

- The second step is to derive the infrastructure VM from the product line VM. This means to first identify the variation points of the product line that are related to infrastructure issues, then to collect the existing descriptions of the respective infrastructure services and mechanisms (for instance, descriptions of service models, bundles, etc.), and eventually to derive their constraints and dependencies. This results in an infrastructure VM that is precisely tailored to the infrastructure requirements formulated in the product line VM.
- The third step is a combination of the automated and a manual derivation of the infrastructure VM. Here, the challenge is to integrate the delta coming from the domain engineers into the infrastructure VM. A similar mechanism copes with evolutions in the product line and it's VM, adding the thereby emerging delta variability. This approach provides for the evolution of the infrastructure VM.

Having the infrastructure VM in place is prerequisite to identifying the relations between the infrastructure VM and the product line VM which in turn is needed to identify the appropriate infrastructure services/mechanisms from a specific application configuration.

Establishing the relations between the two VMs has to be done explicitly in the initial case where the infrastructure VM is defined manually. In the second case, these relations are established together with the construction of the VM. In the hybrid case, the manually defined VM delta has to be analyzed and the respective relations with the product line VM and its extensions have to be identified.

The derivation of an infrastructure configuration from a given application configuration is essentially based upon the two VMs, the product line and the infrastructure VM, and their relation as established by the previous subtask (in the following, we call this relation VM-relation). It also relies upon the behavior description (probably given as a business process model, etc.) of the given application.

In order to obtain an initial infrastructure configuration, the VM-relation has to be exploited, thereby identifying the specific infrastructure services/mechanisms required by the given application configuration. The infrastructure VM with its own dependency relations and constraints then provides the information for the eventual concrete configuration of the infrastructure.

Reacting to the need of a run time re-configuration of the application follows the same lines; however, the challenge here is to take into account that – since only a part of the original application configuration is changed – only as much modifications of the infrastructure are feasible as absolutely necessary (run time re-configurations have to be as less invasive as possible). For instance, through changing QoS requirements the re-configuration of only the infrastructure becomes necessary. Checking back with the original application configuration (for keeping the constraints and dependencies valid) will ensure the correctness of the revised infrastructure configuration.

Processing Adaptability

Requirements are variable in time by nature. Building SOA-based software product lines relies on the idea to achieve variability and therefore dynamic adaptability. The approach requires that all requirements have to be specified not only – as classically suggested – in a structured list of requirements, but also – as proposed in [Pohl, 2005] – in the form of a variability model. Consequently, the requirements layer has to get extended with an additional formal model, that means that, for instance, groups of requirements R1 and R2 have to be synchronized with Variant 1 and Variant 2, respectively (see Figure 1).

Typical problems, which regularly occur during the life-time of an application, comprise but are not limited to the following cases: data formats or data semantics varying from service to service, unexpected behavior not foreseen during design, missing required functionalities related to implementation errors, altered values of non-functional properties and changing QoS requirements. Adaptability would mean that parts of the software application should be changed in order to cover newly upcoming requirements or to avoid emergent issues. In the business process driven SOA-infrastructure this implies either a change in the part of the workflow, or a substitution of some of the subprocesses with different functionality. Obviously, utilizing the second possibility is

less costly. It can be realized by using already existing processes from other components/services of the related software product line keeping already existing functionality and extending it with required features.

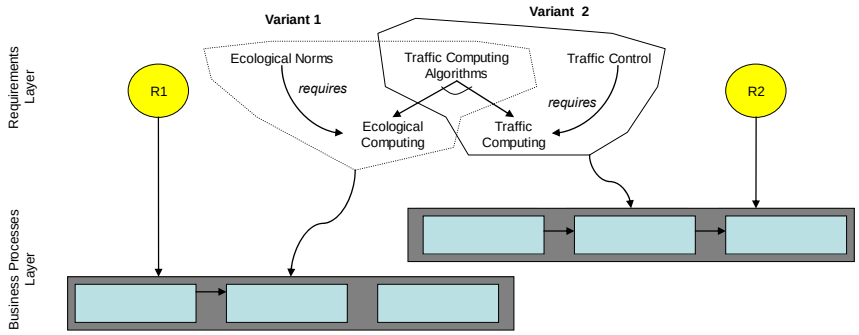


Figure 1. Extension of the Requirements Layer with VM

Managing business process specifications with a variability model can be done using rule-based mechanisms like the Object Constraint Language (OCL), Prolog or others.

A SOA-based infrastructure is built on top of existing services and their models which usually are available in repositories for easy access.

Replacing one subprocess by another is usually not sufficient. The whole dependency chain of related services has to be taken into account (so, a substitute, for instance, may require different protocols and access methods; it may show incompatibility with neighbor services and interface inconsistencies; or different semantics of the input values require further changes or imply the need to find similar services). This means that the whole related information within the infrastructure should be checked and taken into account. Dependency information is specified in form of dependency relations, extensions, and extension points (as defined in OSGi specification [OSGi, 2008]). This allows to extend existing variability models, and thus to provide for more reliable solutions. Such a task can be performed by using existing OSGi and SCA (see [SCA, 2008]) specifications for building dependency graphs, and further transforming that information into a variability model.

Use Case

Consider as an example a large traffic system built on top of a distributed SOA infrastructure. This means there are a lot of participants involved in an application such as permanent entities – traffic lights, cameras and sensors – and temporal entities like cars, bikers and pedestrians, all brought together to interact and to perform various services. The distribution of a SOA infrastructure allows defining particular algorithms for specific places (like townships, crossroads, or sections of a road), depending on the environment and its conditions. A typical task of such a large traffic system is to avoid traffic jams based on the information about the current traffic situation, the traffic density in neighboring roads, the concentration of pedestrians and the local traffic policies (probably including policies to enforce ecological regulations). Each entity can be represented by a service and provides necessary information, like speed of the cars, estimated fuel consumption, grade of air pollution, traffic light status, etc. Such an application is configured according to the particular requirement of the place it is located. If the situation changes it has to adapt itself – taking into account the environ of neighboring roads – by managing traffic lights and applying new strategies, for instance, changing driving directions on multiline highways (when this is possible).

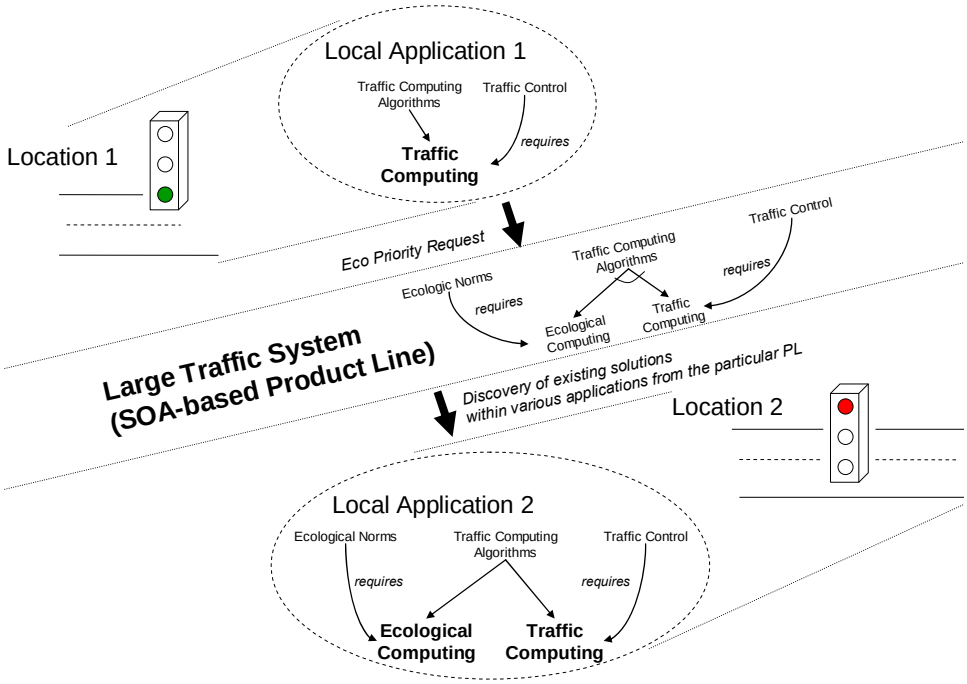


Figure 2. Large Traffic System

The whole large traffic system provides local applications, which are heterogeneous and designed for a particular local use. Thus, the appearance of an unexpected situation which is not covered by an application is not a surprise. Thereby, real time evolution for an application is a must. It can be achieved by using variable mechanism of the infrastructure.

Consider a situation, when a motorway is blocked by an incident, and cars have to take a bypass through a small town, where such a situation was not foreseen. The sudden appearance of a large number of cars is rather critical, thus, there is the necessity to take measures, for instance, to prevent air pollutions by applying strategies not planned beforehand.

The proposed approach assumes the existence of a variability model for the whole large traffic system and a monitoring system, which tracks changes in the behavior of local applications. If, for instance, a device recognizes increased air pollution, it gives a signal whereupon the traffic system changes its signaling patterns: because fuel consumption depends on the number of the cars' stops-and-goes, streets with heavy load will get a prolonged phase of "green light". This re-configuration of the local application relies on selecting appropriate features and tracing their dependencies within the variability tree (see Figure 3).

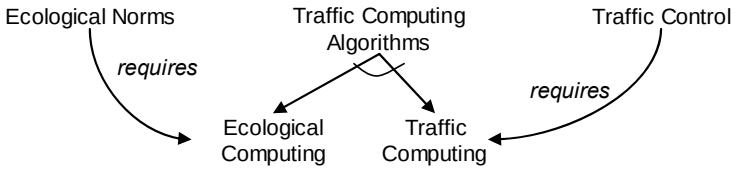


Figure 3. Part of the product line VM "Large Traffic System"

Initially, as it was mentioned above, the actual application uses only traffic-related policies without covering ecological constraints. To cope with the new situation, “Traffic Computing” and “Ecological computing” are combined to satisfy the new requirement “eco priority”.

The presented approach suggests an easy way of identifying an already existing business process from another local application that satisfies the new requirements. It could be part of already existing applications tailored to bigger cities, where considering ecology was already foreseen.

The next step is a combination of the initial process with a new one, where replacement of the activities implies the change in the whole set of required services and infrastructure settings, i.e., installation of the necessary protocols or extending a process with new services.

Conclusion

Adaptive mechanisms for SOA infrastructures are becoming more important with the increasing number of available SOA applications. In fact, without reliable and safe adaptive solutions, it becomes impossible to change existing and to build new services satisfying varying or versatile requirements. The proposed approach together with variability management guarantees that derived composite services with an appropriate infrastructure remain compliant to the varying situations.

Bibliography

[Pohl, 2005] K. Pohl, G. Boeckle, F. v. der Linden. Software Product Line Engineering, Springer, 2005.

[Kakola, 2006] T. Kakola, J.C. Duenas. Software Product Lines, 2006.

[OSGi, 2008] www.osgi.org

[SCA, 2008] www.osoa.org

[Abu-Matar, 2007] Mohammad Abu-Matar Toward a service-oriented analysis and design methodology for software product, 2007 http://www.ibm.com/developerworks/library/ar-soaspl/index.html

Authors' Information

Peter Graubmann *– Senior Engineer, CT SE, Siemens AG; e-mail: peter.graubmann@siemens.com*

Mikhail Roshchin *– PhD Student of Volgograd State Technical University, working in collaboration with CT SE, Siemens AG; e-mail: roshchin@gmail.com*

GRID APPROACH TO SATELLITE MONITORING SYSTEMS INTEGRATION

Nataliia Kussul, Andrii Shelestov, Serhiy Skakun

***Abstract**: This paper highlights the challenges of satellite monitoring systems integration, in particular based on Grid platform, and reviews possible solutions for these problems. We describe integration issues on different levels: data integration level and task management level (job submission in terms of Grid). We show example of described technologies for integration of monitoring systems of Ukraine (National Space Agency of Ukraine, NASU) and Russia (Space Research Institute RAS, IKI RAN). Another example refers to the development of InterGrid infrastructure that integrates several regional and national Grid systems: Ukrainian Academician Grid (with Satellite data processing Grid segment) and RSGS Grid (Chinese Academy of Sciences).*

***Keywords**: Grid systems, satellite monitoring systems.*

***ACM Classification Keywords**: H. Information Systems - H.1 Models and Principles*

***Conference**: The paper is selected from Sixth International Conference on Information Research and Applications – i.Tech 2008, Varna, Bulgaria, June-July 2008*

Introduction: Specifics of Earth Observation Problems

At present global climate changes in the world made rational land use, environmental monitoring and prediction of natural and technological disasters the tasks of a great importance. The basis for solving these problems is the use of data of different nature: modeling data, in situ measurements and indirect observations such as airborne and spaceborne remote sensing data. However, mutual disarrangement of heterogeneous data and measurement technologies, spatial and temporal inconsistency of measurements are limiting potentials of modern technologies for solving actual problems of environmental monitoring and forecasting of disasters. Thus, development of effective technologies for heterogeneous data integration is a very important issue.

Nowadays Earth Observation (EO) data play a major role in solving problems in different domains. Satellite observations enable acquisition of data for large and hard-to-reach territories, can provide continuous measurements and human-independent information, etc. EO domain, in turn, is characterized by large volumes of data that should be processed, catalogued, and archived. For example, GOME instrument onboard Envisat satellite generates nearly 400 Tb data per year [1]. EUMETCast system that is part of global GEONETCast system [2] of GEOSS enables acquisition of more than 50 Tb of processed and unprocessed information per year. Moreover, the processing of satellite data is carried out not by the single application with monolithic code, but by distributed applications. This process can be viewed as a complex workflow that is composed of many tasks: geometric and radiometric calibration, filtration, reprojection, composites construction, classification, products development, post-processing, visualization, etc. [3].

To enable processing and management of such volumes of data sets and information flows an appropriate infrastructure is needed that will support [1, 4]: access to distributed resources; high flexibility; portal enabling easy and homogeneous accessibility; collaborative work; seamless integration of resources and processes; allow processing of large historical archives; avoid unauthorised access to/use of resources.

Grid can provide appropriate facilities for high-performance computations and efficient data management in EO domain. Grid computing is an emerging paradigm for global computing and a very active research domain for complex, dynamic, distributed and flexible computing and resource sharing [5]. Grid computing belongs to main trends of on-line environment development among with web services, semantic web and peer-to-peer networking. The integration on these technologies is essential for the next generation networks.

Grid systems are recognized to be very efficient for EO and geospatial community for a number of reasons: geospatial data and associated computational resources are naturally distributed; the multi-discipline nature of

geospatial research and applications requires the integrated analysis of huge volume of multi-source data from multiple data centres; most geospatial modelling and applications are both data and computational intensive. The aggregated computational power of Grid system can provide for the application.

In this paper we highlight the challenges of satellite monitoring systems integration, in particular based on Grid platform, and review possible solutions for these problems.

State of the art: Grid-based systems for EO data processing

At present, Grid technologies are widely applied in different domains, in particular EO domain. EU-funded European DataGrid Project (EDG) was one of the first Grid-enabled projects allowing European Space Agency (ESA) to gain firsthand experience in the use of emerging Grid technologies [1]. Based on the gained experience European Space Agency (ESA) and European Space Research Institute (ESRIN) are developing Grid Processing on Demand (G-POD) for Earth Observation Applications (http://gpod.eo.esa.int). Online access to different data is enabled within this project, in particular to data provided by various instruments on Envisat satellite (http://envisat.esa.int), SEVIRI instrument onboard MSG (Meteosat Second Generation) satellite [6], ozone profiles derived from GOME instrument, etc. One of the most important applications is the analysis long-term data. Grid Web Portal provides access to the "Grid-on-demand" resources enabling: personal certification, time/space selection of data directly from the ESA catalogue, data transfer, job selection, launching and live status, data visualization.

DEGREE (Dissemination and Exploitation of GRids in Earth science) project (http://www.eu-degree.eu) is initiated within EGEE/EGEE-II. A major challenge for DEGREE is to build a bridge linking the Earth Science and GRID communities throughout Europe, and focusing in particular on the EGEE-II Project. Grid provides appropriate infrastructure enabling international cooperation within GMES and GEOSS. The following problems are within the scope of DEGREE: earthquake analysis, floods modeling and forecasting, influence of climate changes on agriculture

Japan Aerospace eXploration Agency (JAXA) and KEIO University started establishing "Digital Asia" system aimed at semi-real time data processing and analyzing. They use GRID environment to accumulate knowledge and know-how to process remote sensing data. The Digital Asia project is the part of bigger Sentinel Asia project that is targeting on building natural disasters monitoring system [7].

CEOS Wide Area Grid (WAG) project is initiated by CEOS Working Group on Information Systems and Services (WGISS), and aims at providing horizontal infrastructure enabling efficient integration of resources of different space agencies. WAG testbed infrastructure is currently under development within ESA Cat-1 project "Wide Area Grid Testbed for Flood Monitoring Using Spaceborne SAR and Optical Data" (no. 4181) [8]. Within WAG project Space Research Institute NASU-NSAU have developed testbed that integrates resources of Ukrainian Grid segment (Ukrainian Academician Grid) with resources of international organisations (ESA, RSGS-CAS).

Tendencies of globalization and integration of satellite monitoring systems

Nowadays there is a trend for globalization of monitoring systems with purpose of solving more complex problems and reducing collaboration expenses. EO data are naturally distributed over many organizations involved in data receiving and processing. This leads to the need of integration of existing systems for solution of complex problems. The development of GEOSS (Global Earth Observation System of Systems) [9] is coordinated by Group on Earth Observations (GEO) [10] that was launched in response to calls for action by the 2002 World Summit on Sustainable Development and the G8 (Group of Eight) leading industrialized countries. GEO is a voluntary partnership of governments and international organizations that provides a framework within which these partners can develop new projects and coordinate their strategies and investments. It is recognised that GEOSS work with and build upon existing national, regional, and international systems to provide comprehensive, coordinated Earth observations from thousands of instruments worldwide, transforming the data collected into vital information for society.

Modern tendencies of globalization and development of “system of systems” GEOSS lead to the need of integration of heterogeneous satellite monitoring systems. Integration can be done on different levels: (i) data exchange level, (ii) task management level. Data exchange level is supposed to provide tools for sharing data and products. This infrastructure enables data integration where different entities provide various kinds of data to support joint solution of complex problems (Fig. 1). Task management level envisages running applications on distributed computational resources provided by different entities (Fig. 2). Since many of the existing satellite monitoring system rely on Grid technologies appropriate approaches and technologies should be evaluated and developed to enable Grid system integration (so called InterGrid).

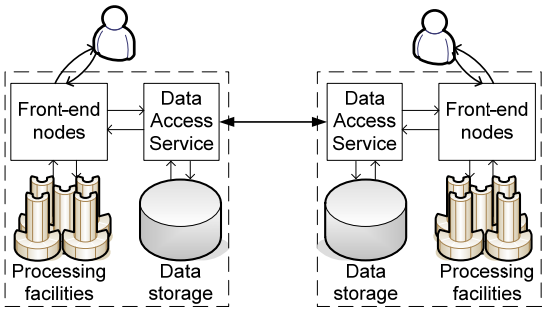


Fig 1. Data integration level

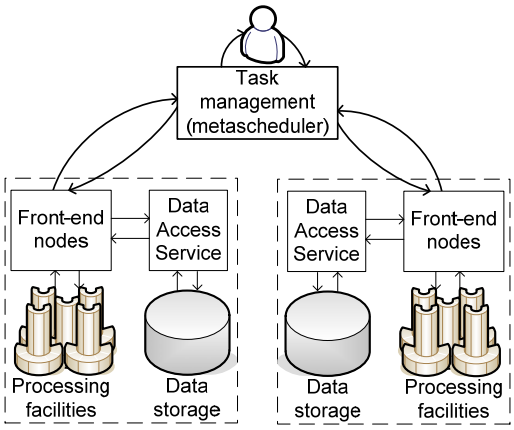


Fig 2. Task management level

The next sections will highlight main challenges and possible solutions for satellite monitoring systems integration on both levels, and provide description of case-studies for both cases.

Levels of integration: main problems and possible solutions

Integration on data exchange level could be done by using common standards of EO data exchange, common user interfaces, and common data and metadata catalog. As to task management level the following problems additionally should be solved: joint computational infrastructure setup; development of jobs submission and scheduling algorithms; load monitoring enabling; security policy enforcement.

Data exchange level. At present the most appropriate standards for data integration is OGC standards. Data visualization issues can be solved by using the following set of standards: WMS (Web Map Service), SLD (Style Layer Descriptors) and WMC (Web Map Context). OGC’s WFS (Web Feature Service) and WCS (Web Coverage Service) standards provide uniform ways for data delivery. In order to provide interoperability on the level of catalogues CSW (Catalogue for Web) standard can be applied.

Since data are stored on geographically distributed sites there can be issues regarding optimization of visualization schemes. In general, there are two possible ways for distributed data visualization: centralized visualization scheme and distributed visualization scheme. Advantages and faults of each scheme were described in [11].

Task management level. In this subsection we present main issues and possible solutions for Grid-system integration. Main prerequisite of such kind of integration is certificates trust. It could be done, for example, through EGEE infrastructure that nowadays brings together the resources of more than 70 countries. Another problems concerned with different Grid systems integration are as follows: enabling data transfers and high-level access to geospatial data; development of common catalogues; enabling jobs submission and monitoring; enabling information exchange.

Data transfer. GridFTP is an appropriate and reliable solution for data transfer. The only limitation is the requirement of transparent LAN (local area network) infrastructure.

Access to geospatial data. High-level access to geospatial data can be organised in two possible ways: using pure WSRF services or using OGSA-DAI container. Each of this approach has its own advantages and weaknesses. Basic functionality for WSRF-based services can be easily implemented (with proper tools), packed and deployed. But advanced functionality such as security delegation, third-party transfers, indexing should be implemented by hands. WSRF-based services can also pose some difficulties if we need to integrate them with other data-oriented software.

OGSA-DAI framework provides uniform interfaces to heterogeneous data. This framework makes possible to create high-level interfaces to data abstracting hiding details of data formats and representation schemas. Most of problems in OGSA-DAI are handled automatically, e.g. delegation, reliable transfer, data flow between different sources and sinks. OGSA-DAI containers are easily extendable and embeddable. But comparing to WSRF basic functionality implementation of OGSA-DAI extensions is more difficult. Moreover, OGSA-DAI requires preliminary deployment of additional software components.

Task management. There are two possible approaches for task management. One of them is to use Grid portal (Fig. 3) supporting different middleware platforms, such as GT4, gLite, etc. Grid portal is an integrated platform to end-users that enables access to Grid services and resources via standard Web browser. Grid portal solution is easy to deploy and maintain, but it doesn't provide application interface and scheduling capabilities.

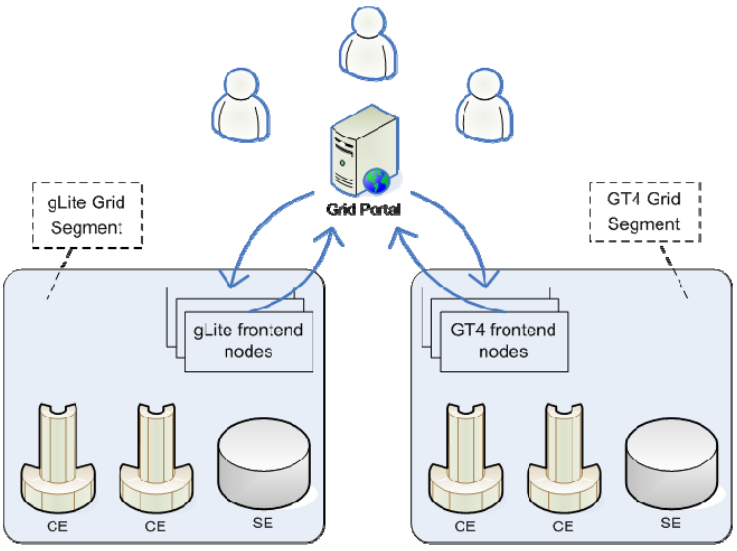


Fig. 3. Portal approach to Grid system integration

Another approach is to develop high-level Grid scheduler (Fig. 4) that will support different middleware by providing some standard interfaces. Such metascheduler interacts with low-level schedulers (used in different Grid systems) enabling in such way system interoperability. Metascheduler approach is much more difficult to maintain comparing to portals; however, it provides API with advanced scheduling and load-balancing capabilities. At present, the most comprehensive implementation for the metascheduler is a GridWay system. The GridWay metascheduler is compatibility with both Globus and gLite middlewares. Starting from Globus Toolkit v4.0.5 GridWay become standard part of its distribution. GridWay system provides comprehensive documentation for both users and developers that is a important point for implementing new features.

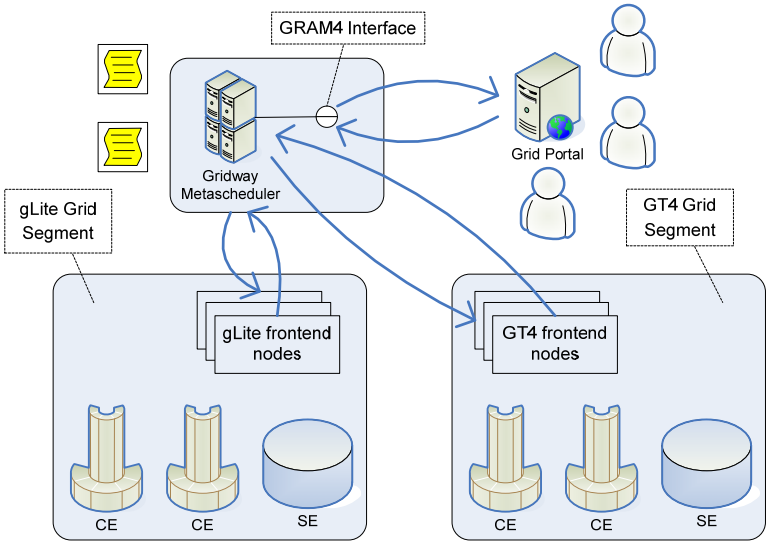


Fig. 4. Metascheduler approach

In the next section we show the examples of application of described approaches to integration of satellite monitoring systems and development of InterGrid environment.

Implementation: lessons learned

Integration of satellite monitoring systems. The first case-study refers to the integration of satellite monitoring systems of NSAU (Ukraine) and IKI RAN (Russia). The overall architecture for integration of data provided by two organizations is depicted in Fig. 5. The proposed approach is applied for the solution of problems for agriculture resources monitoring and crop yield prediction. Within integration NSAU provides WMS interfaces to NWP modelling data (using WRF model) [12], in-situ observations from meteorological ground stations in Ukraine, and land parameters (such as temperature, vegetation indices, soil moisture) derived from satellite observations from MODIS instrument onboard Terra satellite. IKI RAN provides WMS interfaces to operational land and disaster monitoring system. Both NSAU and IKI RAN provides user Web-interfaces to monitoring systems that support OGC WMS standards.

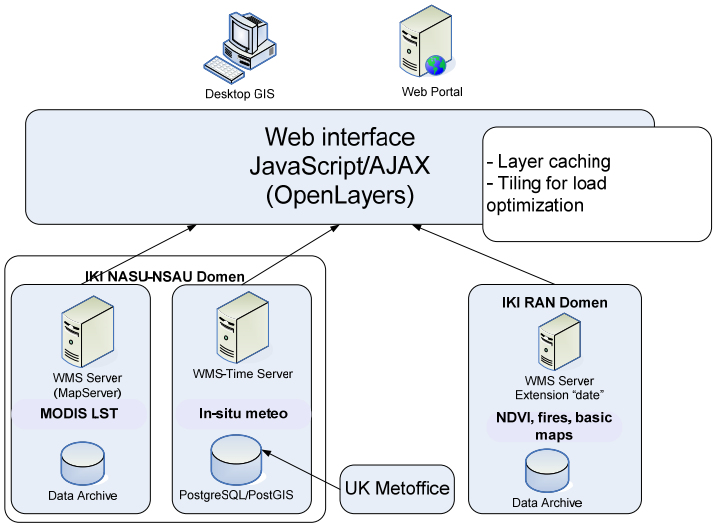


Fig. 5. Architecture of satellite monitoring system integration

In order to provide user interface that will enable visualization of data from multiple sources we use open-source OpenLayers framework (http://www.openlayers.org). OpenLayers is “thick client” software based on JavaScript/AJAX and fully operational on client side. Main OpenLayers features also include: support for several WMS servers, support for different OGC standards (WMS, WFS), cache and tiling support to optimize visualization, support for of both raster and vector data. The provided data and products are accessible via Internet http://land.ikd.kiev.ua. The example of OpenLayers visualization of data from multiple sources is depicted in Fig. 6.

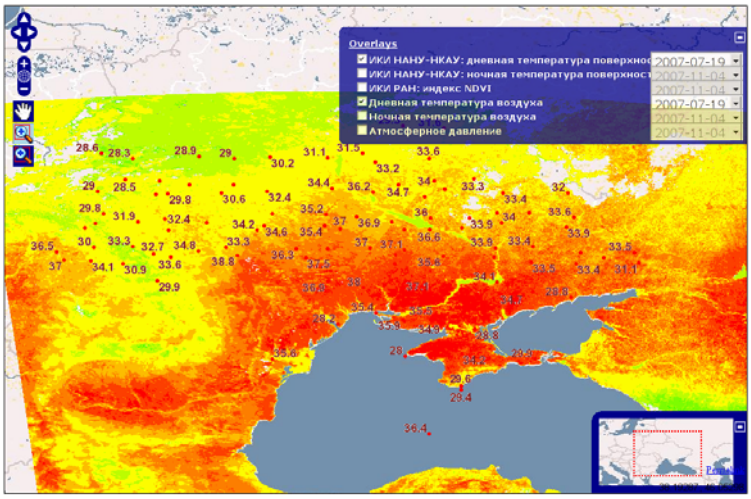


Fig. 6. OpenLayers interface to multiple data

InterGrid testbed development. The second case-study refers to the development of InterGrid for environmental and natural disaster monitoring. InterGrid integrates Ukrainian Academician Grid (with Satellite data processing Grid segment) and RSGS Grid (Chinese Academy of Sciences) and is considered as a testbed for Wide Area Grid (WAG) implementation—a project initiated within CEOS Working Group on Information Systems and Services (WGISS).

The important application that is being solved within InterGrid environment is flood monitoring and prediction. This task requires adaptation and tuning of existing hydrological and hydraulic models for corresponding territories and the use of heterogeneous data stored on multiple sites. Flood monitoring and prediction requires the use of the following data sets: NWP modelling data (provided by Satellite data processing Grid segment), SAR imagery from Envisat/ASAR and ERS-2/SAR satellites (provided by ESA), products derived from optical and microwave satellite data such as soil moisture, precipitation, flood extent etc., in-situ observations from meteorological ground stations and digital elevation model (DEM). The process of model adaptation can be viewed as a complex workflow and requires the solution of optimization problems (so called parametric study). Satellite data processing and products generation tasks also represent complex workflow and require intensive computations. All these factors lead to the need of using computational and informational resources of different organizations and their resources into joint InterGrid infrastructure. The architecture of proposed InterGrid is depicted in Fig. 7.

GridFTP was chosen to provide data transfer between Grid systems. In order to enable interoperability between different middleware (for example, Satellite data processing Grid segment is using GT4; RSGS Grid is using gLite 3.x; Ukrainian Academician Grid is based on NorduGrid) we developed Grid portal that is based on GridSphere portal framework (http:// www.gridsphere.org). The developed Grid portal allows users to transfer data between different nodes and submit jobs on computational resources of the InterGrid environment. The portal also provides facilities to monitor statistics of the resources such as CPU load, memory usage, etc. The

further works on providing interoperability between different middlewares are directed to the development of metascheduler using GridWay system. In the nearest future we are intended to provide integration with ESA's EO Grid-on-Demand infrastructure.

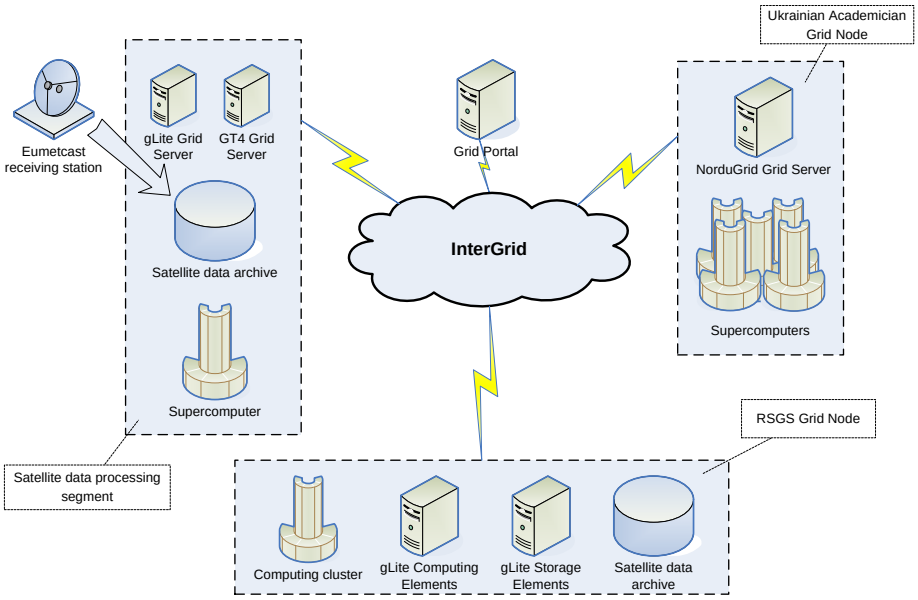


Fig. 7. InterGrid architecture

Conclusions

This paper focuses on the problems of integration of satellite monitoring systems, in particular those using Grid platform. We described two possible levels of integration (data level and task management level) reviewing possible solution for implementing each of them. Considering data integration level we found that integration could be provided by using existing standards for geospatial data, in particular OGC standards. We demonstrated applicability and usability of this approach for integrating existing satellite monitoring systems of Ukraine and Russia for agriculture applications. The use of standard OGC interfaces makes it possible to standardise and facilitate development of integrated satellite monitoring systems (based on existing ones) to exploit synergy and acquire information of new quality.

As to integration on task management level we reviewed two solutions: portal-based and metascheduling approach. We implemented portal solution based on GridSphere framework for the InterGrid environment that integrates several regional and national Grid systems. In order to provide advanced scheduling and load-balancing capabilities the further works will be directed to the implementation of metascheduler based on GridWay system. Also we are intended to provide integration with ESA's G-POD. Further investigations will be directed to integration of distributed monitoring systems with SensorWeb networks to provide automatic delivery of data from heterogeneous sources.

Acknowledgment

The work is supported by ESA CAT-1 project "Wide Area Grid Testbed for Flood Monitoring using Spaceborne SAR and Optical Data" (#4181) and by INTAS-CNES-NSAU project "Data Fusion Grid Infrastructure" (Ref. Nr 06-1000024-9154).

Bibliography

1. Fusco L., Goncalves P., Linford J., Fulcoli M., Terracina A., D'Acunzo G. Putting Earth-Observation on the Grid. In: ESA Bulletin, 2003, 114, pp. 86-91.
2. GEONETCast, http://www.earthobservations.org/progress/GEONETCast.html.
3. Rees W.G. Physical Principles of Remote Sensing, Cambridge University Press, 2001.
4. Shelestov Andrey Yu., Kussul Nataliya N., Skakun Sergey V. Grid Technologies in Monitoring Systems Based on Satellite Data. J. of Automation and Information Science, 2006, vol. 38, issue 3, pp. 69-80.
5. Foster I., Kesselman C. The Grid: Blueprint for a New Computing Infrastructure. 2nd Edition, Morgan Kaufmann, 2004.
6. EUMETSAT, http://www.eumetsat.int.
7. Fukui H. Sentinel Asia/Digital Asia: Building Information Sharing Platform by Geo web services and contributing to Disaster Management Support in the Asia-Pacific Region.
8. Kopp P., Petiteville I., Shelestov A., Li G. Wide Area Grid (WAG). In: Proc. The 7th Ukrainian Conference on Space Research, National Flight and Control Center, Evpatoria, Ukraine, 2007, p. 209.
9. GEOSS Portal, http://www.geossportal.com.
10. Group on Earth Observations (GEO), http://www.earthobservations.org.
11. Shelestov A., Kravchenko O., Ilin M. Distributed visualization systems in remote sensing data processing GRID, International Journal "Information Technologies and Knowledge" - volume 2 / 2008. (in print)
12. Kussul N., Shelestov A., Skakun S., Kravchenko O. Data Assimilation Technique For Flood Monitoring and Prediction. International Journal on Information Theory and Applications, 2008, volume 15. (in print).

Authors' Information

Nataliia Kussul *– Prof., Dr., Head of Department of Space Information Technologies and Systems, Space Research Institute of NASU-NSAU, Glushkov Ave 40, build. 4/1, Kyiv-187, 03680 Ukraine,*
e-mail: inform@ikd.kiev.ua

Andrii Yu. Shelestov *– PhD, Senior Researcher, Department of Space Information Technologies and Systems, Space Research Institute of NASU-NSAU, Glushkov Ave 40, build. 4/1, Kyiv-187, 03680 Ukraine,*
e-mail: inform@ikd.kiev.ua

Serhiy V. Skakun *– PhD, Research Assistant, Department of Space Information Technologies and Systems, Space Research Institute of NASU-NSAU, Glushkov Ave 40, build. 4/1, Kyiv-187, 03680 Ukraine,*
e-mail: serhiy.skakun@ikd.kiev.ua

DATA PROTECTION AND PACKET MODE IN THE DISTRIBUTED INFORMATION MEASUREMENT AND CONTROL SYSTEM FOR RESEARCH IN PHYSICS

Sergey Kiprushkin, Nikolay Korolev, Sergey Kurskov, Vadim Semin

***Abstract**: The present paper is devoted to creation of cryptographic data security and realization of the packet mode in the distributed information measurement and control system that implements methods of optical spectroscopy for plasma physics research and atomic collisions. This system gives a remote access to information and instrument resources within the Intranet/Internet networks. The system provides remote access to information and hardware resources for the natural sciences within the Intranet/Internet networks. The access to physical equipment is realized through the standard interface servers (PXI, CAMAC, and GPIB), the server providing access to Ethernet devices, and the communication server, which integrates the equipment servers into a uniform information system. The system is used to make research task in optical spectroscopy, as well as to support the process of education at the Department of Physics and Engineering of Petrozavodsk State University.*

***Keywords**: distributed information measurement and control system, equipment server, PXI server, CAMAC server, GPIB server, distance learning.*

***ACM Classification Keywords**: H.3.4 Systems and Software: Distributed systems.*

***Conference**: The paper is selected from Sixth International Conference on Information Research and Applications – i.Tech 2008, Varna, Bulgaria, June-July 2008*

Introduction

A distributed information measurement and control system was implemented at the Department of Physics and Engineering of Petrozavodsk State University (Russia) to fortify research in the field of optical spectroscopy and facilitate academic activities [Gavrilov et al, 2003], [Kiprushkin et al, 2004 – 2005].

The system is quit unique because it integrates various tool interfaces into one network functioning on the basis of the TCP/IP protocol stack.

The client-server technology was chosen as the key element of the system; in addition, an application protocol [Гаврилов et al, 2002] over the TCP/IP was developed to access physical experimental equipment – which enables the system to function in the Intranet/Internet networks. It was necessary to create our own protocol because common Web-technologies lack flexibility in experiment monitoring since, in this case, experimental procedures are run by executable codes, saved on the computer directly connected to the experimental setup, rather than client programs [e.g. Зимин et al, 2006]

The heterogeneous system includes client programs, that run the experiment, a communication server, the key element of the system, equipment servers (CAMAC server [Zhiganov et al, 2000], GPIB server [Кашуба et al, 2002], PXI Server, Intel MCS-196 microcontroller server, the Ethernet devices server [Kiprushkin, Kurskov, Sukharev, 2007], the server of access to GDS-840C digital oscilloscope etc.), measuring and execution units of the experimental setup, and a database server [Kiprushkin, Kurskov, Semin, 2007]. The scheme of the distributed information measurement and control system is presented in Figure 1.

Output protocoling and database storing are realized on the basis of DBMS Oracle 9i. Client programs call it directly, bypassing the client server, because the latter has no information on the type of the ongoing experiment.

The communication server, the equipment servers, and client programs are realized as Java applications. The data exchange among them is based on TCP stream sockets provided by Java.net information packet, which is

included into Java API standard packet. The methods of using the input-output ports for the access to the interface controllers are written in C programming language.

Administration of the distributed system is based on server-side Java servlet.

The goal of this project was to switch the information measurement and control system to the SSL protocol (Secure Socket Layer) and to upgrade it with the packet mode.

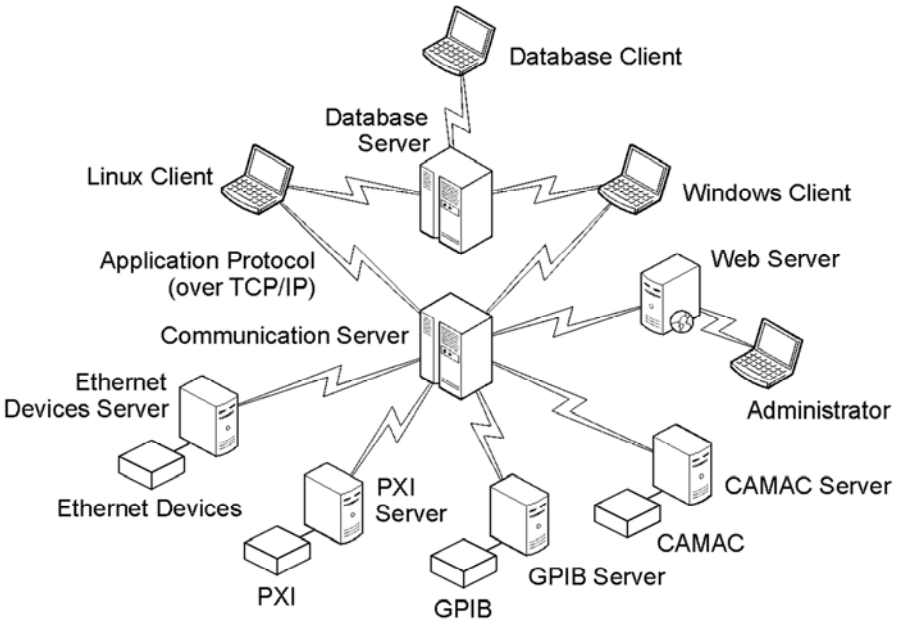


Figure 1. The scheme of the distributed information measurement and control system

Data Protection

The SSL protocol is a standard protocol based on the TCP/IP protocol stack with an encrypted connection between the client and the server. The SSL protocol follows invariable communication security steps – which are an advantage and a disadvantage – on the one hand, it operates time-proven methods, on the other hand, there is often a lack of flexibility when it comes to develop a unique system that requires more advanced methods. It is crucial to note that the majority of modern software products support the SSL protocol – that reflects a growing importance of security in information technologies.

The SSL protocol is a connection protocol developed by Netscape Communications Corporation. It runs over the TCP/IP. The SSL protocol provides confidentiality via data encryption, protection of data integrity with Message Authentication Code (MAC), client and server authentication and validity. The SSL protocol may be used by higher-level protocols such as, for example, the HTTP. The SSL protocol Version 1 did not become popular while Version 2 was introduced by the Netscape Company in the first version of Netscape Navigator. The third version of this protocol is the most modern and widely-spread. The TLS protocol (Transport Layer Security) developed by the Internet Engineering Task Force (IETF), broadens capabilities of the SSL protocol Version 3 regarding authentication. The WTLS protocol (Wireless Transport Layer Security) is a version of the TLS protocol for wireless networks.

The developed system of cryptographic security of the information measurement and control system is double-leveled in terms of security policies. The communication server works as a core element that distributes secret keys and authenticates all public keys. It is an issuer, i.e. it issues certificates to public keys. It is completely trusted. A certificate is a means of client and server authentication when establishing the connection. A certificate is a block of data that contains information required for principal's identification. This information contains

principal's public key, information on the principal, period of validity of the certificate, issuer's information signature.

Java 2 has a keytool utility that allows monitoring of keystores. The keystores contain generated keys and trusted certificates. Key generation is a private key plus a sequence of X.509 certificates that authenticates a corresponding public key. The keytool utility does not support symmetric keys; instead, during an SSL connection a secret session key (temporary keys) is generated for traffic encryption.

Data security in the system is based on Java Secure Socket Extension (JSSE), that is, basically, a standard API-interface for the SSL protocol versions 2 and 3 and the TLS protocol (class library – file jsse.jar). Cryptographic features of the SSL protocol are hidden from the programmer.

Client and server applications that use the SSL protocol based on SSL-sockets are generated in a similar way: the route to the keystore location, its type and password as well as a certificate authority and trusted certificate authorities are indicated in the program. The utility keytool facilitates generation and use of key stores. It enables a user to set the following parameters of a key: an alias, an encryption algorithm, a size, validity, and a keystore location. Option defaults are as follows: -alias "mykey", -keyalg "DSA", -keysize 1024, -validity 90, -keystore the file named keystore in the user's home directory. If the main algorithm is of type "DSA", the signature algorithm option defaults to "SHA1withDSA". If the underlying main algorithm is of type "RSA", -the signature algorithm defaults to "MD5withRSA".

To establish an SSL-connection, the server and the client first have to export public keys' certificates to the file, then exchange and import them to their keystores.

The use of SSL-sockets reduces the costs of encrypted communications and broadens capabilities of the system. Whereas, one of the disadvantages is that SSL-sockets do not support all cryptographic algorithms.

The former cryptographic classes [Kiprushkin, Korolev, Kurskov, 2005] developed for this system are still of importance because they can be used to secure cryptographically almost any information measurement system.

Packet Mode of Distributed System

As for the second part of this project, it is necessary to point out that client-server architecture enables users (clients) to get a secure access to information, independent from hardware and software combinations. The client-server model fortifies the use of new automation technologies, brings data processing closer to the client, simplifies the use of graphic interfaces and transition to open systems.

However, an experimental complex can be run directly by the commands of a client program only if the connection is secure and the network capacity is sufficient. Otherwise, there is a risk that the experiment will be delayed or results lost.

The packet mode enables the program to send the client command packets instead of single commands.

The packet mode makes it possible to reduce data volume transmitted over the net as well as shorten the timing of useful data transmission. This is crucial when organizing distance lab activities for students studying off campus.

To realize the packet mode, we developed modules (classes) that allow transmission and processing of blocks of commands. In addition, method libraries of equipment servers connected to physical setups were upgraded with methods that do not require client's interference in experimental measurements (for example, a procedure of setting up the monochromator to the chosen wave length or measuring the impulse counting rate from the photomultiplier tube).

Let us take a closer look at how the system functions in a standard mode and a packet mode (without considering security procedures).

In a standard mode, a client asks a server to reserve a resource. Then, a communication server verifies that an equipment server has this resource and that is not being used by another client. Depending on the output, the server responds to the client that the resource is available, being used by another user reserved by this user or

does not exist. If the resource is available, the client sends the first command to the communication server, the latter checks and forwards it to the equipment server. The equipment server, having processed the command, sends the output to the communication server that sends it then back to the client. After that, the client may send the next command. Having done working with the resource, the client sends the communication server a command that the resource is available.

The system works differently in a packet mode. In the beginning, just like in a standard mode, a resource is reserved. After that, the client communicates to the server that packet transmission is initiated. Following this command, the communication server starts to send the equipment server the commands received from the client without waiting for them to be executed. The equipment server, having received a packet transmission begin command, stores all consecutive commands in a file until the packet transmission is over. After that, the equipment server starts to execute the commands reading them from the file. At this time, the client may disconnect from the communication server since there is no need to remain connected. When the server has finished executing the commands, the client can ask it to send an output packet.

Command or output packet transfer is run by single frames with a server or a client confirming the reception of each frame.

During the development of the packet mode, the interface of the communication server CServerProtocol was upgraded with the following commands:

- CS_GETHSERVERSTATUS – check the status of the addressed equipment server;
- CS_RECEIVEPACKET – a command that switches the communication server to the receive packet mode (begin packet) ;
- CS_ENDPACKET – a command that ends packet transmission (end packet);
- CS_SENDPACKET – send an output packet.

Likewise, the commands of equipment servers were upgraded with the commands that do not require any response from a running client program. For the CAMAC server they are as follows:

- CMS_WRITE_WITH_L – write data and wait for L-request;
- CMS_WAIT_L – wait for L-request;
- CMS_CHECK_Q – check the status of signal Q;
- CMS_BEGIN_LINE – set up the monochromator to the chosen wave length;
- CMS_SPECTRUM – register the spectrum in a set wave length diapason;
- CMS_FUNCTION – write the excitation function of a spectral line in a set diapason of energy collisions;
- et al.

Commands stop running and communicate an error if there is no L-request within a set time period.

Conclusion

This distributed information measurement and control system is based on the modular approach implemented both in the structure and in the software. Clients and equipment servers are built into the system according to the unified rules and interact on a unified protocol by the principles of open systems. Note that an open system is a system that implements open specifications or standards for interfaces, services and formats in order to provide software portability with minimal changes in a wide range of systems (mobility) as well as interaction with other applications on local or remote systems (interoperability) and users (user mobility). In particular, distributed systems are based on OSE/RM model that describes systems by client/server architecture.

The use of the SSL protocol working over the TCP/IP protocols in data encryption significantly simplified traffic encryption between a client and a server ensuring information integrity and confidentiality. The SSL protocol is

particularly worth-using in distributed information measurement and control systems that are normally utilized in research and academic labs with less strict data and equipment security requirements.

As for the developed packet mode, it considerably increases system security in general by preventing experimental data loss due to lost connection with the client and reduces network load.

It is necessary to point out that the developed distributed information measurement and control system is used for the beam and plasma object analysis with the help of optical spectroscopy methods [Kurskov et al, 2006], [Кашуба, 2006]. In particular, the researches on excitation processes of atomic collisions with inert gas atoms' participation are carried out with its help as well as the laboratory works with senior students of the Department of Physics and Engineering of Petrozavodsk State University.

Acknowledgments

We would like to express our gratitude to the laboratories' Head I. P. Shibaev for support of this work as well as engineers A. N. Cykunov and A. V. Mandychev, and Masters of Philosophy M. A. Gvozd, V. G. Mullamekhametov, and D. V. Korolev.

The research described in this publication was made possible in part by Award of the U.S. Civilian Research & Development Foundation (CRDF) and of the Ministry of Education and Science of Russian Federation.

Conclusion

This exemplar is meant to be a model for manuscript format. Please make your manuscript look as much like this exemplar as possible.

In the case of serious deviations from the format, the paper will be returned for reformatting.

Bibliography

[Gavrilov et al, 2003] S.E.Gavrilov, S.A.Kiprushkin, S.Yu.Kurskov. Distributed information system with remote access to physical equipment. In: Proceedings of the International Conference on Computer, Communication and Control Technologies: CCCT '03 and The 9th International Conference on Information Systems Analysis and Synthesis: ISAS'03. Orlando, 2003.

[Kiprushkin et al, 2004] S.A.Kiprushkin, N.A.Korolev, S.Yu.Kurskov. Data security in the distributed information measurement system. In: Proceedings of the 8th World Multi-Conference on Systemics, Cybernetics and Informatics: SCI 2004. Orlando, 2004, Vol. 1, pp. 13-16.

[Kiprushkin et al, 2005] S.A.Kiprushkin, S.Yu.Kurskov, N.G.Nosovich. Resources Control in Distributed Information Measurement System. In: Proceedings of the 3rd International Conference on Computing, Communication and Control Technologies: CCCT '05. Austin, Texas, USA, 2005.

[Kiprushkin et al, 2005] S.A.Kiprushkin, N.A.Korolev, S.Yu.Kurskov. Sharing of Instrument Resources on the Basis of Distributed Information Measurement System. In: Proceedings of the Second IASTED International Multi-Conference on Automation, Control, and Information Technology – Automation, Control, and Applications: ACIT-ACA 2005. Novosibirsk, ACTA Press, 2005, pp. 170-175.

[Гаврилов et al, 2003] С.Е.Гаврилов, Е.Д.Жиганов, С.А.Кипрушкин, С.Ю.Курсков. Распределенная информационно-измерительная система для удаленного управления экспериментом в области оптической спектроскопии". Труды Всероссийской научной конференции Научный сервис в сети Интернет 2002. Москва, Издательство Московского государственного университета, 2002, сс. 157–159.

[Зимин et al, 2006] А.М.Зимин, Б.В.Букеткин, А.П.Почуев и др. Учебная Интернет-лаборатория "Испытания материалов". Информационные технологии, No. 10, 2006, сс. 58–65.

[Zhiganov et al, 2000] E.D.Zhiganov, C.A.Kiprushkin, S.Yu.Kurskov. CAMAC Server for Remote Access to Physical Equipment. In: Learning and Teaching Science and Mathematics in Secondary and Higher Education. Joensuu, University of Joensuu, 2000, pp. 170–173.

[Кашуба et al, 2002] А.С.Кашуба, С.А.Кипрушкин, С.Ю.Курсков. Сервер канала общего пользования распределенной информационной системы поддержки научных исследований в области оптической спектроскопии. В: Технологии

информационного общества – Интернет и современное общество 2002 (IST/IMS 2002): Материалы V Всерос. объединенной конф. Санкт-Петербург, Издательство С.-Петербургского университета, 2002, сс. 104–105.

[Kiprushkin, Kurskov, Sukharev, 2007] S.Kiprushkin, S.Kurskov, E.Sukharev. Connection of network sensors to distributed information measurement and control system for education and research. International Journal "Information Technologies & Knowledge", Vol. 1, No. 2, 2007, pp. 171–175.

[Kiprushkin, Kurskov, Semin, 2007] S.Kiprushkin, S.Kurskov, V.Semin. Development of database for distributed information measurement and control system. In: Proceedings of the International Conference "e-Management & Business Intelligence": eM&BI 2007, Bulgaria, Eds.: Kr.Markov, Kr.Ivanova. Sofia, Institute of Information Theories and Applications FOI ITHEA, 2007, pp. 48–51.

[Kurskov, 2006] S.Yu.Kurskov, A.D.Khakhaev. On mechanisms of He I collisional excitation in He-He system. Czechoslovak Journal of Physics, Vol. 56, 2006, pp. B297–B302.

[Кашуба, 2006] А.С.Кашуба. Проблемно-ориентированная распределенная информационно-измерительная и управляющая система для изучения процессов возбуждения при столкновениях тяжелых частиц. Системы управления и информационные технологии, No. 4.2 (26), 2006, сс. 234–238.

Authors' Information

Sergey Kiprushkin *– Senior lecturer, Petrozavodsk State University, Department of Physics and Engineering, Lenin Ave., 33, Petrozavodsk-185910, Russia; e-mail: skipr@dfe3300.karelia.ru*

Sergey Kurskov *– Associate professor, Petrozavodsk State University, Department of Physics and Engineering, Lenin Ave., 33, Petrozavodsk-185910, Russia; e-mail: kurskov@psu.karelia.ru*

Nikolay Korolev *– Researcher, Petrozavodsk State University, Department of Physics and Engineering, Lenin Ave., 33, Petrozavodsk-185910, Russia; e-mail: kna@sampo.ru*

Vadim Semin *– PhD student, Petrozavodsk State University, Department of Physics and Engineering, Lenin Ave., 33, Petrozavodsk-185910, Russia; e-mail:semin@psu.karelia.ru*

THE COMPLEX UNIFIED EVOLUTIONARY APPROACH TO THE CREATION OF THE MULTILEVEL DISTRIBUTED CONTROL SYSTEM OF A GAS-TRANSPORT COMPANY

Victor Borisenko, Bogdan Kluk, Jury Ponomarev, Anton Starovoytov

***Abstract**: the objects of a large-scale gas-transport company (GTC) suggest a complex unified evolutionary approach, which covers basic building concepts, up-to-date technologies, models, methods and means that are used in the phases of design, adoption, maintenance and development of the multilevel automated distributed control systems (ADCS).. As a single methodological basis of the suggested approach three basic Concepts, which contain the basic methodological principles and conceptual provisions on the creation of distributed control systems, were worked out: systems of the lower level (ACS of the technological processes based on up-to-date SCADA), of the middle level (ACS of the operative-dispatch production control based on MES-systems) and of the high level (business process control on the basis of complex automated systems ERP).*

***Keywords**: gas-transport company, distributed control system, concept of the creation of the complex distributed control system, unified integration platform, adaptive expandable data-pump, modified FDD-design-technology, evolutionary strategy, corporate maintenance methodology.*

***ACM Classification Keywords**: H.4.2 Types of Systems*

***Conference**: The paper is selected from Sixth International Conference on Information Research and Applications – i.Tech 2008, Varna, Bulgaria, June-July 2008*

Introduction

The main characteristics of the up-to-date gas-transport enterprises as control objects are:

- ramified, multilevel, geographically distributed (more than 37 000 km) logistical structure;
- continuous, technically difficult and explosive technology of gas transportation through high-pressure pipe main lines;
- essential influence on the country's economy (considerable percent of the government budget input of the country).

That's why considerable reliability, increase of effectiveness and safety of the gas-transport system based on the creation of a complex multilevel distributed control system is a very important and an actual task.

The principal logistical, technological and economic characteristics of the up-to-date nets of the gas mains and gas-transport enterprises it is necessary to take into account as control objects

1.The creation concept of distributed ACS TP based on modern SCADA

In 2005-2006 the research and development institute of the gas-transport ACS worked out the“Creation concept of the automated distributed control systems of technological processes (ADCS TP)” for the enterprises of the national stock company (NSC) “Neftegaz Ukraine” (further the ADCS TP Concept).

This first basic conceptual document was adopted as a normative-methodological branch standard. It contains the basic methodological principles, functional requirements and conceptual provisions on the creation of distributed control systems: of the lower level (ACS of the technological processes based on up-to-date SCADA).

Nowadays the ADCS TP Concept is rather widely used by all design organisations and developer companies for unification, standardization, quality improving and increase of effectiveness of designing new means and modernization of present ones, complexes and automatization systems of technological objects and also for

solving one of the most important problem of system-wide integration of all interstitial local systems into one complex distributed control system based on using where possible unified type decisions, which passed preliminary testing.

2. The creation concept of complex ACS of business processes based on ERP system

In 2006-2007 the institute worked out the “Creation concept of complex automated control systems (CACS) of the basic business processes (BP) of the national stock company (NSC) “Naftogaz Ukraine” (further the CACS BP Concept).

This second basic conceptual document was also adopted as a normative-methodological branch standard that is used by the creation of systems of the upper level of hierarchy of managing the company. CACS BP Concept contains complex hierarchial analysis of all basic occupations and business processes of enterprises, which affiliate the company, basic principles, system-wide and specialized requires, and also the main conceptual provisions on the creation of CACS BP as a complex distributed system: the upper level of managing the company.

In the CACS BP Concept the method of building of its basic part (data-pump) based on buying and implementing of a complex highly parametrized “brand name” from the system Enterprise Resource Planning (ERP) was chosen substantially. Nowadays as basic ERP-system was substantially chosen SAP Enterprise Business Suite Companies SAP AG.

3. The creation concept of ACS of the operative-dispatch control based on MES-systems

In 2007 the institute worked out the “Creation concept of automated operative-dispatch control system for the subsidiary company (SC) “Ukrtransgaz” on the basis of MES-sytem (further the MES Concept).

This third basic conceptual document is ment for the creation of systems of the middle level of hierarchy of managing the company. MES Concepts cover also analytic description of all main functions of the system of this kind, review of the main range of application and the main members of the domestic and external economic markets of the most famous developers of these systems.

Thereby after the creation and confirmation of the MES Concept as a common a normative-methodological branch basis all main hierarchy levels of managing the objects of the Ukraine's gaz-transport system.

4. The modified flexible Feature-Driven Development

For the effective support of the design phases and the software engineering (SW) of a complex multilevel distributed control system of the gaz-transport company the modified Feature-Driven Development (mFDD) based on expanded for the creation of multilevel distributed applications from the pattern library (design patterns) is suggested. In addition the modification of the methodology aimed at creation of the set of unified methodologies and technologies oriented on engineering specificity of the modern three-level distributed management-information systems.

The suggested methodology m FDD bases on the following systems engineering principles. The software engineering is represented as a single integral system process containing a set of EFT and the developers of these projects. For a well-organized low-level working at the project the unified FDD-batches are used as instruments for the automated engineering documentation and the realization of the unified basic processes within the projects.

5. The evolutionary strategy and the corporative creation and adoption technology of the system

As a principle of the creation and the adoption of ADCS GTC the evolutionary strategy was taken, that realizes the modern screw model of the system's creation.

The practical realization of the suggested evolutionary strategy bases on the multiphase technology of the EFT-engeneering. In the first phases as a result of the realization of basic EFT a common expandable data-pump of the system based on the adoption of the chosen set (batch) of the universal purchased parametrized components is created. In addition in the first phases the integration of new and inherited applications is carried out.

In the further phases the expansion of the data pump functions and also the engineering, integration and adoption of the new applications , which automatize isolated, specific and ad-hoc functional task complexes.

For the efficient regulation, unification and standardization of realization of the suggested evolutionary strategy the common corporate methodology control of the system's project maintenance control was worked out.

For a more convenient application by users (project teams) a burst of normative-methodological documents was created, which includes user guide, interactive handbook on how to use the principal provisions of the methodology, tutor with many working models.

Conclusion

As a result of the pursued researches a range of conceptual, methodological, strategic provisions and engineering developments was suggested, it allows ensuring of efficient support of the engineering processes, design, adoption, maintenance and development of multilevel distributed control systems aimed at the gay-transport branch.

To support the design phases and the phases of speciality application-dependent software of ADCS GTC efficiently a modified FDD-methodology based on expanded for the creation of multilevel distributed applications from the pattern library (design patterns) is suggested. As a principle of the creation and the adoption of ADCS GTC the evolutionary strategy was taken, that realizes the modern screw model of the system's creation. For the efficient regulation, unification and standardization of realization of the suggested evolutionary strategy the common corporate methodology control of the system's project maintenance control was worked out.

Bibliography

[Pavlenko, 2007] E.P.Pavlenko, V.P.Borisenko, A.G. Starovoytov, T.I. Borisenko. Complex methodological approach and flexible CASE-technology of data development of the Web-based IMS. 519-5202 international scientific conference "Modern IMS . Problems and trends of development". Conference reports– Kharkov: KHNURE, 2007

Authors' Information

Viktor Borysenko – *head of IT-department, scientific secretary of the Institute Council, Scientific-research and design Institute of Gas Transport, Marshal-Konev-Street, 16, Kharkov, 61004, Ukraine, e-mail: vborisenko@itransgaz.com*

Bogdan Kluk *– director on scientific issues, perspective development and external economic activity of the subsidiary "Ukrtransgaz," Klovskiy Uzviz, 9/1, Kiev, 01021,Ukraine; e-mail: ukrtransgas.utg@naftogaz.net*

Jury Ponomarev *– deputy director on research issues, Scientific-research and design Institute of Gas Transport, Marshal-Konev-street, 16, Kharkov, 61004, Ukraine, e-mail: ponomarev@itransgaz.com*

Anton Starovoytov – *director general of the Ukranian-German enterprise ProfITsoft, Kosmicheskayastr, 21, Kharkov, 61145, Ukraine; e-mail: anthony@profitsoft.com.ua*

KEY AGREEMENT PROTOCOL (KAP) BASED ON MATRIX POWER FUNCTION*

Eligijus Sakalauskas, Narimantas Listopadskis, Povilas Tvarijonas

***Abstract**: The key agreement protocol (KAP) is constructed using matrix power functions. These functions are based on matrix ring action on some matrix set. Matrix power functions have some indications as being a one-way function since they are linked with certain generalized satisfiability problems which are potentially NP-Complete. A working example of KAP with guaranteed brute force attack prevention is presented for certain algebraic structures. The main advantage of proposed KAP is considerable fast computations and avoidance of arithmetic operations with long integers.*

***Keywords**: key agreement protocol, matrix power function, one-way function (OWF).*

***ACM Classification Keywords**: E.3 Data encryption, F.2.1 Numerical Algorithms and Problems.*

***Conference**: The paper is selected from Sixth International Conference on Information Research and Applications – i.Tech 2008, Varna, Bulgaria, June-July 2008*

Introduction

After the sound Diffie-Hellman key agreement protocol (KAP) some attempts have been made to construct this protocol using hard problems in infinite non-commutative groups. The ideas were based on either conjugator search problems or decomposition problems (double co-set problems) which were reckoned as potentially hard problems for construction of one-way functions (OWF). One of the first ideas appeared in [Sidelnikov et. al., 1993]. From this time main attempts were directed to the suitable platform group or semigroup selection.

In 1999, first algorithms appeared using braid groups as a platform groups. In [Anshel et. al., 1999] the KAP was based on both simultaneous multiple conjugator search problem and so–called membership problem. Authors pointed out that the realization of proposed algorithm could be perspective using braid groups. In [Ko et. al., 1999] the multiple conjugator search problem in braid groups was used.

But nevertheless, it was pointed out [Shpilrain and Ushakov, 2004], that using conjugator search problem in braid groups is unnecessary and insufficient condition for KAP security. Moreover, authors noticed that the main problem for construction of cryptographic primitives in infinite non-commutative groups is to reliably hide the factors in the group word. In some groups the hiding procedure can take almost the same resources as to reveal these factors. Hence, one of the directions of investigations in this field is to combine together at least two hard problems in infinite non-commutative groups [Shpilrain and Ushakov, 2005].

The papers presented above can be interpreted as an investigation direction based on hard problems in infinite non-commutative group presentation level, i.e. using the group combinatorial theory [Magnus et. al., 1966]. This approach is also named symbolic computation.

The cryptographic application of group or semigroup action in finite dimensional vector spaces or, more generally, in some module is presented in [Monico, 2002]. This action is related with multidimensional generalization of classical modular exponent in cyclic group. This generalization pretends to be an OWF with higher complexity when compared with one based on classical exponent function in cyclic group related with discrete logarithm problem (DLP).

The idea to use non-commutative infinite group (e.g. braid group) representation was also used for construction of the other kind of OWFs as a background of both digital signature scheme and key agreement protocol [Sakalauskas, 2005], [Sakalauskas et. al., 2005]. The (semi)group representation level allows us to hide the

* Work is partially supported by the Lithuanian State Science and Studies Foundation

factors in the publicly available group word in a very natural way. However, the original hard problems, such as conjugator search or decomposition problems in (semi)group presentation level are considerably weakened when they are transferred to the representation level. Therefore in this case these problems must be considerably strengthened by simultaneously adding other additional hard problems.

The construction of KAP presented there is based on some matrix semiring $\mathcal{R}$ action on matrix set $\mathcal{M}$. The set $\mathcal{M}$ is not specified as a closed set with respect to some internal operation. Both $\mathcal{R}$ and $\mathcal{M}$ are defined over two different algebraic structures. $\mathcal{R}$ is defined over some commutative semiring $\mathcal{S}$ and $\mathcal{M}$ over some finite semigroup $\mathcal{T}$. The KAP is constructed using two external action operations of $\mathcal{R}$ on $\mathcal{M}$. These operations are named matrix power functions and were used for matrix power S-box construction [Sakalauskas and Luksys, 2007]. In some sense they are linked with well–known decomposition problem in infinite non-commutative (semi)groups [Shpilrain and Ushakov, 2005], but in contrary they are based on external action operation. The functions so defined have some indications as being one-way functions (OWF).

Matrix power functions

The classical definitions and notations in this section can be found in [Van der Waerden, 1967] and [Birkhoff and Bartee, 1974]. Let $\mathcal{R}$ be a matrix semiring consisting of m-dimensional square matrices with entries in some commutative semiring $\mathcal{S}$, i.e. $\mathcal{R}$ is a matrix semiring over $\mathcal{S}$. The elements of $\mathcal{R}$ we call a set of operators and denote them by X, Y, Z, and etc. The matrix edition and multiplication in $\mathcal{R}$ are defined in a convenient way, so since $\mathcal{S}$ is commutative, the matrix multiplication satisfies the associative law. We assume that these operators (matrices) are acting on some set of m-dimensional square matrices denoted by $\mathcal{M}$ over some finite semigroup $\mathcal{T}$. Hence we defined some action of matrix ring $\mathcal{R}$ on a set of matrices in $\mathcal{M}$. More precisely this action is the action of elements of $\mathcal{R}$ on elements of $\mathcal{M}$ in a particular way, i.e. for any $X\in\mathcal{R}$ there exists some action function f_X: $\mathcal{M}\to\mathcal{M}$. Then for all $Q\in\mathcal{M}$ and all $X\in\mathcal{R}$ there exist some A in $\mathcal{M}$, such that $f_X(Q)=A$. Hence we assumed that set $\mathcal{M}$ is closed under the action of $\mathcal{R}$. According to classical definition, the action function corresponds to the left composition function $f_X(\ )$ which arguments are in $\mathcal{M}$. Then for any such function $f_X(\ )$ the corresponding left action operation can be defined, which we denote by $\triangleright$: $\mathcal{R}\times\mathcal{M}\to\mathcal{M}$ and

$$f_X(Q)=X\triangleright Q=A \qquad (1)$$

Alternatively, assume that for any left composition function $f_X(\ )$ on $\mathcal{M}$ there exists right composition function $(\ )f_X$. Analogously to the action of left compositions functions we can define the corresponding right action operation which we denote by $\triangleleft$: $\mathcal{M}\times\mathcal{R}\to\mathcal{M}$. Then for any $Y\in\mathcal{R}$ there exists some $B\in\mathcal{M}$ satisfying equation

$$(Q)f_Y=Q\triangleleft Y=B. \qquad (2)$$

Definition 1. Functions $f_X(\ )$, $(\ )f_Y$ and the corresponding action operations $\triangleright$, $\triangleleft$ are bi-associative, if

$$(X\triangleright Q)\triangleleft Y=X\triangleright(Q\triangleleft Y). \qquad (3)$$

Further action operations $\triangleright$, $\triangleleft$ we interpret as functions. These functions are defined in abstract algebraic structures. For KAP construction we present below a more concrete realization of these functions.

Using matrix notation we write matrices as sets of their elements, i.e. $X=\{x_{ij}\}$, $Q=\{q_{jk}\}$, $Y=\{y_{ki}\}$, $A=\{a_{ik}\}$ and $B=\{b_{ji}\}$. Since matrices are of the m-th order then the indexes are $i,j,k\in\{1\ldots, m\}$.

To define the left action function $\triangleright$ of X on Q yielding the matrix A, we write the following formula relating the elements of these matrices

$$a_{ik}=\prod_{j=1}^{m} q_{jk}^{x_{ij}} \qquad (4)$$

Analogously, the result of right action operation $\triangleleft$ of matrix Y on Q is the matrix B which entries satisfies the following equations

$$b_{ji} = \prod_{k=1}^{m} q_{jk}^{y_{ki}} . \quad (5)$$

These functions were introduced in [Sakalauskas and Luksys, 2007] for the matrix power S-box construction.

To illustrate action of functions $\triangleright$ and $\triangleleft$ let us assume that matrices A, B, X, Q and Y are of the 2-nd order, i.e. having two rows and two columns. Then m=2 and (4), (5) can be rewritten in the form

$$A = X \triangleright Q = \begin{pmatrix} x_{11} & x_{12} \\ x_{21} & x_{22} \end{pmatrix} \triangleright \begin{pmatrix} q_{11} & q_{12} \\ q_{21} & q_{22} \end{pmatrix} = \begin{pmatrix} q_{11}^{x_{11}} q_{21}^{x_{12}} & q_{12}^{x_{11}} q_{22}^{x_{12}} \\ q_{11}^{x_{21}} q_{21}^{x_{22}} & q_{12}^{x_{21}} q_{22}^{x_{22}} \end{pmatrix}, \quad (6)$$

$$B = Q \triangleleft Y = \begin{pmatrix} q_{11} & q_{12} \\ q_{21} & q_{22} \end{pmatrix} \triangleleft \begin{pmatrix} z_{11} & z_{12} \\ z_{21} & z_{22} \end{pmatrix} = \begin{pmatrix} q_{11}^{y_{11}} q_{12}^{y_{21}} & q_{11}^{y_{12}} q_{12}^{y_{22}} \\ q_{21}^{y_{11}} q_{22}^{y_{21}} & q_{21}^{y_{12}} q_{22}^{y_{22}} \end{pmatrix}. \quad (7)$$

As we see functions $\triangleright$ and $\triangleleft$ can be interpreted as left and right matrix power operations. Then using the analogy to the power (exponent) function defined in certain algebraic structures (say, in a ring of integers $\mathcal{Z}_n$) the action operations can be rewritten in the form reflecting the left and right matrix power operations

$$X \triangleright Q = {}^{X}Q = A, \quad (8)$$

$$Q \triangleleft Y = Q^{Y} = B. \quad (9)$$

Definition 2. Functions $\triangleright$ and $\triangleleft$ we define as left and right matrix power functions, correspondingly.

These functions are properly defined if powering operations of q_{jk} by the elements of x_{ij} and y_{ki} have a sensible meaning. In the most simple (but practically significant) case the semiring $\mathcal{S}$ can be assumed as being a semiring of natural numbers $\mathcal{N}$={1, 2, 3 …}, i.e. $\mathcal{S}$= $\mathcal{N}$. Then the variables x_{ij} and y_{ki} are natural numbers and the elements of matrices A and B denoted by $\{a_{ik}\}$ and $\{b_{ik}\}$ in (4) and (5) can be calculated by powering the elements q_{jk} in finite semigroup $\mathcal{T}$ by natural numbers x_{ik} and y_{ik} using the multiplication operation defined in $\mathcal{T}$.

The following theorem can be formulated for functions $\triangleright$ and $\triangleleft$.

Theorem 1. If Z=XY, where X, Y and Z are in $\mathcal{M}$, then

$$Z \triangleright Q = (XY) \triangleright Q = X \triangleright (Y \triangleright Q) = X \triangleright Y \triangleright Q. \quad (10)$$

$$Q \triangleleft Z = Q \triangleleft (XY) = Q \triangleleft (X \triangleleft Y) = Q \triangleleft X \triangleleft Y. \quad (11)$$

▼ Proof. The proof directly follows from the (4), (5) and the rule of convenient matrix multiplication in $\mathcal{R}$. ▲

Theorem 2. . If $\mathcal{S}$=$\mathcal{N}$, then functions $\triangleright$ and $\triangleleft$ are bi-associative.

▼ Proof. Since the elements of matrices X and Y are the natural numbers in $\mathcal{N}$, then for all $q_i \in \mathcal{T}$ and $x_i, y_k \in \mathcal{N}$, the following exponentiation rules in $\mathcal{T}$ are valid $\left(q_j^{x_i}\right)^{y_k} = \left(q_j^{y_k}\right)^{x_i} = q_j^{x_i y_k} = q_j^{y_k x_i}$ and $q_j^{x_i} q_j^{y_k} = q_j^{x_i + y_k}$.

Using association law of matrix multiplication in $\mathcal{R}$ and (4), (5), and applying direct calculations we find that $(X \triangleright Q) \triangleleft Y = X \triangleright (Q \triangleleft Y) = D$, where $D=\{d_{ij}\}$ is the matrix in $\mathcal{M}$. ▲

Key agreement protocol

Using a combination of functions $\triangleright$ and $\triangleleft$ we construct the key agreement protocol (KAP). It is based on the conjecture that these functions are one-way functions (OWFs). Let us define two subsets of commuting matrices $\mathcal{R}_L$ and $\mathcal{R}_R$ in $\mathcal{R}$. This means that for all $X, U \in \mathcal{R}_L$ and $Y, V \in \mathcal{R}_R$

$$XU = UX, \quad (12)$$

$$YV = VY. \quad (13)$$

Then we propose the following KAP.

1. Parties agree on publicly available matrix Q in $\mathcal{M}$ and two subsets $\mathcal{R}_L$ and $\mathcal{R}_R$ in $\mathcal{R}$.

Alice chooses at random the secret matrix X in $\mathcal{R}_L$ and Y in $\mathcal{R}_R$, respectively, calculates matrix A and sends it to Bob, where

$$A=X^{\triangleright} Q^{\triangleleft} Y. \quad (14)$$

2. Bob chooses at random the secret matrix U in $\mathcal{R}_L$ and V in $\mathcal{R}_R$ respectively, calculates matrix B and sends it to Alice, where

$$B=U^{\triangleright} Q^{\triangleleft} V. \quad (15)$$

3. Both parties compute the following common secret key K:

$$K=X^{\triangleright} B^{\triangleleft} Y= X^{\triangleright} U^{\triangleright} Q^{\triangleright} V^{\triangleleft} Y=U^{\triangleright} X^{\triangleright} Q^{\triangleright} Y^{\triangleleft} V=U^{\triangleright} A^{\triangleleft} V. \quad (16)$$

The last identities are valid since Theorems 1, 2 and equations (12), (13) hold.

The proposed KAP is some generalization of well known Diffie-Hellman protocol. Indeed, if all matrices are numbers in Galois field GF(p) then according to (4), (5), and (16) we can write

$$X^{\triangleright} Q^{\triangleleft} Z= {}^{X}Q^{Y}=Q^{XY}=K, \quad (17)$$

where K is a Diffie-Hellman secret key.

To compromise the secret key K one must find any matrices X, Y in (14) and U, V in (15) for given instances Q, A and Q, B correspondingly. Let us consider the case to find any matrices X, Y in (14). Let the elements of X, Y, Q and A are $\{x_{ij}\}$, $\{y_{ij}\}$, $\{q_{jk}\}$ and $\{a_{ik}\}$ correspondingly. For more clarity the matrix equation (14) we write in a form of the system of equations for the matrices of 2-nd order, i.e. for m=2:

$$\begin{cases} q_{11}^{x_{11}y_{11}} & q_{21}^{x_{12}y_{11}} & q_{12}^{x_{11}y_{21}} & q_{22}^{x_{12}y_{21}} & = & a_{11} \\ q_{11}^{x_{11}y_{12}} & q_{21}^{x_{12}y_{12}} & q_{12}^{x_{11}y_{22}} & q_{22}^{x_{12}y_{12}} & = & a_{12} \\ q_{11}^{x_{21}y_{11}} & q_{21}^{x_{22}y_{11}} & q_{12}^{x_{21}y_{21}} & q_{22}^{x_{22}y_{21}} & = & a_{21} \\ q_{11}^{x_{21}y_{12}} & q_{21}^{x_{22}y_{12}} & q_{12}^{x_{21}y_{22}} & q_{22}^{x_{22}y_{22}} & = & a_{22} \end{cases} \quad (18)$$

At the first sight it seems that the problem to find any $X=\{x_{ij}\}$, $Y=\{y_{ij}\}$ is some matrix generalization of discrete logarithm problem (DLP). But nevertheless the solution of DLP is not a sufficient condition to find X and Y even in the case when $\mathcal{T}$ is a group of Galois field GF(p). If we choose some matrix Y in $\mathcal{R}_R$ and will try to find X by solving (14), there is no guarantee that obtained matrix X will be in $\mathcal{R}_L$. Hence the compromising of K is related with the solution of matrix equation (15). This equation for m=2 is presented in (18).

Without proof we declare that the security of proposed KAP relies on the complexity of certain generalized satisfiability problem which conveniently is denoted by SAT(S), [Shaefer, 1978]. According to Shaefer Dichotomy theorem the SAT(S) problem is either P or NP-Complete, [Garey and Johnson, 1979]. The first alternative is rather a very rare exception since the conditions of SAT(S) problem to be in class P occurs in a very special predetermined cases, [Shaefer, 1978]. Hence the key K compromisation with a very big certainty corresponds to the solution of NP-Complete problem.

In contrary to the classical Diffie-Hellman protocol, we think that one of advantages of there proposed protocol is the avoidance of performing arithmetic operations with big integers and faster computations.

Implementation

The concrete realization of KAP requires defining both the matrix semiring $\mathcal{R}$ over commutative semiring $\mathcal{S}$ and the set of matrices $\mathcal{M}$ over the semigroup $\mathcal{T}$. As it was denoted above, we can choose $\mathcal{S}=\mathcal{N}$, when $\mathcal{T}$ was assumed to be finite semigroup. The most known types of $\mathcal{T}$ can be either a semigroup Z_n^* of ring of integers $\mathcal{Z}_n$, or the group $\mathcal{F}^*$ of some Galois (finite) field $\mathcal{F}$, or a group of Elliptic Curve points in some finite field $\mathcal{F}$.

In any case when $\mathcal{T}$ is a semigroup neither matrix multiplication nor addition are defined in $\mathcal{M}$. Hence we have specified $\mathcal{M}$ as a set without any internal operations. As an example we can choose $\mathcal{T}$=GF(251).

We think that essential security parameters in our construction are the order of matrices m and the logarithm of cardinality of $\mathcal{T}$, which we denote by $N=\lceil \log_2 n \rceil$. When $\mathcal{T}$=GF(251), $N=\lceil \log_2 n \rceil=\lceil \log_2 251 \rceil=8$. Let the other security parameter m=32. Then we have the matrix Q in $\mathcal{M}$ of order 32 with elements in GF(251). In this case the matrices X and Y are represented by $k=mxmxN=32x32x8=8192$ bits.

Bibliography

[Anshel et. al., 1999], I. Anshel, M. Anshel and D. Goldfeld, An algebraic method for public-key cryptography, Math. Res. Lett. 6, 1999, pp. 287-291.

[Birkhoff and Bartee, 1974], G. Birkhoff and C. Bartee, Modern applied algebra, McGraw-Hill, 1974.

[Garey and Johnson, 1979], M. Garey and D. Johnson, Computers and intractability: a guide to theory of NP-Completeness. H.Freeman, New York, 1979.

[Ko et. al., 1999], K. H. Ko, S. J. Lee, J. H. Cheon, J. W. Han, J. Kang and C. Park, New public-key cryptosystem using braid groups, Advances in cryptology, Advances in Cryptology, Proc. Crypto 2000, LNCS 1880, Springer–Verlag 2000, pp. 166-183.

[Magnus et. al., 1966], W. Magnus, A. Karrass and D. Solitar, Combinatorial Group Theory, Interscience Publishers, NY, 1966.

[Monico, 2002], C. Monico, Semirings and Semigroup actions in Public–Key Cryptography, Phd. thesis, University of Notre Dame, May 2002, pp. 1-78.

[Sakalauskas, 2005], E. Sakalauskas, One Digital Signature Scheme in Semimodule over Semiring, Informatica, ISSN: 0868-4952, Vol. 16, No. 3, 2005, pp. 383-394.

[Sakalauskas et. al., 2007], E. Sakalauskas, P. Tvarijonas and A. Raulynaitis, Key Agreement Protocol (KAP) Using Conjugacy and Discrete Logarithm Problems in Group Representation Level, Informatica, Vol. 18, No. 1, 2007, pp. 115-124.

[Sakalauskas and Luksys, 2007], E. Sakalauskas and K. Lukšys, Matrix Power S-Box Construction,*Cryptology . ePrint Archive: Report*, no. 214 (2007), http://eprint.iacr.org/2007/214.

[Sidelnikov et. al., 1993], V. Sidelnikov, M. Cherepnev and V. Yaschenko, Systems of open distribution of keys on the basis of noncommutative semigroups. Russian Acad. Sci. Dokl. Math., 48(2), 1993, pp. 566-567.

[Shaefer, 1978], T. J. Shaefer, The Complexity of Satisfiability Problems, *Proceedings of the 10th Annual Symposium on Theory and Computing*, 1978, pp. 216-226.

[Shpilrain and Ushakov, 2004], V. Shpilrain and A. Ushakov, The conjugacy search problem in public key cryptography: unnecessary and insufficient, Available at: http://eprint.iacr.org/2004/321, 2004.

[Shpilrain and Ushakov, 2005], V. Shpilrain and A. Ushakov, A new key exchange protocol based on the decomposition problem, Available at: http://eprint.iacr.org/2005/447, 2005.

[Van der Waerden, 1967], B. L. van der Waerden, Algebra, Springer-Verlag, 1967.

Authors' Information

Eligijus Sakalauskas *– Assoc. prof., Department of Applied Mathematics, Kaunas University of Technology, Studentu str. 50-324a, Kaunas, LT-51368, Lithuania, e-mail Eligijus.Sakalauskas@ktu.lt*

Narimantas Listopadskis *– Assoc. prof., Department of Applied Mathematics, Kaunas University of Technology, Studentu str. 50-324a, Kaunas, LT-51368, Lithuania, e-mail Narimantas.Listopadskis@ktu.lt*

Povilas Tvarijonas *– Lector, Department of Applied Mathematics, Kaunas University of Technology, Studentu str. 50-324a, Kaunas, LT-51368, Lithuania, e-mail Povilas.Tvarijonas@ktu.lt*

MATRIX POWER S-BOX ANALYSIS[1]

Kestutis Luksys, Petras Nefas

***Abstract**: Construction of symmetric cipher S-box based on matrix power function and dependant on key is analyzed. The matrix consisting of plain data bit strings is combined with three round key matrices using arithmetical addition and exponent operations. The matrix power means the matrix powered by other matrix. This operation is linked with two sound one-way functions: the discrete logarithm problem and decomposition problem. The latter is used in the infinite non-commutative group based public key cryptosystems. The mathematical description of proposed S-box in its nature possesses a good "confusion and diffusion" properties and contains variables "of a complex type" as was formulated by Shannon. Core properties of matrix power operation are formulated and proven. Some preliminary cryptographic characteristics of constructed S-box are calculated.*

***Keywords**: Matrix power, symmetric encryption, S-box.*

***ACM Classification Keywords**: E.3 Data Encryption, F.2.1 Numerical Algorithms and Problems.*

***Conference**: The paper is selected from Sixth International Conference on Information Research and Applications – i.Tech 2008, Varna, Bulgaria, June-July 2008*

Introduction

As it is known, the design criteria for the block ciphers as for other cryptographic systems are related with the known cryptoanalytic attacks. It is essential that after the new attack invention the old design criteria must be changed.

Traditional design criteria are oriented to the most powerful attacks such as linear and differential and were successfully satisfied for the several known ciphers, for example AES, Serpent, Camellia Misty/Kasumi etc. It was shown that the non-linearity properties of the inverse function in $GF(2^n)$ used as a single non-linear component in AES are close to optimality with respect to linear, differential and higher-order differential attacks [Canteaut and Videau, 2002].

But nevertheless it is shown that many known "optimal" ciphers have a very simple algebraic structure and are potentially vulnerable to the algebraic attack. This attack was declared in [Schaumuller-Bihl, 1983] and developed in [Courtois and Pieprzyk, 2002]. The vulnerability is related to S-box description by implicit input/output and key variables algebraic equations of polynomial type. For example the AES can be described by the system of multivariate quadratic equations in GF (2^8) for which the XL or XSL attack can be applied in principle. Then there is a principal opportunity to find the solution of these equations by some feasible algorithm that might be of sub-exponential time and recover the key from a few plaintext/ciphertext pairs.

The algebraic attack changes some old security postulates [Courtois, 2005]:

1. The complexity is no longer condemned to grow exponentially with the number of rounds.
2. The number of required plaintexts may be quite small (e.g. 1).
3. The wide trail strategy should have no impact whatsoever for the complexity of the attack.

Despite the fact that there are no practical results of breaking the entire AES by algebraic attack yet, it is sensible to build the new design methods possessing a higher resistance to algebraic attack. According to Courtois the design of ciphers will never be the same again and this is supported by the declared new ideas for the S-box

[1] Work supported by the Lithuanian State Science and Studies Foundation

construction laying on the sufficiently large random S-boxes to prevent all algebraic attacks one can think [Courtois et al., 2005].

In this paper we further discus so called matrix power operation introduced in [Sakalauskas and Luksys, 2007] for a matrix power S-box construction. Matrix power S-box is based on modular exponentiation over GF(2^n). This leads to some generalization of discrete logarithm problem (DLP) using a matrix group action problem over Galois field.

The idea to use the group or semigroup action problem in vectorial spaces for the asymmetric cryptographic primitives' construction can be found in [Monico, 2002]. We have generalized this approach and applied it to our S-box construction. As a result we have obtained some one way function (OWF) which is linked not only with a classical DLP but also with so called decomposition problem (DP), used in the asymmetric cryptosystems based on the hard problems in infinite non-commutative groups [Shpilrain and Ushakov, 2005]. The same kind of DP is used also in digital signature scheme and key agreement protocol construction [Sakalauskas, 2005] and [Sakalauskas et al., 2007].

Preliminaries

Let us define m x m matrices over $GF(2^n)$. The set of all those matrices over $GF(2^n)$ we denote as **M**. Plaintext and ciphertext data is represented in this set. We do not introduce any internal operations in the set **M**. For further considerations we are interested only in external operations performed in this set.

Let $\mathbf{M_G}$ be a group of m x m matrices over N_{2n-1} with the commonly defined matrix multiplication operation and matrix inverse. Keys' matrices should be chosen from $\mathbf{M_G}$.

Matrix group $\mathbf{M_G}$ left and right action operations in the set M are denoted by $\triangleright$ and $\triangleleft$ respectively.

In a formal way $\triangleright$ is a mapping $\triangleright: \mathbf{M_G} \times \mathbf{M} \to \mathbf{M}$ and $\triangleleft: \mathbf{M} \times \mathbf{M_G} \to \mathbf{M}$. Then $\forall L, R \in \mathbf{M_G}$ and $\forall X \in \mathbf{M}$ there exist some $Y, Z \in \mathbf{M}$ such that $L \triangleright X = Y$ and $X \triangleleft R = Z$.

The elements of matrices L, X, R, Y *and* Z we denote by the indexed set of its elements respectively, i.e. by $\{x_{ij}\}$ we denote matrix X.

We have chosen the following action operations which can be written for the matrix equation $L \triangleright X = Y$ elements

$$y_{ij} = \prod_{s=1}^{m} x_{sj}^{l_{is}}\ , \tag{1}$$

and for the matrix equation $X \triangleleft R = Z$ elements

$$z_{ij} = \prod_{t=1}^{m} x_{it}^{r_{ij}}\ . \tag{2}$$

The multiplication and power operations are performed using GF (2^n) arithmetic, i.e. modulo irreducible polynomial.

Example 1. To give a simple example, let us assume that all matrices have two rows and two columns, i.e. $m = 2$. In this case, matrix Y can be expressed in the following way

$$Y = L \triangleright X = \begin{pmatrix} l_{11} & l_{12} \\ l_{21} & l_{22} \end{pmatrix} \triangleright \begin{pmatrix} x_{11} & x_{12} \\ x_{21} & x_{22} \end{pmatrix} = \begin{pmatrix} x_{11}^{l_{11}} x_{21}^{l_{12}} & x_{12}^{l_{11}} x_{22}^{l_{12}} \\ x_{11}^{l_{21}} x_{21}^{l_{22}} & x_{12}^{l_{21}} x_{22}^{l_{22}} \end{pmatrix}.$$

Matrix Z can be expressed in the following way

$$Z = X \triangleleft R = \begin{pmatrix} x_{11} & x_{12} \\ x_{21} & x_{22} \end{pmatrix} \triangleleft \begin{pmatrix} r_{11} & r_{12} \\ r_{21} & r_{22} \end{pmatrix} = \begin{pmatrix} x_{11}^{r_{11}} x_{12}^{r_{21}} & x_{11}^{r_{12}} x_{12}^{r_{22}} \\ x_{21}^{r_{11}} x_{22}^{r_{21}} & x_{21}^{r_{12}} x_{22}^{r_{22}} \end{pmatrix}.$$ □

Matrix power S-box

The S-box input data we denote by m x m matrix D with elements being binary strings in vector space F_2^{n-1}. Using the certain key expansion procedure we can generate the round keys for encryption: matrix K over F_2^{n-1} and matrices L, $R \in \mathbf{M_G}$. Input/output and key matrices are all of the same $m \times m$ size.

S-box transformations of input data D to ciphered output data C are performed as follows:

$$\boldsymbol{D + K + 1 = X}, \quad (3)$$

$$\boldsymbol{L \triangleright X \triangleleft R = C}, \quad (4)$$

where $D + K + \mathbf{1}$ denotes the ordinary arithmetical addition of matrices modulo 2^n; $\mathbf{1}$ is the matrix consisting of arithmetical unity elements in F_2^n. Combining (3) and (4) we obtain

$$\boldsymbol{L \triangleright (D + K + 1) \triangleleft R = C}. \quad (5)$$

From (3) we obtain a matrix $X \in \mathbf{M}$ which does not contain zero elements, i.e. is without zero binary strings. This is necessary because of multiplications. If there would be at least one zero element, then ciphertext will be zero matrix.

We can write now the implicit formula for an element c_{ij}:

$$c_{ij} = \prod_{t=1}^{m}\prod_{s=1}^{m} x_{st}^{l_{is} \cdot r_{tj}} = \prod_{t=1}^{m}\prod_{s=1}^{m} (d_{st} + k_{st} + 1)^{l_{is} r_{tj}}, \quad (6)$$

where 1 is a bit string corresponding to arithmetical unit in F_2^n.

Since $\mathbf{M_G}$ is a group of matrices, then there exists the inverse matrix R^{-1} such that $RR^{-1} = R^{-1}R = \boldsymbol{I}$, where $\boldsymbol{I}$ is the identity matrix.

Decryption operation can be written similarly to (5):

$$\boldsymbol{L^{-1} \triangleright C \triangleleft R^{-1} - K - 1 = D}. \quad (7)$$

Resulting matrix of inverse S-box can be expressed like this:

$$d_{ij} = \prod_{t=1}^{m}\prod_{s=1}^{m} c_{st}^{l'_{is} \cdot r'_{tj}} - k_{ij} - 1, \quad (8)$$

where $\{l'_{ij}\} = L^{-1}$ and $\{r'_{ij}\} = R^{-1}$. Thus, we have to be able to calculate inverse matrices of L and R keys for decryption. Key matrix K remains the same, only during decryption ordinary subtraction is used instead of addition.

For the validity of the last equations the left-right action operations must satisfy the following properties:

1. The action operations must be associative, i.e.

$$\boldsymbol{L_2 \triangleright (L_1 \triangleright X) = (L_2 L_1) \triangleright X}, \quad (9a)$$

$$\boldsymbol{(X \triangleleft R_1) \triangleleft R_2 = X \triangleleft (R_1 R_2)}. \quad (9b)$$

2. The action operations are both left and right invertible, i.e.

$$\boldsymbol{L^{-1} \triangleright (L \triangleright X) = (L^{-1} L) \triangleright X = I \triangleright X = X}, \quad (10a)$$

$$\boldsymbol{(X \triangleleft R^{-1}) \triangleleft R = X \triangleleft (R^{-1} R) = X \triangleleft I = X}. \quad (10b)$$

Theorem 1. The action operations are associative.

Proof. Let us consider encryption and decryption scheme with plaintext matrix X, key matrixes L and R, their inverse matrices L^{-1} and R^{-1} and cipher text matrix C. According to (4) and (7), following relations should be true:

$$L \triangleright X \triangleleft R = C,$$

$$L^{-1} \triangleright C \triangleleft R^{-1} = X.$$

For the simplicity, we omit matrix K here and consider that matrix X has no zero elements. This does not affect generality because matrix K is added before matrix power operation and subtracted after, in case of inverse S-box.

Then plaintext matrix X can be expressed following way:

$$X = L^{-1} \triangleright C \triangleleft R^{-1} = L^{-1} \triangleright \left(L \triangleright X \triangleleft R\right) \triangleleft R^{-1} = L^{-1} \triangleright \left\{ \prod_{t=1}^{m} \prod_{s=1}^{m} x_{st}^{l_{is} \cdot r_{tj}} \right\} \triangleleft R^{-1} = \left\{ \prod_{v=1}^{m} \prod_{u=1}^{m} \left(\prod_{t=1}^{m} \prod_{s=1}^{m} x_{st}^{l_{us} \cdot r_{tv}} \right)^{l'_{iu} \cdot r'_{vj}} \right\} =$$

$$= \left\{ \prod_{v=1}^{m} \prod_{u=1}^{m} \prod_{t=1}^{m} \prod_{s=1}^{m} x_{st}^{l_{us} \cdot r_{tv} \cdot l'_{iu} \cdot r'_{vj}} \right\} = \left\{ \prod_{t=1}^{m} \prod_{s=1}^{m} \prod_{v=1}^{m} \prod_{u=1}^{m} x_{st}^{l_{us} \cdot r_{tv} \cdot l'_{iu} \cdot r'_{vj}} \right\} = \left\{ \prod_{t=1}^{m} \prod_{s=1}^{m} x_{st}^{\sum_{v=1}^{m} \sum_{u=1}^{m} l_{us} \cdot r_{tv} \cdot l'_{iu} \cdot r'_{vj}} \right\} =$$

$$= \left\{ \prod_{t=1}^{m} \prod_{s=1}^{m} x_{st}^{\sum_{u=1}^{m} l_{us} \cdot l'_{iu} \cdot \sum_{v=1}^{m} r_{tv} \cdot r'_{vj}} \right\} = \left\{ \prod_{t=1}^{m} \prod_{s=1}^{m} x_{st}^{l''_{ik} \cdot r''_{tj}} \right\} = L'' \triangleright X \triangleleft R'' = \left(L^{-1} L\right) \triangleright X \triangleleft \left(R R^{-1}\right)$$

Thus we receive that

$$L^{-1} \triangleright \left(L \triangleright X \triangleleft R\right) \triangleleft R^{-1} = \left(L^{-1} L\right) \triangleright X \triangleleft \left(R R^{-1}\right). \quad \textbf{(11)}$$

We used ordinary features of power function and matrix multiplication, so this equation holds for any matrices. □

Lemma 1. Defined matrix power operation has the following property: if key matrices are identities, then resulting matrix of ciphertext is equal to the matrix of plaintext.

Proof. Any element of ciphertext matrix can be written following way:

$$c_{ij} = t_{ij}^{l_{ii} \cdot rjj} \cdot \prod_{u=1}^{m} \prod_{\substack{v=1 \\ v \neq j \& u \neq i}}^{m} t_{vu}^{r_{iv} l_{uj}} .$$

If key matrices are $L = R = \boldsymbol{I}$, then we have that $l_{ii} = r_{jj} = 1$ for any i and j from1 to m, and $l_{ij} = r_{jj} = 0$, if $i \neq j$:

$$c_{ij} = t_{ij}^{1 \cdot 1} \cdot \prod_{u=1}^{m} \prod_{\substack{v=1 \\ v \neq j \& u \neq i}}^{m} t_{vu}^{0} = t_{ij} . \qquad \square$$

Theorem 2. The action operations are both left and right invertible.

Proof. According to (11) and Lemma 1 we obtain:

$$L^{-1} \triangleright \left(L \triangleright X \triangleleft R\right) \triangleleft R^{-1} = \left(L^{-1} L\right) \triangleright X \triangleleft \left(R R^{-1}\right) = \boldsymbol{I} \triangleright X \triangleleft \boldsymbol{I} = X . \quad \textbf{(12)} \ \square$$

Key matrices

Key matrices L and R should be invertible in order that defined matrix power S-box would be bijective, i.e. would have inverse S-box. This is obvious from the (12). Decryption of the ciphertext can be done only with L^{-1} and R^{-1} matrices. If L or R did not have inverse, then matrix power S-box is surjective and inverse S-box does not exist.

Matrices L and R are chosen from group $\mathbf{M_G}$, therefore they have inverse matrices. The problem is, how to construct group $\mathbf{M_G}$ that it would be large enough and brute force attacks would be useless.

One of the methods is to use the certain non-commutative group representation in the set of matrix group $GL(m, GF(2^n))$. The non-commutative group is presented by finite sets of generators and relations. Then it is required to construct representation matrices and their inverses for each initial group generator [Sakalauskas and Luksys, 2007].

Other method is to generate random matrices and to check if they are invertible. We have used this method to evaluate key space and matrix power S-box security properties.

For the analysis, we have chosen 3 x 3 matrices (m = 3) and n = 7. In this case key matrices L and R are over N_{127} and data matrices are over $GF(2^7)$. Irreducible polynomial was $x^7 + x + 1$.

Key matrices L and R have nine elements and each element can be randomly chosen from 127 values. Thus we can generate $127^9 \approx 2^{62.9}$ distinct matrices. But that would be all possible variants, including those matrices, which do not have inverse. We have generated 2^{34} matrices and 0.969% of those matrices were not invertible. Therefore rough estimate of matrix power key space would be around 2^{63} (for two key matrices).

Table 1. Cryptographic characteristics of 500 000 random invertible matrix power S-boxes

Group No.	Bijective	Algebraic degree	Nonlinearity	*k-uniform*	Algebraic quadratic equations immunity	Algebraic biaffine equations immunity	Percentage, %
1.	T	4	56	2	$2^{19,53}$	-	10,9
2.	T	6	54	2	$2^{15,63}$	$2^{12,68}$	5,5
3.	T	5	44	4	$2^{19,53}$	$2^{19,65}$	3,5
4.	T	5	44	4	$2^{19,53}$	$2^{19,44}$	1,8
5.	T	5	44	4	$2^{19,53}$	$2^{19,23}$	0,2
6.	T	5	44	6	$2^{13,84}$	2^{12}	10,9
7.	T	4	56	2	$2^{10,75}$	$2^{19,65}$	3,5
8.	T	4	56	2	$2^{10,75}$	$2^{19,44}$	1,8
9.	T	4	56	2	$2^{10,75}$	$2^{19,23}$	0,2
10.	T	4	56	2	$2^{11,72}$	2^{12}	11,0
11.	T	3	56	2	$2^{19,53}$	2^{999}	11,0
12.	T	3	44	4	$2^{19,53}$	$2^{19,65}$	3,6
13.	T	3	44	4	$2^{19,53}$	$2^{19,44}$	1,8
14.	T	3	44	4	$2^{19,53}$	$2^{19,23}$	0,2
15.	T	3	44	6	$2^{13,84}$	2^{12}	11,0
16.	T	2	56	2	$2^{10,75}$	$2^{19,65}$	3,5
17.	T	2	56	2	$2^{10,75}$	$2^{19,44}$	1,8
18.	T	2	56	2	$2^{10,75}$	$2^{19,23}$	0,2
19.	T	2	56	2	$2^{11,72}$	2^{12}	10,9
20.	T	1	0	128	$2^{7,814}$	$2^{6,34}$	5,5
21.	F	7	0	2	$2^{7,17}$	$2^{6,34}$	1,5
22.	F	7	0	2	$2^{7,135}$	$2^{6,294}$	0,1

It is very difficult to evaluate cryptographic characteristics of S-box with 54 bit input and 63 bit output. Therefore we have made two simplifications for the security analysis. First of all we did not do key addition (3). If data matrix had zero element (-s) that matrix was left unchanged. This let us to analyze S-box with equal input and output size. For the second simplification, we fixed all input matrix elements except one and analyzed only one particular element of the output matrix. This led us to the analysis of the S-box with input and output size of 7 bits.

We have chosen to evaluate five cryptographic characteristics: algebraic degree [Meier et al., 2004], nonlinearity, differential coefficient *k-uniform*, algebraic quadratic equations immunity and algebraic biaffine equations immunity [Courtois et al., 2005]. Algebraic immunity is calculated as follows:

$$\Gamma = \left(\frac{t}{n}\right)^{\left\lceil \frac{t}{r} \right\rceil},$$

where t is number of terms in algebraic normal form (ANF) of Boole function, n – number of variables, r – number of biaffine or quadratic equations.

We have generated 500 000 random invertible matrix power S-boxes. Analysis results are shown in Table 1.

We have grouped all generated S-boxes into 22 groups according their characteristics. Two groups of S-boxes are not bijective, i.e. the representation of one element of input matrix into one output matrix element is not bijective, but the whole matrix power S-box remains invertible. Group 20th represents S-boxes which performs a linear transformation. Characteristics of groups 1–19 are similar to those of ordinary power functions, like Gold, Kasami, Niho etc. [Cheon and Lee, 2004].

These are just preliminary results and further analysis of matrix power S-box should be done.

Conclusion

In this paper we have analyzed key depended S-box based on introduced matrix power operation. We have formulated and proven core properties of this operation.

Some preliminary cryptographic characteristics of constructed S-box are calculated. Characteristics of simplified version of matrix power S-box are similar to those of ordinary power functions, like Gold, Kasami, Niho etc.

Bibliography

[Canteaut and Videau, 2002] A. Canteaut and M. Videau. Degree of composition of highly nonlinear functions and applications to higher order differential cryptanalysis. Advances in Cryptology Eurocrypt'2002, Springer Verlag 2002.

[Cheon and Lee, 2004] J.H. Cheon, D.H. Lee. Resistance of S-boxes against Algebraic Attacks. Fast Software Encryp¬tion, LNCS 3017, pp. 83-94, Springer-Verlag, 2004.

[Courtois, 2005] N.T. Courtois. General Principles of Algebraic Attacks and New Design Criteria for Cipher Components. Advanced Encryption Standard – AES, LNCS 3373, pp. 67-83, 2005.

[Courtois et al., 2005] N.T. Courtois, B. Debraize and E. Garrido. On exact algebraic [non-]immunity of S-boxes based on power functions. Cryptology ePrint Archive: Report, no. 203 (2005), http://eprint.iacr.org/2005/203.

[Courtois and Pieprzyk, 2002] N.T. Courtois and J. Pieprzyk. Cryptanalysis of Block Ciphers with Overdefined Systems of Equations. Proceedings of Asiacrypt'2002, LNCS 2501, pp. 267-287, Springer-Verlag, 2002.

[Meier et al., 2004] W.Meier, E.Pasalic and C.Carlet. Algebraic Attacks and Decomposition of Boolean Functions. Advances in Cryptology - EUROCRYPT 2004, LNCS 3027, Springer Berlin / Heidelberg, 2004, pp. 474-491.

[Monico, 2002] C. Monico. Semirings and Semigroup actions in Public–Key Cryptography. PhD. thesis, University of Notre Dame, May 2002.

[Sakalauskas and Luksys, 2007] E.Sakalauskas and K.Luksys, Matrix Power S-Box Construction. Cryptology ePrint Archive: Report, no. 214 (2007), http://eprint.iacr.org/2007/214.

[Sakalauskas et al., 2007] E. Sakalauskas, P. Tvarijonas and A. Raulinaitis. Key Agreement Protocol (KAP) Using Conjugacy and Discrete Logarithm Problems in Group Representation Level, Informatica, Vol. 18, No. 1, 2007, pp. 115-124.

[Sakalauskas, 2005] E. Sakalauskas. One Digital Signature Scheme in Semimodule over Semiring, Informatica, vol. 16, no. 3, 2005, pp. 383-394.

[Schaumuller-Bichl, 1983] Schaumuller-Bichl. Cryptanalysis of the Data Encryption Standard by the Method of Formal Coding. Advances in Cryptology EUROCRYPT-1982, LNCS 149, Springer-Verlag, 1983, pp. 235-255.

[Shpilrain and Ushakov, 2005] V. Shpilrain and A. Ushakov. A new key exchange protocol based on the decomposition problem. Cryptology ePrint Archive: Report, no. 447 (2005), http://eprint.iacr.org/2005/447.

Authors' Information

Kestutis Luksys *– PhD student, Kaunas University of Technology, Studentu st. 50-327A, Kaunas 51368, Lithuania; e-mail: kestutis.luksys@ktu.lt.*

Petras Nefas *– Dr., Head of Division, GRC of State Security Department, Lithuania; e-mail: petras.nefas@gmail.com.*

KEY AGREEMENT PROTOCOL USING ELLIPTIC CURVE MATRIX POWER FUNCTION*

Artūras Katvickis, Paulius Vitkus

***Abstract**: The key agreement protocol (KAP) using elliptic curve matrix power function is presented. This function pretends be a one-way function since its inversion is related with bilinear equation solution over elliptic curve group. The matrix of elliptic curve points is multiplied from left and right by two matrices with entries in Z_n. Some preliminary security considerations are presented.*

***Keywords**: key agreement protocol, elliptic curve cryptography, NP-complete problem.*

***ACM Classification Keywords**: E.3 Data Encryption, F.2.1 Numerical Algorithms and Problems.*

***Conference**: The paper is selected from Sixth International Conference on Information Research and Applications – i.Tech 2008, Varna, Bulgaria, June-July 2008*

Introduction

Key agreement protocols (KAP) is one of the basic cryptographic protocols. KAP allows two or more parties negotiate a common secret key using insecure communications.

First KAP was presented by Diffie-Hellman [Diffie, Hellman, 1976] which caused rapid development of asymmetric cryptography.

In 1993 new ideas appeared in asymmetric cryptography [Sidelnikov et al, 1993] – using known hard computational problems in infinite non- commutative groups instead of hard number theory problems such as discrete logarithm or integer factorization problems to construct one-way functions.

This idea was realized in [Anshel et al, 1999] where KAP was constructed using conjugator search problem and membership problem in Braid groups. The similar result was presented in [Ko et al, 2000].

Later, [Shpilrain, Ushakov, 2004] showed that conjugator search problem does not produce sufficient security level. The others hard problems were investigated to construct KAP and were based on triple decomposition problem [Kurt, 2006], subgroup membership problem [Shpilrain, Zapata, 2006] and elliptic curve pairing [Smart, 2002].

The idea to use non-commutative infinite group (e.g. braid group) representation was also used for the other kind of one-way functions construction as a background of both digital signature scheme and key agreement protocol [Sakalauskas, 2005], [Sakalauskas et al, 2007]. The (semi)group representation level allows us to avoid a significant problem of hiding the factors in the publicly available group word when using its presentation level. The hiding of factors in representation level occurs in a very natural way. However, the original hard problems, such as conjugator search or decomposition problems in (semi)group presentation level are considerably weakened when they are transformed into the representation level. Therefore using representation level these problems must be considerably strengthened by simultaneously adding the other additional hard problems.

In this paper we present KAP using elliptic curve matrix power function. This function pretends be a one-way function since its inversion is related with bilinear equation over elliptic curve group. The matrix of elliptic curve points is left and right side multiplied by two matrices with entries in Z_n.

* Work is partially supported by the Lithuanian State Science and Studies Foundation

Mathematical background

Let $p > 3$ be a prime integer. An elliptic curve $E_p(a, b)$ over GF(p) is defined by equation

$$y^2 = x^3 + ax + b, \quad (1)$$

where $a, b \in$ GF(p) and $4a^3 + 27b^2 \bmod p \neq 0$.

The addition operation between two points $P = (x_1, y_1)$ and $Q = (x_2, y_2)$ on elliptic curve is written in following algebraic formulas:

$$\begin{aligned} x_3 &= \lambda^2 - x_1 - x_2, \\ y_3 &= \lambda(x_1 - x_3) - y_1, \end{aligned} \quad (2)$$

where $\lambda = \begin{cases} \dfrac{y_2 - y_1}{x_2 - x_1}, & P \neq Q, \\ \dfrac{3x_1^2 + a}{2y_1}, & P = Q. \end{cases}$

A set of all points (x, y), $a, b \in$ GF(p), which satisfy (1) equation, together with special point O, called infinity point, and addition operation forms a finite cyclic group with O as its identity.

Another operation, defined on elliptic curve is multiplication of point P by integer k. This operation is defined straightforward, i.e. $4P = P + P + P + P$.

Elliptic curve group order $n = \#E_p(a, b)$ can be roughly estimated using Hasse theorem [York, 1992]:

Let $E_p(a, b)$ is a group on elliptic curve $y^2 = x^3 + ax + b$ and $t = p + 1 - \#E_p(a, b)$. Then

$$|t| \leq 2\sqrt{p}. \quad (3)$$

Equation (3) can be rewritten in more comfortable form:

$$p + 1 - 2\sqrt{p} \leq \#E_p(a, b) \leq p + 1 + 2\sqrt{p}.$$

Since elliptic curve group is cyclic with order n, fixed point P multiplication by any integer k can be replaced with multiplication by number $\tilde{k} \in Z_n$, where $\tilde{k} = k \bmod n$ and $0P = O$, i.e. any point multiplied by zero is an infinity point.

Key agreement protocol (KAP)

Now we propose the following two parties key agreement protocol.

1. Parties agree on publicly available matrix Q over elliptic curve $E_p(a, b)$ and matrices L, R over Z_n.

2. Alice randomly generates two secret sequences $\{x_i\}$, $\{y_i\}$, $i = 0, 1, \ldots, k$ in Z_n and computes

$$X = \sum_{i=0}^{k} x_i L^i = x_0 I + x_1 L + \ldots + x_k L^k,$$

$$Y = \sum_{i=0}^{k} y_i R^i = y_0 I + y_1 R + \ldots + y_k R^k.$$

3. Bob randomly generates two secret sequences $\{u_i\}$, $\{v_i\}$, $i = 0, 1, \ldots, k$ in Z_n and computes

$$U = \sum_{i=0}^{k} u_i L^i = u_0 I + u_1 L + \ldots + u_k L^k,$$

$$V = \sum_{i=0}^{k} v_i R^i = v_0 I + v_1 R + \ldots + v_k R^k .$$

4. Alice computes intermediate value K_A and sends result to Bob.

$$K_A = XQY \qquad (4)$$

5. Bob computes intermediate value K_B and sends result to Alice.

$$K_B = UQV \qquad (5)$$

6. Since matrices X, U and Y, V are commutative, both parties compute common secret key

$$K = XK_BY = UK_AV = XUQVY. \qquad (6)$$

Preliminary security analysis

The security parameters are matrix dimension m, elliptic curve group order n and secret sequences length k. They must be large enough to prevent brute force attack. To compromise the key K, the adversary must solve the (4), (5) matrix equations to find X, Y and U, V with known instances Q, K_A, K_B.

Let $X = \{x_{ij}\}$, $Y = \{y_{ij}\}$, $Q = \{Q_{ij}\}$, $A = \{A_{ij}\}$ are matrices of 2-nd order. Then matrix equation $XQY = K_A = A$ can be rewritten as system of bilinear equation over elliptic curve group:

$$\begin{cases} x_{11}y_{11}Q_{11} + x_{11}y_{21}Q_{12} + x_{12}y_{11}Q_{21} + x_{12}y_{21}Q_{22} = A_{11} \\ x_{11}y_{12}Q_{11} + x_{11}y_{22}Q_{12} + x_{12}y_{12}Q_{21} + x_{12}y_{22}Q_{22} = A_{12} \\ x_{21}y_{11}Q_{11} + x_{21}y_{21}Q_{12} + x_{22}y_{11}Q_{21} + x_{22}y_{21}Q_{22} = A_{21} \\ x_{21}y_{12}Q_{11} + x_{21}y_{22}Q_{12} + x_{22}y_{12}Q_{21} + x_{22}y_{22}Q_{22} = A_{22} \end{cases} \qquad (7)$$

We do not know the actual complexity of such systems. It is known that solution of a system of polynomial equations over any field is NP-Complete [Garey, Jonson, 1979]. But in this case the obtained system is not over the field. This system can be interpreted also as a system of equations in vector space of elliptic curve points over Z_n. Thus, we can make a conjecture that solving a system of bilinear equations over elliptic curve points vector space is not easier than solving a system of bilinear polynomial equations over any field.

We can also refer to Schaefer Dixotomy theorem for a constraint satisfiability problem denoted by SAT(S) [Schaefer, 1978]. In general, the complexity of any computational problem can be estimated by reformulating this problem into the decisional problem and reducing some known NP-Complete problem into this decisional problem. Without proof we assert that there is a SAT(S) problem reducible in polynomial time to the decisional problem corresponding to (4), (5).

On the other hand, notice that proposed KAP is a generalized elliptic curve Diffie-Hellman KAP (ECDH). Indeed, if we set matrix dimension to $m = 1$ and secret sequence length to $k = 1$, we get algorithm similar to ECDH.

Further investigations are required to select the values of security parameters and estimate the security level.

Bibliography

[Anshel et al, 1999] Anshel I., Anshel M., Goldfeld D. An algebraic method for public key cryptography. Mathematical Research Letters 6, pp. 1–5, 1999.

[Diffie, Hellman, 1976] Diffe W., Hellman M.. New Directions in Cryptography. In IEEE Transaction on Information Theory, IT-22 (6, 644-654), 1976.

[Garey, Jonson, 1979] Garey M. R., Johnson D. S. Computers and Intractability: A Guide to the Theory of NP-Completeness, W. H. Freeman and Company, 1979.

[Ko et al, 2000] Ko K. H., Lee S. J., Cheon J. H., Han J. W., Kang J. S., Park C. New Public key Cryptosystem Using Braid Groups. Advances in Cryptology, Proc. Crypto 2000, LNCS 1880, Springer–Verlag, pp. 166–183, 2000.

[Kurt, 2006] Kurt Y. A New Key Exchange Primitive Based on the Triple Decomposition Problem. Available at: http://eprint.iacr.org/2006/377, 2006

[Sakalauskas, 2005] Sakalauskas E., One Digital Signature Scheme in Semimodule over Semiring, *Informatica*. ISSN: 0868-4952, vol. 16, no. 3(2005), pp. 383-394.

[Sakalauskas et al, 2007] Sakalauskas E., Tvarijonas P., Raulinaitis A., *Key Agreement Protocol (KAP) Using Conjugacy and Discrete Logarithm Problems in Group Representation Level,* Informatica, Vol. 18, No. 1, 2007, pp. 115-124.

[Sidelnikov et al, 1993] Sidelnikov V., Cherepnev M., Yaschenko V. Systems of open distribution of keys on the basis of non-commutative semigroups. Russian Acad. Sci. Dokl. Math., 48(2), pp. 566-567, 1993.

[Schaefer, 1978] Schaefer T.J., The Complexity of Satisfiability Problems. In Proceedings 10th ACM Symposium on Theory of Computing, 216-226, 1978.

[Shpilrain, Ushakov, 2004] Shpilrain V., Ushakov A., The conjugacy search problem in public key cryptography: unnecessary and insufficient, Available at: http://eprint.iacr.org/2004/321, 2004.

[Shpilrain, Zapata, 2006] Shpilrain V., Zapata G. Using the subgroup membership search problem in public key cryptography. Contemp. Math., Amer. Math. Soc. 418 (2006), 169–179

[Smart, 2002] Smart N. P. An Identity Based Authenticated Key Agreement Protocol Based on the Weil Pairing. Electronics Letters, 38 (13). ISSN 00135194, pp. 630–636. 2002.

[York, 1992] York E. Elliptic Curves Over Finite Fields. Aviable at: http://www.math.rochester.edu/people/grads/jdreibel/ref/yorkECC.pdf, 1992.

Authors' Information

Artūras Katvickis *– PhD student, Department of Applied Mathematics, Kaunas University of Technology, Studentu str. 50-324a, Kaunas, LT-51368, Lithuania, e-mail arturas.katvickis@ktu.lt*

Paulius Vitkus *– M.Sc. student, Department of Applied Mathematics, Kaunas University of Technology, Studentu str. 50-324a, Kaunas, LT-51368, Lithuania, e-mail paulius.vitkus@ktu.lt*

ASYMMETRIC CIPHER PROTOCOL USING DECOMPOSITION PROBLEM

Andrius Raulynaitis, Saulius Japertas

***Abstract**: The asymmetric cipher protocol based on decomposition problem in matrix semiring M over semiring of natural numbers N is presented. The security parameters are defined and preliminary security analysis is presented.*

***Keywords**: asymmetric cipher, decomposition problem.*

***ACM Classification Keywords**: E.3 Data Encryption, F.2.1 Numerical Algorithms and Problems.*

***Conference**: The paper is selected from Sixth International Conference on Information Research and Applications – i.Tech 2008, Varna, Bulgaria, June-July 2008*

Introduction

Main object of asymmetric cipher constructing is one way function, which must be based on hard mathematical problems. For example traditional cryptosystems is based either the problem of factoring large integer number or on the discrete logarithm problem (DLP).

New ideas in public key cryptography using hard problems in infinite non-commutative groups and semigroups appeared in [Sidelnikov et. al., 1993]. The realization of these ideas appeared in [Ko et al., 2000], using the braid group as a platform. The security of this cryptosystem was based on conjugator search problem. But according [Shpilrain and Ushakov, 2004] the approach is not sufficient and necessary.

The other approach to use non-commutative infinite group (e.g. braid group) representation was also used for the other kind of one way functions construction as a background of both digital signature scheme and key agreement protocol [Sakalauskas, 2005], [Sakalauskas et al., 2007]. The (semi)group representation level allows us to avoid the significant problem to hide the factors in the publicly available group (braid group) word when using its presentation level. The hiding factors in representation level are achieved in a very natural way. However, the original hard problems, such as conjugator search or decomposition problems in (semi)group presentation level, are considerably weakened when transferred to the representation level. Hence these problems must be considerably strengthened by simultaneously adding the other additional hard problems in representation level.

Lately the idea to use matrix group conjugacy problem together with matrix discrete logarithm problem for the one way function construction is presented in [Sakalauskas et al., 2007]. Another approach is based on so called matrix power operation for a matrix power S-box construction, is introduced in [Sakalauskas and Luksys, 2007].

In this study we propose new asymmetric cipher using decomposition (double coset) problem in matrix semiring $\mathcal{M}$ over semiring $\mathcal{N}$ of natural numbers.

Preliminaries

We consider an infinite multiplicative matrix semiring $\mathcal{M}$ over the semiring at natural numbers $\mathcal{N}$. We assume $\mathcal{N}$ = {0, 1, 2, …}. The elements of $\mathcal{M}$ are *m*-dimensional square matrices with entries in $\mathcal{N}$. Let us choose two distinct matrices M_L and M_R in $\mathcal{M}$ and define the set of all possible polynomials P= $\{p_i( )\}$ over $\mathcal{N}$. Then the set $\mathcal{P}_L$ we define as a set of all matrices of all polynomial functions in P with argument M_L and $\mathcal{P}_R$ as a set of all polynomials functions with arguments M_R. In other words $\mathcal{P}_L$ =$\{p_i(M_L)\}$ and $\mathcal{P}_R$ = $\{p_i(M_R)\}$. It is evident, that all matrices in $\mathcal{P}_L$ and all matrices in $\mathcal{P}_R$ are commuting.

To choose, for example, some matrices X, U in $\mathcal{P}_L$ and Y, V in $\mathcal{P}_R$ we can select two pairs of polynomials p_X, p_U and p_Y, p_V in $\mathcal{P}$ and using the addition and multiplication operations in $\mathcal{N}$ find the following matrices:

$$X = p_X(M_L), Y = p_Y(M_R) \tag{1}$$

$$U = p_U(M_L), V = p_V(M_R) \tag{2}$$

As we can see, the matrices X, U and Y, V are commuting, i.e.:

$$XU = UX; YV = VY \tag{3}$$

Asymmetric cipher

On the bases of presented above formalism we can construct an asymmetric ciphering algorithm. Let's choose distinct matrices M_{L1} and M_{L2} from $\mathcal{P}_L$ and M_{R1} and M_{R2} from $\mathcal{P}_R$ to calculate polynomial matrices X and Y by (2.1) in the following way:

$$X = p_{X1}(M_{L1}) \cdot p_{X2}(M_{L2}) \tag{4}$$

$$Y = p_{Y1}(M_{R1}) \cdot p_{Y2}(M_{R2}) \tag{5}$$

$$U = p_{U1}(M_{L1}) \cdot p_{U2}(M_{L2}) \tag{6}$$

$$V = p_{V1}(M_{R1}) \cdot p_{V2}(M_{R2}) \tag{7}$$

where all polynomials are in P.

All polynomials in (4), (5) are represented by the following vectors $a_L = (a_1, a_2, \ldots, a_n)$, $b_L = (b_1, b_2, \ldots, b_n)$, $a_R = (c_1, c_2, \ldots, c_n)$, $b_R = (d_1, d_2, \ldots, d_n)$ with components in $\mathcal{N}$. Let the matrices M_{R1}, M_{L1}, M_{R2}, and M_{L2} are at the form:

$$M_{L1} = \begin{pmatrix} L_1 & \Theta \\ \Theta & h_1 I \end{pmatrix}, M_{L2} = \begin{pmatrix} h_2 I & \Theta \\ \Theta & L_2 \end{pmatrix}, M_{R1} = \begin{pmatrix} R_1 & \Theta \\ \Theta & r_1 I \end{pmatrix}, M_{R2} = \begin{pmatrix} r_2 I & \Theta \\ \Theta & R_2 \end{pmatrix} \tag{8}$$

where Θ are $m/2$-dimensional zero matrix, L_1, L_2, R_1 and R_2 are $m/2$-dimensional square matrix over $\mathcal{N}$, I is $m/2$-dimensional identity matrix, h_1, h_2, r_1 and r_2 are numbers in $\mathcal{N}$. Let's choose any matrix Q in $\mathcal{M}$ not equal M_{L1}, M_{L2} and M_{R1}, M_{R2} and calculate matrix, using the matrices X and Y calculated by (4) and (5)

$$A = XQY \tag{9}$$

Assymmetric cipher public parameters we declare $\mathcal{M}$, $\mathcal{R}$ and matrices $M_{L1}, M_{L2}, M_{R1}, M_{R2}$. The private key is $\mathrm{PrK} = \{X, Y\}$ and public key $\mathrm{PuK} = \{Q, A\}$. When vectors a_L, a_R, b_L, b_R are unknown, matrices X and Y are also unknown. Using (2) and PuK we define encryptor and decryptor operators.

Definition 1: Encryptor ε is an element in $\mathcal{M}$ which is calculated by following equation:

$$\varepsilon = UAV \tag{10}$$

Definition 2: Decryptor δ is an element in $\mathcal{M}$ satisfying following equation:

$$\delta = UQV \tag{11}$$

It is clear that the elements of $\mathcal{N}$ can be transformed in the binary form.

Definition 3: The bitwise XOR operation $\oplus$ of the elements (numbers) in $\mathcal{N}$ is a sum modulo 2 of bits of numbers presented in binary form.

Let Alice intents to encrypt her message *t* with Bob's public key $\mathrm{PuK_B} = \{Q, A\}$ obtaining a ciphertext *C*. Then Bob decrypts received *C* using his private key $\mathrm{PrK_B} = \{X, Y\}$. For the ciphering message *t* Alice must perform encoding *t* by the numbers in $\mathcal{N}$ and to form a *m*-dimension matrix *T*, corresponding *t*.

Then the encryption algorithm is following:

Step 1. Alice takes $M_{L1}, M_{L2}, M_{R1}, M_{R2}$ matrices, chooses at random vectors of polynomials coefficients a_L, a_R, b_L and b_R and using (6), (7) calculates matrices *U* and *V*.

Step 2. Alice calculates encryptor ε using (10).

Step 3. Alice calculates decryptor δ using (11).

Step 4. Alice obtains the cyphertext *C* computed by the formula:

$$C = \varepsilon \oplus T = UAV \oplus T \tag{12}$$

Step 5. Alice sends to Bob the following data $D = (C, \delta)$.

Decryption algorithm:

Bob gets data $D = (C, \delta)$ and using his private key $\mathrm{PrK_B}$ calculates the decoded plaintext *T*:

$$X\delta Y \oplus C = T \tag{13}$$

The last equation is valid since using (3) the following identities take place:

$$X\delta Y \oplus C = X(UQV)Y \oplus C = X(UQV)Y \oplus UAV \oplus T = XUQVY \oplus UXQYV \oplus T = T \tag{14}$$

4. Security analysis

To break this asymmetric cipher, Bob's $\mathrm{PrK_B}$ must be compromised, i. e. to find any *X'* and *Y'*, satisfying (9) and commutativity conditions (3). Hence for compromising $\mathrm{PrK_B}$ it is not required to find the true values of *X* and *Y*. The required matrices *X'* and *Y'*, must satisfy equation:

$$X'QY' = A \tag{15}$$

It is easy to notice, if (15) is satisfied, then

$$X'\delta Y' \oplus C = X'(UQV)Y' \oplus C = U(X'QY')V \oplus UAV \oplus T = UAV \oplus UAV \oplus T = T \tag{16}$$

Definition 4. The computational decomposition (or double coset) problem (DP) in $\mathcal{M}$ is to find any matrices *X'* and *Y'* in $\mathcal{M}$ when given *A* and *Q* satisfying equation (15).

Definition 5. The decisional (YES/NO) DP is to get an answer, if there are there any matrices *X'* and *Y'* in $\mathcal{M}$ satisfying (15) for given *Q* and *A*.

Definition 6. The DP is strong one way function (OWF) if determination of any *X'* and *Y'* is infeasible when given *A* and *Q*.

On the complexity of formulated computational DP relies on security of proposed cipher algorithm. So formulated DP is equivalent to task find any coefficients of polynomials p_{X1}, p_{X2} and p_{Y1}, p_{Y2} in (4) and (5) when the matrices *X'* and *Y'* computed using these equations satisfies (15). Let

$$\begin{aligned} X' &= p'_{X1}(M_{L1}) \cdot p'_{X2}(M_{L2}) = \\ &= \left(a'_0 M^0_{L1} + a'_1 M^1_{L1} + \ldots + a'_n M^n_{L1}\right) \cdot \left(b'_0 M^0_{L2} + b'_1 M^1_{L2} + \ldots + b'_n M^n_{L2}\right) \end{aligned} \tag{17}$$

$$\begin{aligned} Y' &= p'_{Y1}(M_{R1}) \cdot p'_{Y2}(M_{R2}) = \\ &= \left(c'_0 M^0_{R1} + c'_1 M^1_{R1} + \ldots + c'_n M^n_{R1}\right) \cdot \left(d'_0 M^0_{R2} + d'_1 M^1_{R2} + \ldots + d'_n M^n_{R2}\right) \end{aligned} \tag{18}$$

Then the DP according the Definition 4 is equivalent to find any vectors $a'_L = (a'_0, a'_1, \ldots, a'_n)$, $a'_R = (b'_1, b'_2, \ldots, b'_n)$, $b'_L = (c'_0, c'_1, \ldots, c'_n)$ and $b'_R = (d'_0, d'_1, \ldots, d'_n)$, satisfying (15), when *X'* and *Y'* are computed using ((17) and (18).

The set of possible values of vectors a'_L, a'_R, b'_L and b'_R must be large enough to prevent the total scan (i.e. *brutal force* attack), to find solution. If this is done the other way is to try to solve the matrix equation (15), using some more advanced algorithm. We can write (17) (18) in following way:

$$X' = \sum_{i,j} \left(a'_i b'_j M^i_{L1} M^j_{L2} \right) \tag{19}$$

$$Y' = \sum_{k,l} \left(c'_k d'_l M^k_{R1} M^l_{R2} \right) \tag{20}$$

Then (15) can be rewritten as:

$$X'QY' = \sum_{i,j,k,l} \left(a'_i b'_j c'_k d'_l M^i_{L1} M^j_{L2} Q M^k_{R1} M^l_{R2} \right) \tag{21}$$

This matrix equation corresponds to the *m*x*m* system of polynomial equation with fourth order monomials $a'_i b'_j c'_k d'$. But nevertheless this system allows a direct linearization. To linearize this system, let us introduce a set of new variables $\{z_{ijkl}\}$, when $z_{ijkl} = a'_i b'_j c'_k d'_l$, then (21) can be rewritten in the form:

$$\sum_{i,j,k,l} \left(z_{ijkl} M^i_{L1} M^j_{L2} Q M^k_{R1} M^l_{R2} \right) = A \tag{22}$$

As we see there are m^2 equations and $(n+1)^4$ unknowns in every equation. Depending on m^2 and $(n+1)^4$ ratio, this system is:

a) Under defined, when $m^2 > (n+1)^4$;

b) Equal defined, when $m^2 = (n+1)^4$;

c) Over defined, when $m^2 < (n+1)^4$;

We conjecture that greatest computational complexity of (22) can be achieved when the cases a) and b) are near the equal defines case. We do not know the algorithmically affective methods, how to find z_{ijkl} in semiring $\mathcal{N}$ of natural numbers Hence we can make a conjecture that private key computed by (9) represents the one-way function.

In a natural way we can choose the following security parameters for our cipher:

- dimension of matrices *m*;
- maximum order of matrices' $(M_{L1}, M_{L2}, M_{R1}, M_{R2}, Q)$ elements *r*;
- maximum order of polynomials *n*;
- maximum order of polynomials' coefficients *s*;

We need to define optimal limits of these parameters to prevent the brute force attack, qualitatively estimate the security of the cipher and minimize needs of computer's memory for matrix storage. The total scan to find a coefficients of the polynomials requires to perform the number of verification operations η:

$$\eta = s^{4n+4} \tag{23}$$

The number of bits β required to store the matrix *A* is:

$$\beta = m^2 \log_2 \left(\left(m/2 \right)^{2n} r^{4n+1} s^4 \right) \tag{24}$$

For example, consider such case: let $n = 2$, $s = 2^8$, $r = 2^4$, $m = 8$. Then $\eta = \left(2^8\right)^{4\cdot 2+4} = 2^{96}$ and the number of bits representing matrix A is $\beta = 8^2 \log_2\left((8/2)^{2\cdot 2} \cdot \left(2^4\right)^{4\cdot 2+1}\left(2^8\right)^4\right) = 64 \cdot 76 = 4864$ bits.

It is clear that under these parameters we prevent the brute force attack. In this case we have 64 equations and 81 monoms corresponding to (22). Hence our system is under defined. If we use linearization method to compromise cipher, we should freely choose 17 monoms values and then we need to solve system of 64 equations over semiring of natural number $\mathcal{N}$. We reckon this problem is hard enough to compromise a private key. Even if suitable variables z_{ijkl} will be found the problem of restoring the coefficients of polynomials remains hard.

Conlusions

In this paper we proposed one asymmetric cipher protocol using decomposition problem in matrix semiring $\mathcal{M}$ over semiring of natural numbers $\mathcal{N}$. We showed that the compromisation of cipher relies on the intractability of solution of system of linear equation over the semiring $\mathcal{N}$. After that the other problem is to restore the coefficients of polynomials which we reckon to be also hard task. The complexity estimation requires further investigations in order to find the estimates of security parameters and their relation to the other security parameters of known cryptographic primitives.

Bibliography

[Sidelnikov et. al., 1993] V. Sidelnikov, M. Cherepnev and V. Yaschenko (1993). Systems of open distribution of keys on the basis of noncommutative semigroups. // Russian Acad. Sci. Dokl. Math., 48(2), 566567.

[Ko et al., 2000] Ki Hyoung Ko , Sang Jin Lee , Jung Hee Cheon, Jae Woo Han, Ju–sung Kang, and Choonsik Park (2000). New Public–key Cryptosystem Using Braid Groups. // Advances in Cryptology, Proc. Crypto 2000, LNCS 1880, Springer–Verlag (2000), 166–183

[Shpilrain and Ushakov, 2004] V. Shpilrain and A. Ushakov (2004). The conjugacy search problem in public key cryptography: unnecessary and insufficient. // Available at: http://eprint.iacr.org/2004/321

[Sakalauskas, 2005] E. Sakalauskas, One Digital Signature Scheme in Semimodule over Semiring, Informatica. ISSN: 0868-4952, vol. 16, no. 3(2005), pp. 383-394.

[Sakalauskas et al., 2007] E. Sakalauskas, P. Tvarijonas and A. Raulynaitis, Key Agreement Protocol (KAP) Using Conjugacy and Discrete Logarithm Problems in Group Representation Level, Informatica, Vol. 18, No. 1, 2007, pp. 115-124.

[Sakalauskas and Luksys, 2007] E. Sakalauskas and K. Luksys, Matrix Power S-Box Construction, Cryptology . ePrint Archive: Report, no. 214 (2007), http://eprint.iacr.org/2007/214.

Author's Information

Andrius Raulynaitis *– PhD student in Institute of Defense Technologies of Kaunas University of Technology, Kęstučio g. 27, LT-44312 Kaunas, Lithuania, e-mail: Andrius.Raulynaitis@stud.ktu.lt*

Saulius Japertas *– associated professor in Department of Telecommunications, Kaunas University of Technology, Studentų str. 50, LT-51368 Kaunas, Lithuania, e-mail: Saulius.Japertas@ktu.lt*

IMPROVED CRYPTOANALYSIS OF THE SELF-SHRINKING P-ADIC CRYPTOGRAPHIC GENERATOR

Borislav Stoyanov

***Abstract**: The Self-shrinking p-adic cryptographic generator (SSPCG) is a fast software stream cipher. Improved cryptoanalysis of the SSPCG is introduced. This cryptoanalysis makes more precise the length of the period of the generator. The linear complexity and the cryptography resistance against most recently used attacks are invesigated. Then we discuss how such attacks can be avoided. The results show that the sequence generated by a SSPCG has a large period, large linear complexity and is stable against the cryptographic attacks. This gives the reason to consider the SSPSG as suitable for critical cryptographic applications in stream cipher encryption algorithms.*

***Keywords**: Cryptoanalys, FCSRs, Encryption Algorithm, Stream Cipher, Self-Shrinking p-adic Cryptographic Generator.*

***ACM Classification Keywords**: G.3 [Probability and Statistics]: Random Number Generation; E.3 [Data Encryption]; F.2.2 [Nonnumerical Algorithms and Problems]: Computations on discrete structures*

***Conference**: The paper is selected from Sixth International Conference on Information Research and Applications – i.Tech 2008, Varna, Bulgaria, June-July 2008*

Introduction

Stream ciphers have several properties that make them suitable for use in telecommunication applications. But apart from the security tried to obtain, the main property that makes stream ciphers distinguishable from block ciphers is that they are in general fast and have low hardware complexity. Stream ciphers process the plaintext character by character, so no buffering is required to accumulate a full plaintext block.

Most stream ciphers are based on simple devices that are easy to implement and run efficiently. A common example of such a device is the linear feedback shift register (LFSR). Such simple devices produce predictable output given some previous output. This is due to the linear property of the device. Therefore, in order to use LFSRs in cryptographical primitive, and particularly in a stream cipher, the linearity must be destroyed [Yilmaz, 2004]. Unfortunately, the classical fast and cheap LFSRs are vulnerable to the so-named "Berlekamp-Massey crypto attack" [Lidl, Niederreiter, 1983], [Oorshot, Menezes, Vanstone, 1997], [Schneier, 1996].

Having in mind the advantages of the stream ciphers with simple structure, recently some theorists have used new approach of stream cipher design and have proposed a few new architectures named Shrinking generator [Coppersmith, Krawczyk, Mansour, 1994] and Self-Shrinking generator [Meier, Staffelbach, 1998]. With regard to positive features of the two generators and Feedback with Carry Shift Registers (FCSRs) [Klapper, Goresky, 1994], [Xu, 2000] the SSPCG [Tasheva, 2005], [Tasheva, Bedzhev, 2005], [Tasheva, Bedzhev, Stoyanov, 2005] have created. The results show that the SSPCG is a promising candidate for high-speed encryption applications due to its simplicity and provable properties.

In this paper, the SSPCG is further investigated with main focus on the period, linear complexity and resistance against the recently used cryptographic attacks. The improved cryptoanalysis shows SSPCG suitable for critical cryptographic applications.

Description of the Self-shrinking p-adic cryptographic generator

In contrast with the classic Self-Shrinking generator the SSPCG architecture (Fig. 1) uses a p-adic FCSR instead of LFSR. This allows the generator to produce a number in the range 0 to p−1 (p-its) in one step (a_i = [0, 1, …,

p−1]). TheSSPCG selects a portion of the output p-adic FCSR sequence by controlling of the p-adic FCSR itself using the following algorithm:

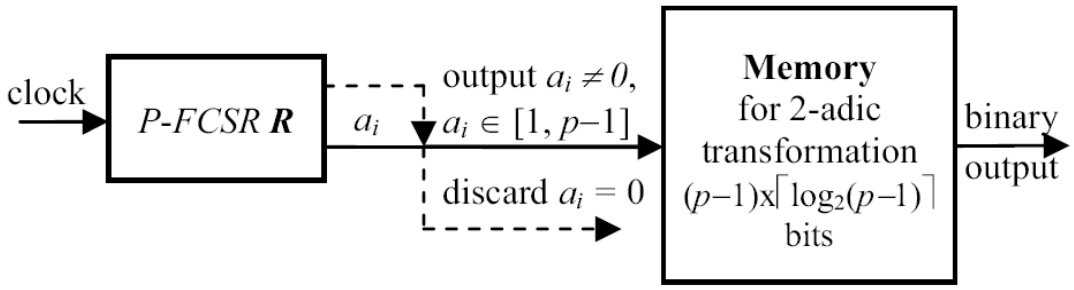


Fig. 1 Self-shrinking p-adic cryptographic generator

Definition 1: The algorithm of the Self-Shrinking p-adic Generator (Fig. 1) consists of the following steps:

1. The p-adic FCSR R is clocked with clock sequence with period τ_0.

2. If the p-adic FCSR output number is not equal to 0 ($a_i \neq 0$), the output bit forms a part of the p-adic SSPCG sequence. Otherwise, if the output number of the p-adic FCSR is equal to 0 (ai = 0), the p-adic output number of SSPG is discarded.

3. The shrunken p-adic SSPG output sequence is transformed in 2-adic sequence in which every p-adic number is presented with $\lceil \log_2(p-1) \rceil$ binary digits, where $\lceil x \rceil$ is the smallest integer which is greater or equal to x. Every output number i from 1 to p−1 of p-adic SSPCG sequence is depicted with p−adic expansion of the number:

$$i-1+\frac{2^{\lceil \log_2(p-1) \rceil}-(p-1)}{2} \qquad (1)$$

The SSPCG uses the generalization of 2-adic FCSRs with stage contents and feedback coefficients in Z/(p) where p is a prime number, not necessarily 2.

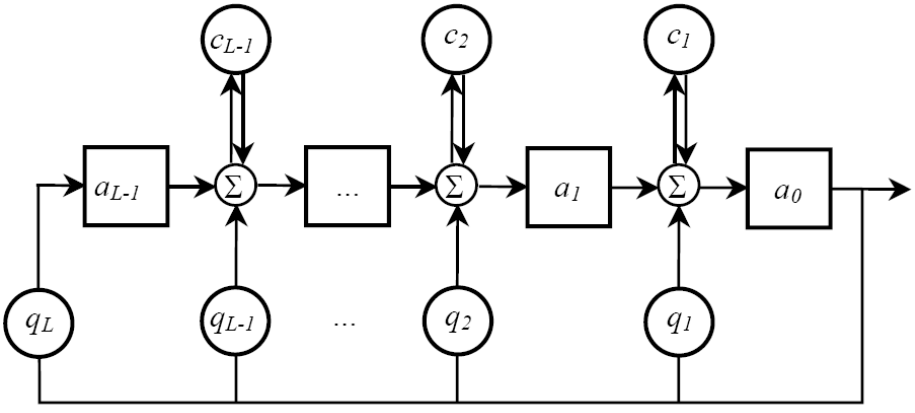


Fig. 2 Galois FCSR

Definition 2: A p-adic feedback with carry shift register with Galois architecture of length L (Fig. 2) consists of L stages (or delay elements) numbered 0, 1, …, L-1, each capable to store one p-adic (0, 1, …, p-1) number and having one input and one output; and a clock which controls the movement of data. During each clock cycle the following operations are performed:

1. The content of stage 0 is output and forms part of the output sequence;

2. The sum modulo p after stage i is passed to stage i - 1 for each i, $1 \le i \le L-1$;

3. The output of the last stage 0 is introduced into each of the tapped cells simultaneously, where it is fully added (with carry) to the contents of the preceding stages.

The $q_1, q_2, \ldots, q_L$ are the feedback multipliers and the cells denoted with $c_1, c_2, \ldots, c_{L-1}$ are the memory (or carry) bits. If $q = -1+q_1p + q_2p^2 +\ldots+ q_Lp^L$ is the base p expansion of a positive integer $q \equiv -1 \pmod p$, then q is a connection integer for a FCSR with feedback coefficients $q_1, q_2, \ldots, q_L$ in Z/(p).

With each clock cycle, the integer sums $\sigma_j = a_j + a_0 q_j + c_j$ is accumulated. At the next clock cycle this sum modulo p $a'_{j-1} = \sigma_n (\mathrm{mod}\, p)$ is passed on the next stage in the register, and the new memory values are $c'_{j-1} = \sigma_n (div\ p)$.

The nonlinearity of the proposed SSPG follows from the fact that it is unknown at which positions the FCSR sequence is shrunken. As a result the linear algebraic structure of the original FCSR sequence is destroyed. The software SSPG implementation is very fast because the pseudorandom generator produces $\lceil \log_2 (p-1) \rceil$ binary digits in every step.

It is proved that the period of the SSPCG realized by maximum length p-adic FCSR of length L and connection integer q is $S_0 = T_0^* \lceil \log_2 (p-1) \rceil$, where T_0^* is the number of output p-adic FCSR numbers different from 0.

The self-shrunken output SSPCG sequence generated by maximum length p-adic FCSR of length L and connection integer q is a balanced sequence.

The results from statistical analysis show that the sequence generated by SSPG is uniform, scalable, uncompressible and unpredictable.

Improved cryptoanalysis of the Self-shrinking p-adic cryptographic generator

In this section novel results concerning period, linear complexity, and cryptoresistance of SSPCG sequences are presented.

First, we will remind that the period of p-adic FCSR is $T_0 = 2d$, where the connection integer $q = 2d + 1$ [Goresky, Klapper, 2002], and p_0 and q are strong p-prime numbers [Xu, 2000]. Since the output sequence of the SSPCG is balanced each different from p-its will have

$$T_0^p \approx \left\lceil \frac{T_0}{p} \right\rceil \qquad (2)$$

number of appearances in a period T_0. Due to of the fact that there is an exit only in different from 0 p-its, then T_0^* acquires the following appearance:

$$T_0^* \approx (p-1) \left\lceil \frac{T_0}{p} \right\rceil \qquad (3)$$

Then the period of the SSPCG S_0 could be found with the help of the following expression:

$$S_0 \approx (p-1) \left\lceil \frac{T_0}{p} \right\rceil \lceil \log_2 (p-1) \rceil = (p-1) \left\lceil \frac{2d}{p} \right\rceil \lceil \log_2 (p-1) \rceil \qquad (4)$$

From (4) follows that linear complexity has the following value:

$$\lambda(Z) \ge \log_2 \left((p-1) \left\lceil \frac{2d}{p} \right\rceil \lceil \log_2 (p-1) \rceil \right) \qquad (5)$$

Both the Shrinking Generator and Self-Shrinking Generator use the LFSRs and have a simple structure. Despite of this fact no successful cryptanalytic attack for both generators has been published so far.

Due to the nonuniform exit from the SSPCG it is impossible to apply the p-adic Rational Approximation attack [Xu, 2000].

All attacks acting against the Self-Shrinking pseudorandom generator [Zenner, Krause, Lucks, 2001] are unapplicable, due to the fact that they act against the generator constructed by LFSRs.

With a respect of [Zenner, Krause, Lucks, 2001] we recommend the design of the including FCSR to be upper than 256 memory cells in order to hinder the cryptographic attacks time-memory-data and Backtracking Algorithm.

Conclusion

The calculation of the period and the linear complexity of the SSPCG gives an opportunity of more successful software realization of this generator due to the greater security, due to using easily generated p-prime numbers. The impossibility for attacking the SSPCG with the familiar cryptographic attacks increases the reliability in the properties of the generator.

The SSPCG is one example of a nonlinear combining function. The results presented in previous section mostly have a straightforward extension to general nonlinear combining functions in algebraic normal form.

On the basis of the simplified design the SSPCG shows properties of a successful candiate as a main or slave pseudorandom generator in stream cipher encryption.

Bibliography

[Coppersmith, Krawczyk, Mansour, 1994] D. Coppersmith, H. Krawczyk, Y. Mansour. The Shrinking Generator. Proceedings of Crypto 93, Springer-Verlag, pp. 22-39, 1994.

[Goresky, Klapper, 2002] M. Goresky, A. Klapper. Fibonacci and Galois Representations of Feedback-With-Carry Shift Registers. IEEE Trans. Inform. Theory, vol. 48, 2002, pp. 2826−2836.

[Klapper, Goresky, 1994] A. Klapper, M. Goresky. 2-adic Shift Register. Fast Software Encryption, Second International Workshop. Lecture Notes in Computer Science, vol. 950, Springer Verlag, N. Y., 1994, pp.174-178

[Lidl, Niederreiter, 1983] R. Lidl, H. Niederreiter. Finite Fields. Addison-Weslley Publishing Company, London, England, 1983.

[Meier, Staffelbach, 1998] W. Meier, O. Staffelbach. The Self-Shrinking Generator. Proceedings of Advances in Cryptology, EuroCrypt '94, Springer-Verlag, pp. 205-214, 1998.

[Oorshot, Menezes, Vanstone, 1997] P. van Oorshot, A. Menezes, S. Vanstone. Handbook of Applied Cryptography. CRC Press, 1997.

[Schneier, 1996] B. Schneier. Applied Cryptography. John Wiley & Sons, New York, 1996

[Tasheva, 2005] Zh. Tasheva. An Algorithm for Fast Software Encryption. International Conference on Computer Systems and Technologies - CompSysTech 2005, Technical University, Varna, Bulgaria, 16-17 June 2005, pp.II.18-1-II.18-6.

[Tasheva, Bedzhev, 2005] Zh. Tasheva, B. Bedzhev. Software Implementation of p-adic Self-shrinking Generator for Aerospace Cryptographic Systems. Scientific Conference "SPACE, ECOLOGY, SAFETY" with International Participation, 10–13 June 2005, Varna, Bulgaria, pp. 439-444.

[Tasheva, Bedzhev, Stoyanov, 2005] Zh. Tasheva , B. Bedzhev, B. Stoyanov. Self-Shrinking p-adic Cryptographic Generator. XL International Scientific Conference on Information, Communication and Energy Systems and Technologies, ICEST 2005, Nic, Serbia and Montenegro, June 29-July 1, 2005, pp.7-10.

[Xu, 2000] J. Xu. Stream Cipher Analysis Based on FCSRs, PhD Dissertation, University of Kentucky, 2000, http://www.cs.engr.uky.edu/etd/theses/uky-cocs-2000-d-002/.

[Yilmaz, 2004], E. Yilmaz. Two Versions of the Stream Cipher Snow. Master Thesis, The Graduate school of natural and applied sciences of Middle east technical university, 2004, p. 60.

[Zenner, Krause, Lucks, 2001] E. Zenner, M. Krause, S. Lucks. Improved Cryptanalysis of the Self-Shrinking Generator. LNCS, Vol. 2119, 2001, pp. 21-35.

Authors' Information

Borislav Stoyanov *– Assistant Prof., PhD in the Faculty of Computer Informatics, Shumen University; http://crypt.co.nr, e-mail: bpstoyanov@yahoo.com*

МЕТОДЫ АВТОМАТИЗИРОВАННОГО ПРОЕКТИРОВАНИЯ И СОПРОВОЖДЕНИЯ ПОЛЬЗОВАТЕЛЬСКИХ ИНТЕРФЕЙСОВ

Валерия Грибова

Аннотация: *В данной работе представлены методы автоматизации проектирования и сопровождения компонента представления информации в проекте пользовательского интерфейса. Для каждого метода описана его основная идея, область использования, проведено сравнение с аналогами.*

Ключевые слова: *Онтология, проект интерфейса, автоматическая генерация*

ACM Classification Keywords*: I.2.2 Artificial Intelligence – automatic programming*

Conference*: The paper is selected from XIVth International Conference "Knowledge-Dialogue-Solution" KDS 2008, Varna, Bulgaria, June-July 2008*

Введение

Разработка интерфейсов программных средств, удовлетворяющих требованиям пользователей, является одной из важнейших задач при создании программного обеспечения. Общей тенденцией является усложнение пользовательских интерфейсов, связанное как с увеличением функциональности программ, так и с различными, часто изменяющимися условиями их эксплуатации. В результате, его проектирование, разработка, модификация и сопровождение требуют значительных затрат времени, так как необходимо учитывать множество факторов, тесно взаимосвязанных между собой и зачастую противоречащих друг другу. Так, по оценкам специалистов, например [1], 50% времени, требуемого на разработку программного средства, уходит на пользовательский интерфейс; он занимает в среднем 48% программного кода.

Для снижения трудоемкости и времени разработки пользовательских интерфейсов предложен онтологический подход к автоматизации его разработки и сопровождения. Основная идея подхода заключается в формировании проекта интерфейса на основе предлагаемых разработчику онтологий и автоматическая генерация по проекту исполнимого кода интерфейса на некоторый язык программирования и связывание его с кодом прикладной программы (язык программирования и способ взаимодействия интерфейса с прикладной программой – локальный либо распределенный определяет разработчик). В результате разработка и сопровождение пользовательского интерфейса сводятся к разработке и сопровождению его проекта.

Опыт использования инструментария, основанного на онтологическом подходе показал, что разработка и сопровождение компонента представления информации проекта интерфейса при разработке сложных и комплексных интерфейсов остается трудоемкой, так требуют от разработчика знания принципов юзабилити[1], различных стандартов разработки, учета требований пользователей, среды использования интерфейса и др. Поэтому актуальными являются исследования, направленные на разработку методов и программных средств автоматизации проектирования компонента представления информации.

[1] Концепция разработки пользовательских интерфейсов, ориентированная на максимальное психологическое и эстетическое удобство для пользователя.

Проект пользовательского интерфейса

Проект пользовательского интерфейса задает множество процессов диалога с пользователем для осуществления ввода исходных данных в прикладную программу, вывода результатов ее работы, управления прикладной программой, а также для организации контекстно-зависимой помощи пользователю в процессе его взаимодействия с программным средством [2, 3]. Проект интерфейса состоит из следующих компонентов: системы понятий диалога, задач пользователя, представления информации, связи интерфейса с прикладной программой, сценария диалога, отображения.

Проект системы понятий диалога (ПСПД) описывает систему терминов предметной области, в которых выражаются входные/выходные данные прикладной программы, информация об управлении прикладной программой и ее пользовательским интерфейсом, а также об интеллектуальной поддержке действий пользователя. Данная компонента является конкретизацией Онтологии системы понятий диалога и представляет собой пару ПСДП=(G, σ), где G – ориентированный граф ПСПД, σ - разметка графа. Ориентированный граф ПСПД G=<Vertexes, Arcs, RootVertex>, где Vertexes - множество вершин графа, Arcs – множество дуг графа, RootVertex - корневая вершина. Множество вершин графа Vertexes=$\{Vertex_i\}_{i=1}^{vertexescount}$ состоит из двух подмножеств – множества терминальных и нетерминальных вершин, Vertexi $\in$Neterminal_Vertexes $\cup$ Terminal_Vertexes, где Neterminal_Vertexes - множество нетерминальных вершин, Terminal_Vertexes - множество терминальных вершин.

Множество дуг графа Arcs = $\{Arc_i\}_{i=0}^{arcsount}$, Arc_i=<$VertexFrom_i$, $VertexTo_i$ >, где $VertexFrom_i \in$ Vertexes, $VertexTo_i \in$ Vertexes. Корневой вершиной графа RootVertex является нетерминальная вершина, RootVertex$\in$Neterminal_Vertexes. Разметка графа σ - отображение множества вершин и дуг графа во множество имен Ω, где Ω = ИмяСистемыПонятий $\cup\{ИмяГруппы_i\}_{i=1}^{termgroupcount}$ $\cup$ $\{ИмяТермина_j\}_{j=0}^{termcount}$ $\cup$ $\{ИмяАтрибута_i\}_{i=1}^{attributecount}$ $\cup$ {Качественное значение$_i$}$_{i=2}^{valuecount}$ $\cup\{((ВещественноеМин, ВещественноеМакс), ВещественноеМера)_i\}_{i=0}^{floatcount}$ $\cup\{((ЦелоеМин, ЦелоеМакс), ЦелоеМера)\}_{i=0}^{integercount}$ $\cup\{Ni\}_{i=0}^{stringscount}$ $\cup${Группа Терминов, Термин, Атрибут, Совместный, Несовместный, ВещественноеЗначение, ЦелоеЗначение, СтроковоеЗначение}. Множество имен Ω определяется компонентами онтологии системы понятий диалога, при этом разметка графа определяется типом вершин $VertexFrom_i$ и $VertexTo_i$.

Проект задач пользователя (ПЗП). Любое программное средство предназначено для решения задач пользователя. Задачи могут быть разбиты на подзадачи, представляющие собой шаги для решения исходной задачи. Между подзадачами существуют отношения [4], которые определяют порядок и условия их выполнения. ПЗП=(T, σ), где T – дерево задач, σ - разметка дерева. Дерево задач T=<Vertexes, Arcs, RootVertex>. Разметка дерева σ - это отображение множества вершин дерева во множество имен Ω, где Ω =$\{ИмяЗадачи_i\}_{i=1}^{taskcount}$ $\cup$ ТипМножества. Множество имен Ω состоит из имен задач и имен множеств ТипМножества ={«выбор», «объединение», «разрешение», «деактивация»}. Разметка σ задается следующим образом: RootVertex$\rightarrow$ ИмяОбщейЗадачи, корневая вершина отображается в имя общей задачи. Каждая вершина $Vertex_i \rightarrow (Mark1_i, Mark2_i)$, где $Mark1_i \in \{ИмяЗадачи_i\}_{i=1}^{taskcount}$, $Mark2_i \in$ТипМножества, (семантика множеств определена в [4]).

Проект представления информации описывает структуру и свойства визуального представления элементов WIMP[1]-интерфейсов и является конкретизацией модели онтологии WIMP-интерфейсов. Данный проект характеризуется множеством окон Windows, Windows=$\{Window_i\}_{i=1}^{windowcount}$, таких что

[1] Windows, Icons, Menus, Pointing devices

$Window_i \in Controls$ | Controltype = Окно-контейнер. Согласно модели онтологии WIMP-интерфейсов, каждый элемент интерфейса $Control_i$ задается своим типом, множеством параметров, функций и событий. Тип, функции и события элемента интерфейса $Control_i$ заданы в модели онтологии, соответственно, данный проект описывает значения параметров элементов интерфейса, характеризующих конкретный проект интерфейса. Параметрами окна $Window_i$ могут быть другие элементы интерфейса, в этом случае $Param\_{Typek} = Control_m$, где $Control_m \in Controls$.

Проект связи интерфейса с прикладной программой описывает способ взаимодействия интерфейса и прикладной программы, а также программные интерфейсы, посредством которых обеспечивается связь между интерфейсом и прикладной программой. Данный проект является конкретизацией соответствующей онтологии и представляет собой множество $Interfaces=\{Interface_i\}_{i=1}^{\text{int } erfacecount}$. Каждый элемент множества содержит описание программного интерфейса, предоставляемого прикладной программой, $Interface_i = < Interfacename_i, InteractionModel_i, Functions_i$, где $Interfacename_i$– уникальное имя программного интерфейса, $InteractionModel_i$ принимает значение из множества IModels={Локальная, Распределенная}, множество функций Functions - описание программных интерфейсов, предоставляемых прикладной программой.

Проект сценария диалога состоит из описания начального окна StartWindow, с которого начинается диалог с пользователем, $StartWindow=Window_i$, где $Window_i \in Windows$, а также множества возможных состояний States ={$State_i$}. Каждое состояние Statei содержит: описание события $Event_i$, множество переменных $Variables_i$, которые необходимы для описания последовательности инструкций $Instructions_i$, последовательность инструкций $Instructions_i=<Instruction_{ij}>_{j=1}^{instructioncount}$. $\}_{i=1}^{statecount}$. Инструкциями могут быть: вызовы программных интерфейсов прикладной программы, вызовы системных функций, составные элементы (условная конструкция, цикл), содержащие последовательности из перечисленных элементов). Проект сценария диалога является конкретизацией Онтологии сценария диалога.

Проект отображения в общем случае определяет множество отображений элементов одного из компонентов проекта интерфейса в другой: элементов проектов системы понятий диалога и задач пользователя в параметры элементов интерфейса проекта представления информации, параметров элемента интерфейса на выходные данные (результаты) прикладной программы проекта сценария диалога и др.

Методы построения проекта представления

Одной из задач при формировании проекта интерфейса является формирование проекта представления информации. Данный проект формируется по проектам системы понятий диалога и задач пользователя, т.е. разработчик интерфейса задает множество отображений указанных компонентов проекта в проект представления информации. Его задача сводится к тому, чтобы каждому термину предметной области, либо группе терминов сопоставить его (их) представление в интерфейсе – множество интерфейсных элементов с заданными свойствами (см. рис. 1).

При таком отображении возникают следующие проблемы:

- трудоемкость разработки проекта представления информации, если проект системы понятий диалога либо задач пользователя имеют большой размер (большая предметная область);

- сложность сопровождения проекта интерфейса: при изменении термина (терминов) предметной области необходимо также изменять проект представления информации;

- высокие требования к разработчику: знание им требований юзабилити, стандартов разработки, их совмещение с контекстом использования интерфейса, требованиями пользователей, предметной области к представлению информации и др.

Для реализации основного требования к проекту интерфейса – обеспечению его легкой модифицируемости и сопровождения предлагается несколько методов построения проекта представления информации. Выбор метода построения при разработке проекта интерфейса определяется его разработчиком и зависит от особенностей компонентов проекта (например, размера проекта системы понятий диалога, задач пользователя, особенностей предметной области к представлению информации и др.).

Метод прямого формирования проекта представления информации. В случае прямого формирования проекта представления информации значением параметра элемента интерфейса является прямая ссылка (см. рис. 1) либо на элемент проекта системы понятий диалога, либо на элемент проекта задач пользователя, т.е. для элемента интерфейса проекта представления WIMP-интерфейсов $Control_i \in Windows$, у которого $Param\_Type_{param}=String$, $Param_Value_{param}=$Значение, где Значение$\in \{ИмяЗадачи_i\}_{i=1}^{taskcount} \cup$ ИмяСистемыПонятий $\cup \{ИмяГруппы_i\}_{i=1}^{termgroupcount} \cup \{ИмяТермина_j\}_{j=0}^{termcount} \cup \{ИмяАтрибута_i\}_{i=1}^{attributecount} \cup \{Качественное\ значение_i\}_{i=2}^{valuecount}$. Например, на рис. 1, значениями параметра «Текст элемента» элементов пролистываемого списка является ссылка на значения атрибута «локализация» из проекта системы понятий диалога. При этом количество элементов пролистываемого списка определяется числом значений атрибута «локализация»; при изменении числа значений этого атрибута, изменяется количество элементов списка; аналогично, изменения самих значений этого атрибута в проекте системы понятий диалога приводят к изменениям параметра «Текст элемента».

Данный метод формирования проекта представления удобен, если проект системы понятий диалога имеет небольшой размер, а также требуется точное указание о представлении системы понятий диалога либо задач пользователя в интерфейсе (например, в случае особых запросов пользователей).

Метод регулярного формирования проекта представления информации. В данном случае значением параметра элемента интерфейса являются ссылки не на компоненты проекта системы понятий диалога, а на компоненты соответствующей онтологии (группы терминов, термины, значения, атрибуты), т.е. для элемента интерфейса проекта представления WIMP-интерфейсов $Control_i \rightarrow Element \mid Controli \in Windows$, $Element \in \{ГруппыТерминов, Термины, Атрибут, Качественные\ значения\}\}_{i=1}^{controlcount}$.

В результате такого формирования проекта представления каждая группа терминов (терминов, значений, атрибутов), входящая в проект системы понятий диалога, представляется выбранным дизайнером элементом интерфейса, при этом значением строкового параметра(ов) этого элемента интерфейса являются элементы Проекта системы понятий диалога.

Так, например, значению параметра «Элементы»= $\{Элемент\ списка_i\}_{i=1}^{elemcount}$ элемента интерфейса «Пролистываемый список» сопоставляются все качественные несовместные значения онтологии системы понятий диалога. Это означает, что все качественные совместные значения из проекта системы понятий диалога представляются в интерфейсе одинаковыми элементами интерфейса (см. рис. 2). При этом их количество может сколь угодно большим и определяется предметной областью.

Данный метод формирования проекта представления используется, если размер проекта системы понятий диалога имеет большой размер (сотни и тысячи понятий).

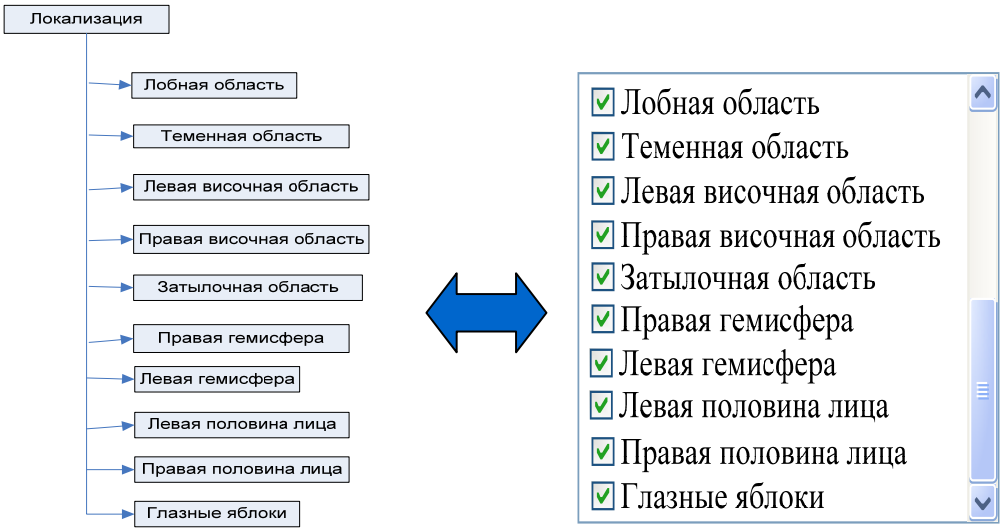


Рис. 1 Пример прямого формирования проекта представления информации

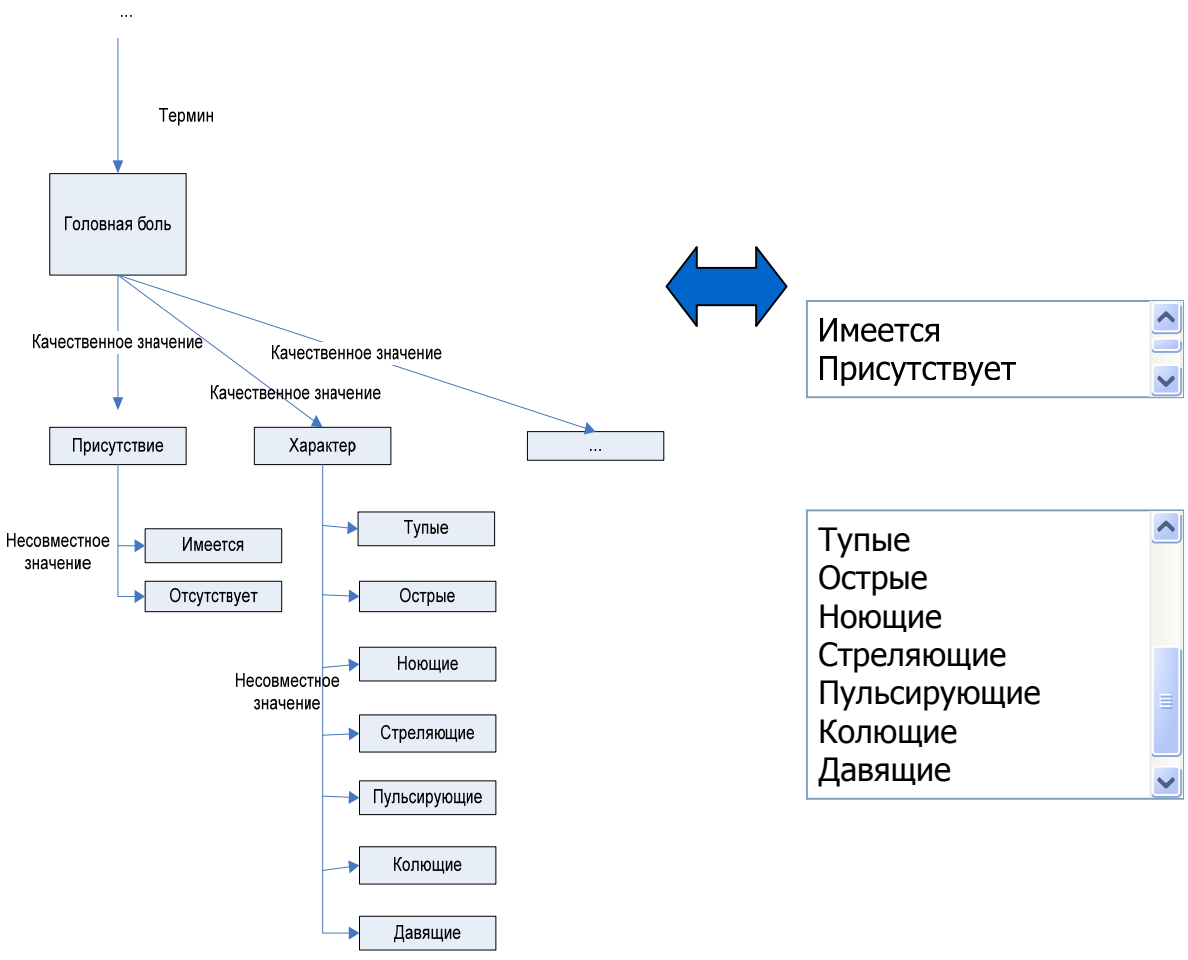


Рис. 2. Пример регулярного формирования проекта представления информации

Метод фрагментарного формирования проекта представления информации

При использовании данного метода формирования проекта представления информации, разработчик интерфейса разбивает проект системы понятий диалога на множество фрагментов, основываясь на требованиях пользователей, а также семантике и функциональности прикладной программы; каждому фрагменту проекта системы понятий диалога сопоставляются их представления в интерфейсе: $\{Fragment\_Presentastion \rightarrow Fragment\_Terms \mid Fragment\_Presentastion \in \{Control_i\}_{i=1}^{controlcount}, Control_i \in Windows, Fragment\_Terms \subset G\}_{i=1}^{fraggmentcount}$.

Основные положения метода заключаются в следующем:

- Каждому фрагменту проекта системы понятий диалога автоматически сопоставляется множество возможных представлений.
- Сопоставления осуществляются в соответствии с правилам юзабилити.
- Если для одного фрагмента возможно несколько представлений, не противоречащих правилам юзабилити, то либо:
  - право выбора единственного представления из множества допустимых предоставляется разработчику интерфейса;
  - представление автоматически выбирается в соответствии с системой умолчаний.
- Разработчику интерфейса предоставляется объяснение, содержащее протокол работы: описание набора критериев, по которым произведен выбор с указанием, какие из выбранных отображений были опровергнуты (не соответствуют правилам юзабилити и почему); какие критерии соответствуют правилам юзабилити и почему.
- Множество отображений должно расширяться (правила юзабилити и интерфейсные элементы постоянно развиваются уточняются, совершенствуются).

Для обеспечения расширяемости множества отображений разработана модель онтологии отображения представлений. В терминах онтологии описываются правила отображения элементов системы понятий диалога в их возможные представления в интерфейсе (множество интерфейсных элементов) с учетом правил юзабилити. Таким образом, формируется множество отображений (база знаний отображений). Любое отображение описывается парой – условием отображения и множеством возможных представлений. Условие отображения может состоять из множества условий, каждое из которых задает значения параметров, истинность которых проверяется в проекте системы понятий диалога. Параметры условий задаются в терминах онтологии системы понятий диалога. Например, в качестве условия может выступать количество качественных значений некоторого термина из проекта системы понятий диалога и способ их выбора (совместный либо несовместный). В зависимости от различных значений этих двух условий им сопоставляются различные представления: множество радио-кнопок, кнопок-флажков, раскрывающийся список, пролистываемый список с элементами различных типов и др. На рис. 3 представлены два различных его разбиения одного фрагмента проекта системы понятий диалога и соответствующий каждому разбиению фрагмент проекта представления информации.

Как видно из рис. 3, различные разбиения проекта системы понятий диалога приводят к различным проектам представления информации. В первом случае фрагмент имеет два разбиения, каждому из которых соответствует - окно («Боли» и «Повышение артериального давления»), во втором случае во фрагменте выделено одно разбиение, которому соответствует окно «Жалобы».

Данный метод формирования проекта представления также, как и предыдущий, удобен, если размер проекта системы понятий диалога имеет большой размер (сотни и тысячи понятий).

Рис. 3. Пример фрагментарного формирования проекта представления информации

Обсуждение результатов

В настоящее время рассмотренные выше методы реализованы и интегрированы в инструментарий для проектирования и сопровождения пользовательского интерфейса, основанный на онтологическом подходе. С использованием данных методов реализованы и сопровождаются интерфейсы для ряда программных средств.

Обсуждение полученных результатов целесообразно проводить с Моделеориентированными средствами (МОС) для разработки и автоматической генерации пользовательских интерфейсов [5-7], поскольку в Построителях интерфейсов такие методы не реализованы.

Метод прямого формирования проекта представления информации используется в некоторых МОС следующим образом: значениям параметров интерфейсных элементов присваиваются понятия предметной области либо задачи пользователя из соответствующих компонентов проекта интерфейса. Однако при их изменении приходится заново осуществлять присваивание, после чего производить перекомпиляцию интерфейса, в результате сильно возрастает трудоемкость его сопровождения. В данном методе вместо явного присваивания значений параметрам элементов интерфейса указываются ссылки на термины предметной области либо задачи пользователя, поэтому при их изменении автоматически происходят изменения и в проекте представления, перекомпиляция интерфейса при этом не требуется.

Метод регулярного формирования проекта интерфейса предложен впервые. Он позволяет значительно снизить трудоемкость разработки и сопровождения интерфейсов с большой системой понятий диалога. Например, проект системы понятий диалога для Системы интеллектуальной поддержки врача-уролога [8] состоит из 700 групп терминов и терминов и около 5000 вариантов значений. В этом случае метод, основанный на регулярном формировании проекта представления, позволяет очень быстро сформировать и автоматически, как и в предыдущем случае, сопровождать проект интерфейса при изменении системы понятий диалога.

Фрагментарные отображения используются в некоторых МОС. В этом случае разработчики определяют окно (множество окон) и информацию из модели задач либо системы понятий диалога, которая должна быть помещена в каждое окно. Далее автоматически выбираются представления для соответствующей информации. С одной стороны, такой подход уменьшает время разработки проекта представления информации, но с другой стороны, основная проблема, с которой столкнулись разработчики – сложность его сопровождения. Известно, что, во-первых, термины предметной области и задачи пользователя подвержены частым изменениям, во-вторых, любое автоматизированное представление требует последующего «ручного» пост-редактирования. В случае модифицирования терминов предметной области или задач пользователя все изменения, произведенные пост-редактированием, теряются. В результате сопровождение становится очень трудоемким. В некоторых МОС были предприняты попытки сохранять настройки пост-редактирования, однако, они не привели к положительному результату, так как не была решена основная проблема: что делать с настройками в случае появления новых терминов или задач, либо удаления существующих. В данной методе также, как и в в предыдущих, вместо явного присваивания значений параметрам элементов интерфейса указываются ссылки на термины предметной области либо задачи пользователя, при этом отображения производятся в соответствии с метриками юзабилити. Следующим недостатком данного метода, реализованного в некоторых МОС является их жесткая встроенность в инструментарий, в результате чего разработчик не знает правил, по которым проводятся автоматические преобразования, что также затрудняет и их расширение. В соответствии с методом, предложенным в данной работе, отображения явно указаны в базе отображений, которая формируется по соответствующей модели онтологии. Разработчику предоставляется возможность ее просмотра и расширения.

Благодарности

Работа выполнена при финансовой поддержке ДВО РАН в рамках Программы №15 фундаментальных исследований ОЭММПУ РАН "Проблемы анализа и синтеза интегрированных систем управления для сложных объектов, функционирующих в условиях неопределённости", проект "Синтез интеллектуальных систем управления базами знаний и базами данных при управлении сложными объектами в условиях неопределённости" и РФФИ, проект 06-07-89071-а "Исследование возможностей коллективного управления в семантическом вебе информационными ресурсами различных уровней общности".

Библиография

1. Myers Brad, Rosson Mary. Survey on user interface programming // Tech. Rpt. CMU-CS-92-113, Carnegie-Mellon, School of Comp. Sci., February 1992. http://citeseer.ist.psu.edu/myers92survey.html
2. Грибова В.В., Клещев А.С. Управление проектированием и реализацией пользовательского интерфейса на основе онтологий // Проблемы управления, 2006. №2. С.58-62.
3. Gribova V., Kleshchev A. From an ontology-oriented approach conception to user interface development International //Journal Information theories & applications. 2003. vol. 10, num.1, p. 87-94
4. Gribova V. Automatic generation of context-sensitive help using a user interface project // Proc. of XIIIth Intern. Conf. “Knowledge-dialogue-solution” – Varna, 2007. V.2. P. 417-422.
5. Puerta A. A model–based interface development environment IEEE Software, 14(1), July/August 1997. P. 41–47.
6. Puerta A.R. Issues in Automatic Generation of User Interfaces in Model-Based Systems. Computer-Aided Design of User Interfaces, ed. by Jean Vanderdonckt. Presses Universitaires de Namur, Namur, Belgium, 1996. P. 323-325.
7. Szekely P. Retrospective and Challenges for Model-Based Interface. 1996. http://citeseer.nj.nec.com/szekely96retrospective.html
8. Грибова В.В., Тарасов А.В., Черняховская М.Ю. Система интеллектуальной поддержки обследования больных, управляемая онтологией // Программные продукты и системы, 2007. №2. С. 49-51

Информация об авторе

Валерия Грибова – *к.т.н.**,** старший научный сотрудник отдела Интеллектуальных систем Института автоматики и процессов управления Дальневосточного отделения Российской академии наук, г. Владивосток, ул. Радио, 5, тел. +7 (4323) 314001, gribova@iacp.dvo.ru, http://www.iacp.dvo.ru/is.*

МЕТАМОДЕЛИРОВАНИЕ И МНОГОУРОВНЕВЫЕ МЕТАДАННЫЕ КАК ОСНОВА ТЕХНОЛОГИИ СОЗДАНИЯ АДАПТИРУЕМЫХ ИНФОРМАЦИОННЫХ СИСТЕМ

Людмила Лядова

***Аннотация**: Рассматриваются методы создания распределенных информационных систем, динамически настраиваемых на меняющиеся потребности пользователей и условия эксплуатации. Описываемые средства основаны на использовании многоуровневых моделей и метаданных, представляющих различные стороны функционирования систем на разных уровнях абстракции и с различных точек зрения. Основные уровни метаданных, описывающих систему: логический (описание объектов системы в терминах предметной области), физический (описание представления данных в базе данных) и презентационный (описание интерфейса пользователя системы). Модели и набор метаданных могут изменяться в процессе функционирования системы. На основе базовых моделей могут разрабатываться новые модели (в частности, созданы Web-модель, модели репортинга и бизнес-процессов). Представленный подход реализуется в CASE-технологии METAS, предназначенной для поддержания всего жизненного цикла адаптируемых систем. Функционирование системы строится на интерпретации построенных моделей. Возможности адаптации основаны на средствах реструктуризации данных, генерации и настройки пользовательского интерфейса, управления документами, подключения новых программных компонентов. В CASE-систему включены средства экспорта-импорта, реплицирования данных и моделей, интеграции с внешними системами, а также средства защиты. Разрабатываемые с использованием технологии информационные системы имеют клиент-серверную архитектуру. Технология METAS базируется на использовании языка UML и предметно-ориентированных языков для разработки моделей системы, описания бизнес-правил, специфических для конкретных предметных областей. Предусмотрены средства, позволяющие настраиваться на использование различных реляционных СУБД. Программная платформа – .NET.*

***Ключевые слова**: адаптируемые информационные системы, CASE-технология, модель предметной области, метамоделирование, метаданные, предметно ориентированные языки, DSL, DSM.*

***ACM Classification Keywords**: D.2 Software Engineering: D.2.2 Design Tools and Techniques – Computer-aided software engineering (CASE); D.2.11 Software Architectures – Domain-specific architectures; D.2.13 Reusable Software – Domain engineering, Reuse models.*

***Conference**: The paper is selected from Sixth International Conference on Information Research and Applications – i.Tech 2008, Varna, Bulgaria, June-July 2008*

Введение

Адаптируемость (*способность систем приспосабливаться к изменениям среды*, *окружения*) является одним из наиболее важных требований, предъявляемых к информационным системам (ИС) различного назначения. Адаптируемость рассматривают достаточно широко, включая в это понятие такие взаимосвязанные нефункциональные требования как способность к развитию, гибкость, расширяемость, интероперабельность и т.п. [1, 2]. Такая широкая природа этого понятия делает его не только интересным для исследования, но и критичным свойством в практике создания ИС, определяющим эффективность вложений в их разработку и внедрение, эксплуатацию и сопровождение, гарантирующим «живучесть» ИС.

Адаптация информационных систем – это процесс их настройки на меняющиеся условия эксплуатации и потребности пользователей и бизнес-процессов как при создании новых систем, так и при сопровождении существующих. Этот итеративный процесс можно считать важнейшей частью жизненного цикла ИС.

Адаптируемость – это характеристика, определяющая способность системы к развитию в соответствии с

нуждами пользователей и бизнеса. Различают понятия адаптивных и адаптируемых систем: *адаптируемые* системы – это «легко изменяемые» системы, включающие средства, которые обеспечивали бы их настройку на новые требования и условия динамически (в ходе эксплуатации), облегчали бы их сопровождение; *адаптивные* системы – это системы, которые меняют свое поведение автоматически в соответствии с изменениями, происходящими в их окружении («контексте»), настраиваются на изменения среды без применения каких-либо средств «ручной» настройки.

Существуют общие подходы к созданию адаптируемых ИС. Одним из таких подходов является ориентация на разработку систем, основанных на *MDA* (*Model Driven Architecture*) и технологии *DSM* (*Domain Specific Modeling*). Средства адаптации могут базироваться на использовании *динамических языков программирования*, допускающих возможность переопределения структуры программ и данных, модернизации программ за счет изменения их компонентов, языков *DSL* (*Domain-Specific Languages*), которые разрабатываются для разнообразных предметных областей [3, 4].

Максимальная гибкость при создании ИС достигается, если работа системы строится на использовании моделей, которые могут изменяться в процессе функционирования системы, управляющих ее поведением. Модели в MDA могут быть организованы на различных уровнях абстракции и платформенной независимости, что обеспечивает максимальную эффективность процесса разработки и возможность трансформации моделей [5, 6, 7].

Создание адаптируемых систем предполагает использование соответствующих инструментальных средств, поддерживающих предъявляемые к системам требования. Таким образом, можно считать, что свойство адаптируемости является необходимым не только для разрабатываемых систем, но и для применяемых при их разработке инструментальных средств.

В данной работе представлен подход к созданию адаптируемых информационных систем, основанный на построении многоуровневых моделей и использовании метаданных, представляющих информационные системы и их окружение с различных точек зрения и на различных уровнях детализации [8].

Метамоделирование и технологии создания информационных систем

Модель – это объект-«заменитель» объекта-«оригинала», который находится в определенном соответствии с оригиналом и обеспечивает представление о некоторых его свойствах. *Модель* системы представляет собой *абстрактное описание* на некотором формальном языке характеристик системы, важных с точки зрения цели моделирования, ее поведения. При создании ИС нельзя ограничиваться созданием только одной модели. Если система сложная, то учет всех ее характеристик в одной модели приведет к чрезвычайной ее сложности. Наилучший подход при разработке любой нетривиальной системы – использовать совокупность нескольких моделей, которые могут быть практически независимыми друг от друга и позволят сделать акценты на разных сторонах системы при решении различных задач поддержания ее жизненного цикла.

В общем случае модели можно разделить на следующие виды: *статические*, описывающие структурные свойства систем; *динамические*, представляющие поведенческие свойства систем; *функциональные*, описывающие функциональные свойства систем. Статическая модель описывает составные части системы, их структуру, атрибуты, взаимосвязи между ними и операции, которые они могут выполнять. Операции статической модели являются событиями динамической и функциями функциональной моделей. Динамическая модель описывает последовательность выполнения операций в процессе функционирования системы. Функциональная модель описывает преобразования, осуществляемые системой. Она раскрывает содержание операций статической модели и событий динамической.

По *степени абстракции* модели можно разделить на *концептуальные модели*, представляющие высокоуровневый взгляд на задачу в терминах предметной области; *модели спецификации*, определяющие «внешний вид» и внешнее поведение системы; *модели реализации*, которые отражают внутреннее устройство системы, конкретный способ реализации наблюдаемого поведения системы.

Существуют различные определение метамодели. Исходя из того, что модели, создаваемые при разработке ИС, должны быть описаны на каком-либо формальном языке – языке моделирования, мы будем считать, что *метамодель* – это модель языка моделирования, применяемого для формализации описания системы. *Лингвистическая метамодель* – это метамодель, которая описывает предметно-независимый язык моделирования. *Онтологическая метамодель* – это метамодель, которая описывает предметно-зависимый язык моделирования.

Четырехуровневая иерархия моделей представляет классический вариант метамоделирования при создании ИС (рис. 1). В данной иерархии каждый вышестоящий уровень определяет язык для описания нижестоящего уровня. Число уровней при реализации конкретных систем может изменяться.

Рис. 1. Классическая четырехуровневая иерархия моделей ИС

При решении задачи моделирования ИС на уровне M0 находятся *данные, описывающие состояние предметной области*, т.е. *модель состояния*. Уровень M1 является *онтологической метамоделью* для уровня M0 и *содержит модель предметной области*. Уровень M2 определяет *лингвистическую метамодель* для уровней M1 и M0: на уровне M2 находится *модель языка моделирования*, с которым работают аналитики, разработчики, CASE-средства и пр. Самый верхний уровень (M3) определяет *язык, на котором описываются метамодели* уровня M2.

В зависимости от количества уровней создаваемых моделей и способа их использования при разработке информационные системы и технологии их создания можно разделить на несколько классов.

В *традиционной информационной* системе «внутри» системы находятся данные, описывающие состояние ее предметной области. Эти данные соответствуют некоторой модели предметной области, которая может быть описана на любом языке (в том числе и на естественном). Модель разрабатывается аналитиками, после чего разработчики реализуют ее при помощи выбранных инструментальных средств создания ИС, систем программирования. В случае изменения модели приходится переписывать и перекомпилировать исходные коды приложений системы. Чаще всего при реализации ИС и в процессе ее сопровождения разработчики «уходят» от начальной модели, которая не обновляется при внесении изменений в структуры данных и алгоритмы их обработки, что приводит к различным проблемам.

В *традиционных CASE-технологиях* модель предметной области определяется формально и находится «внутри» CASE-системы. Модель описывается в терминах метамодели, которая может быть определена на любом языке и реализуется с помощью CASE-средств. Изменение модели ведет к необходимости изменения кода, сгенерированного CASE-системой. Как и в случае с традиционной информационной системой, метамодель разрабатывается аналитиками, после чего реализуется разработчиками. Изменение метамодели влечет за собой переписывание и перекомпиляцию CASE-средства, однако такие изменения происходят крайне редко. Современные CASE-средства предоставляют инструменты для создания и редактирования моделей, а также позволяет сгенерировать большую часть кода информационной системы. Полученная на выходе система обычно реализует все необходимые структуры данных, определяемые моделью, обеспечивает доступ к данным в базах данных (БД) и предоставляет стандартный интерфейс пользователя для работы с ними. Программные компоненты, реализующие специфические для конкретной системы поведенческие и функциональные аспекты, дописываются чаще всего вручную. В случае изменения модели CASE-система позволяет заново

сгенерировать код приложений ИС, при этом код, добавленный программистами «вручную», сохраняется (при соблюдении определенных правил его написания). После повторной генерации обычно требуется ручная доработка кода. Достоинством подхода является то, что существенно экономится время на начальных этапах разработки. Кроме того, поддерживается соответствие между системой и моделью.

Информационные системы, управляемые метаданными, обеспечивают более мощные возможности для динамической адаптации. В данном случае также используются три уровня моделей, однако построенная модель предметной области находится «внутри» информационной системы в процессе ее эксплуатации. Таким образом, программное обеспечение информационной системы выступает в роли *интерпретатора*, а модель – в роли «*управляющей системы*», задающей правила функционирования ИС. Недостатком такого подхода является то, что несколько снижается производительность системы в ходе ее эксплуатации. Кроме того, если отсутствует возможность подключения внешних программных компонентов, расширяющих функциональность системы, то страдает универсальность вследствие невозможности реализации специфических для конкретной системы функций, отражающих бизнес-логику предметной области. Соответственно, метамодель должна быть максимально мощной. К достоинствам следует отнести тот факт, что при изменении модели не требуется повторное кодирование или перекомпиляция – информационная система просто начинает работать в соответствии с новой моделью.

Технология DSM (Domain Specific Modeling) *с генерацией кода* обеспечивает моделирование в терминах предметной области. В данном случае для решения каждой задачи применяется свой язык моделирования, в котором используются исключительно понятия и отношения из предметной области ИС.

Здесь используется уже мета-метамодель, которая реализуется Мета-CASE-средством. При помощи этого средства описывается метамодель, которая определяет предметно-зависимый язык моделирования. На основе этой модели генерируется CASE-средство, при помощи которого описывается модель предметной области и генерируется информационная система. Мета-CASE- и CASE-средства могут быть объединены в одну CASE-систему.

Использование предметно-зависимого языка (Domain Specific Language, DSL) позволяет существенно упростить процесс создания моделей предметной области, в котором могут принимать активное участие эксперты – специалисты в данной предметной области. Прочие преимущества и недостатки, связанные с генерацией кода, совпадают с соответствующими характеристиками традиционной CASE-технологии.

Технология DSM с интерпретацией метаданных (рис. 2) обеспечивает максимальные возможности адаптации. Данный вариант является комбинацией двух предыдущих. Метамодель, модель и данные ИС находятся «внутри» информационной системы. В этом случае CASE-средства позволяют создать модели и интерпретировать их в ходе эксплуатации системы (для этого разрабатываются специальные run-time компоненты).

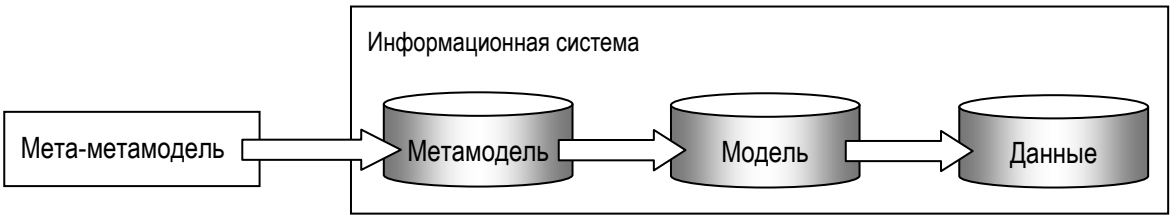


Рис. 2. Технология DSM с интерпретацией метаданных

Для того чтобы данный подход оказался применимым на практике, необходимо чтобы мета-метамодель была максимально выразительной. Интерпретация сразу двух уровней метамоделей приводит к ощутимой потере производительности, однако при достаточной выразительности мета-метамодели получается чрезвычайно гибкая система. Данные подход реализуется в представленной в данной работе CASE-технологии METAS, разрабатываемой в АНО науки и образования «Институт компьютинга».

Технология создания динамически адаптируемых информационных систем, управляемых метаданными

Максимальная гибкость ИС может быть достигнута, если как при разработке системы, так и в ходе ее эксплуатации применяются *метаданные*, описывающие особенности предметной области, для которой создается система, условия ее работы и характеристики бизнес-процессов и пользователей.

CASE-технология METAS (METAdata System) – это основа для создания динамически настраиваемых информационных систем, управляемых метаданными, повышения их адаптируемости в процессе эксплуатации за счет использования многоуровневых моделей. Ключевым моментом технологии является использование *взаимосвязанных метаданных*, описывающих информационную систему и ее окружение с различных точек зрения и на разных уровнях детализации.

Основное отличие данной CASE-технологии от других, генерирующих код приложений ИС на каком-либо языке программирования по заданным спецификациям, описывающим предметную область ИС (ее модель) и окружение, состоит в том, что в данном случае это описание используется во время работы программного ядра ИС, которое выполняет функции представления данных и их обработки по определенным этими метаданными правилам, *интерпретируя* их. Это создает хорошие предпосылки для создания «интеллектуальной» системы, которая может настраиваться на потребности пользователя и меняющиеся условия эксплуатации, *в ходе работы с ней пользователей*. Кроме того, при таком подходе проект обладает высокой степенью обратной связи, так как разработчик, меняя модели (метаданные), сразу видит соответствующие изменения в реализуемой на основе данной технологии ИС (в ее информационных объектах и связях между ними, в интерфейсе пользователя и функциональности и т.п.), потому что он фактически работает с той же системой, что и пользователь, но используя специальный CASE-инструментарий. При обычном же подходе, реализованном в большинстве CASE-систем, разработчик описывает модель системы, после чего выполняется генерация кода приложений, и только после этого он может оценить результат внесенных в модель изменений.

Технология METAS базируется на использовании языка UML и предметно-ориентированных языков для разработки моделей ИС, описания бизнес-правил, специфических для конкретных предметных областей; на технологии RUP (Rational Unified Process) и технологии разработки XP (eXtreme Programming); на платформе .NET Framework и инструментальных средствах MDK Suite (Meta Data Kernel Suite).

Метаданные представляют *формализованное описание* ИС, размещенное в базе метаданных (БМД), используемое для настройки приложения на условия эксплуатации в процессе его разработки, а затем – загрузки и выполнения. Они описывают следующие аспекты ИС: объекты ИС и поведение объектов ИС, бизнес-операции и бизнес-процессы, первичные документы и отчеты, визуальный интерфейс пользователя ИС, модель защиты. Метаданные представляют *модели*, каждая из которых описывает определенную часть, аспект ИС (некоторые модели могут описывать одни и те же части ИС, но с различных точек зрения). Таким образом, метаданные в METAS – это взаимосвязанные модели (рис. 3), одна модель может основываться на другой, и представлять собой более высокоуровневое описание ИС. Программные компоненты системы работают с метаданными соответствующего уровня (или нескольких взаимосвязанных уровней).

Метаданные ИС разделены на слои, представляющие следующие основные модели:

- *Физическая модель* (Physical Model) – метаданные, описывающие представление объектов ИС в БД (например, таблиц БД, в которой хранятся данные об объектах, и связей между ними). В процессе функционирования они служат основой логической модели. Модель автоматически генерируется по созданному на логическом уровне описанию системы.
- *Логическая модель* (Logical Model) – метаданные, описывающие сущности предметной области, для которой создается ИС, их поведение (через операции), а также общие операции ИС. Данная модель основывается на нотациях языка UML и позволяет работать пользователям системы в терминах предметной области.

- *Презентационная модель* (Presentation Model) – метаданные, описывающие визуальный интерфейс пользователя при работе с объектами ИС.

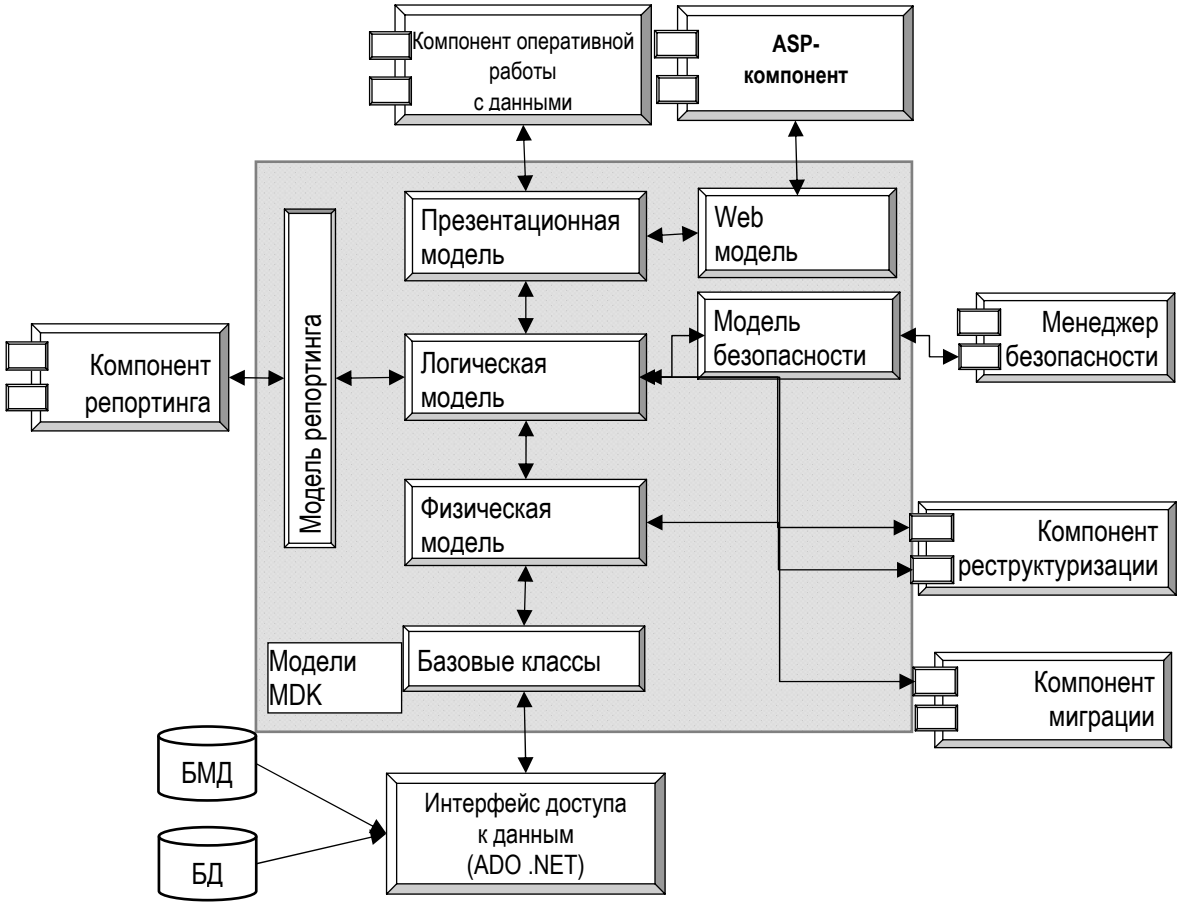


Рис. 3. Модели метаданных и компоненты METAS

Набор характеристик, отражаемых в модели, может быть динамически расширен. Набор метаданных может также расширяться путем добавления новых моделей, описывающих новые стороны и свойства ИС или существующие, но с новых точек зрения. В частности, в систему включены следующие модели, опирающиеся на перечисленные выше основные:

- *модель репортинга* (Reporting Model) – метаданные, описывающие запросы, первичные документы и отчеты, формируемые в ходе выполнения бизнес-операций и бизнес-процессов, используемые для анализа данных;
- *модель бизнес-процессов* (Business-process Model) – метаданные, описывающие бизнес-операции и бизнес-процессы, поддерживаемые ИС;
- *Web-модель*, которая обеспечивает доступ к ресурсам ИС для удаленных пользователей через Web-интерфейс.

Модель безопасности (Security Model) позволяет контролировать полномочия пользователей, их права на выполнение операций над объектами ИС или на доступ к моделям метаданных. Подсистема защиты работает с собственной БД.

CASE-инструментарий позволяет описывать объекты и бизнес-процессы ИС, строить запросы и отчеты в терминах предметной области, настраивать стандартно сгенерированные формы ввода и отображения данных, размещенных в БД системы, а также экспортировать и импортировать модели и данные динамически. Стандартная бизнес-логика может быть расширена путем определения новых типов и

операций, специфичных для конкретной ИС. Обеспечивается адаптация ИС без перепрограммирования ее компонентов и без участия разработчиков. Средства интеграции ИС на основе технологии BizTalk Server реализованы как отдельное приложение.

Архитектура информационной системы, созданной на основе технологии METAS

Использование CASE-технологии METAS позволяет создать ИС, архитектура которой представляет собой клиент-серверное приложение (рис. 4), разбитое на *домены*.

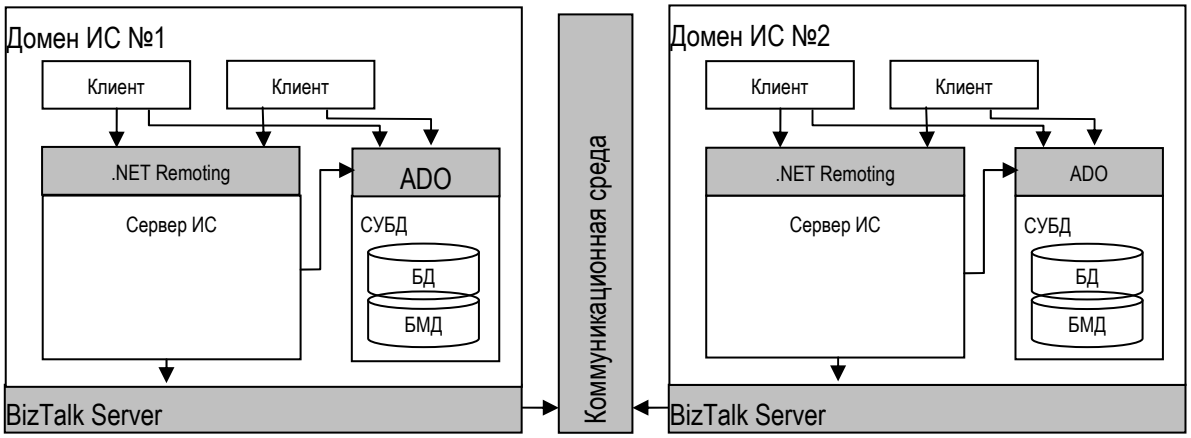


Рис. 4. Доменная архитектура ИС, созданной на основе METAS

Разбивка на домены предназначена для реализации распределенных ИС, включающих автономно функционирующие, не имеющие оперативной связи подсистемы. *Домен ИС* – законченное распределенное приложение, представляющее собой подсистему корпоративной ИС, установленную в отдельном учреждении, состоящее из одного сервера и нескольких клиентов.

Связь между доменами ИС устанавливается для реплицирования моделей и данных, обмена документами, отчетами.

Заключение

Основными преимуществами представленной технологии являются:

- *гибкость и возможность динамической адаптации* системы к изменениям условий функционирования, потребностям пользователей с минимальными затратами;
- *возможность интеграции* с внешними системами;
- *отсутствие необходимости специальной подготовки пользователей*, обеспечение возможности работы в привычной для них среде и в терминах знакомой предметной области;
- *невысокие требования к программно-аппаратной платформе* при достаточно мощных возможностях сбора, хранения и обработки данных.

Разработанные средства служат основой для разработки средств автоматической адаптации, основанных на использовании онтологий и агентных технологиях.

Благодарности

Работа выполнена при поддержке гранта РФФИ № 08-07-90006-Бел_а.

Библиографический список

[1] Chung L., Subramanian N. Adaptable system/Software architectures // Journal of Systems Architecture: the EUROMICRO. Special issue: Adaptable system/Software architectures. Vol. 50, Issue 7 (July 2004). Pp. 365-366.

[2] Subramanian N., Chung L. Software Architecture Adaptability: An NFR Approach // Proc. Int. Workshop on Principles of Software Evolution (IWPSE`01) / Vienna, Austria. ACM Press, (September 2001). Pp. 52 61. [PDF] (www.utdallas.edu/~chung/ftp/IWPSE.pdf).

[3] Cook St. Domain-Specific Modeling and Model Driven Architecture // MDA Journal, January 2004. Pp. 2-10. [PDF]. (www.bptrends.com/publicationfiles/01-04%20COL%20Dom%20Spec%20Modeling%20Frankel-Cook.pdf).

[4] Savidis A. Dynamic software assembly for automatic deployment-oriented adaptation // Preproceedings of the Workshop on Software Evolution through Transformations: Model-based vs. Implementation-level Solutions “SETra 2004”. A satellite event of ICGT 2004. October 2004, Rome, Italy. Pp. 191-198.

[5] Almeida J.P., Pires L.F., van Sinderen M. Costs and Benefits of Multiple Levels of Models in MDA Development // Proceedings of the Second European Workshop on Model Driven Architecture (MDA) with an emphasis on Methodologies and Transformations “Computer Science at Kent”. September 2004. Canterbury, UK. Pp. 12-21.

[6] Favre J.-M., Nguyen T. Towards a Megamodel to Model Software Evolution Through Transformations // Preproceedings of the Workshop on Software Evolution through Transformations: Model-based vs. Implementation-level Solutions “SETra 2004”. A satellite event of ICGT 2004. October 2004, Rome, Italy. Pp. 56-70.

[7] Atkinson C., Kühne Th. The Essence of Multilevel Metamodeling // Proceedings of UML 2001 – The Unified Modeling Language. Modeling Languages, Concepts, and Tools: 4th International Conference, V. 2185 of LNCS. Toronto, Canada, October 2001. Springer. Pp. 19-33.

[8] Лядова Л.Н. Технология создания динамически адаптируемых информационных систем // Труды междунар. науч.-тех. конф. «Интеллектуальные системы» (AIS’07). Т. 2. – М.: Физматлит, 2007. С. 350-357.

Сведения об авторе

Людмила Лядова *– заместитель директора Автономной некоммерческой организации науки и образования «Институт компьютинга»; Россия, 614097, г. Пермь, ул. Подлесная, 19/2-38;*
e-mail: LNLyadova@mail.ru

ОНТОЛОГИЧЕСКИЙ МЕТОД ДООПРЕДЕЛЕНИЯ ИМИТАЦИОННОЙ МОДЕЛИ

Александр Миков, Елена Замятина, Евгений Кубрак

***Аннотация**: В докладе представлена подсистема доопределения имитационной модели. Подсистема является компонентом системы имитации и автоматизированного проектирования вычислительных систем Triad.Net. В Triad.Net на ранних стадиях проектирования пользователь может опустить некоторые детали при описании модели проектируемого объекта. Тем не менее, ему необходимо получить оценки (пусть и приближенные) функционирования этого объекта. Подсистема доопределения по определенным критериям осуществляет поиск фрагмента программного кода, позволяющего доопределить модель. Различают автоматическое и полуавтоматическое доопределение модели. В статье приводится архитектура подсистемы доопределения, описаны алгоритмы автоматического доопределения, основанные на онтологическом подходе, указаны особенности полуавтоматического доопределения.*

***Ключевые слова**: Имитационное моделирование, системы автоматизированного проектирования, автоматическая генерация моделей, онтологии, OWL.*

***ACM Classification Keywords**: I.6 Simulation and Modeling I.6.2 Simulation Languages; J.6 Computer-aided Engineering; I.2 Artificial Intelligence I.2.5 Programming Languages and Software - Expert system tools and techniques*

***Conference**: The paper is selected from XIVth International Conference "Knowledge-Dialogue-Solution" KDS 2008, Varna, Bulgaria, June-July 2008*

Введение

Известно, что имитационное моделирование является одним из наиболее часто используемых методов при исследовании сложных систем и, в частности, при проектировании ВС.

Исследователи сложных систем часто сталкиваются с ситуацией необходимости анализа не полностью описанной модели. Обычно это выражается в том, что неизвестны в точности правила функционирования (поведение) отдельных элементов. Например, при моделировании компьютерных сетей проектировщик не всегда может точно определить алгоритм работы маршрутизатора. На ранних стадиях проектирования достаточно довольно грубо определить его поведение. Для исследователя важно, чтобы данные передавались по сети, а конкретный алгоритм работы маршрутизатора его временно не интересует, исследователь абстрагируется от этих деталей.

Ясно, что в таких условиях имитационное моделирование не даст абсолютно точной картины процессов, происходящих в сложной системе, тем не менее, приближенный результат может быть получен. Трудность заключается в том, что программная система имитационного моделирования не может провести сеанс имитации в отсутствие хотя бы одной процедуры, описывающей функционирование элементов моделируемой сложной системы. Требуется замещение отсутствующих процедур какими-либо имеющимися «подходящими» библиотечными процедурами.

В работе рассматривается подход, базирующийся на знаниях, представленных в виде онтологии моделируемой предметной области. Отсутствующие процедуры выбираются из библиотеки на основе информации о конкретных моделируемых элементах системы, представленной в виде семантического типа – особой метки, присваиваемой элементу модели для обозначения его функции, а также некоторых дополнительных ограничений. Выбор процедур подходящих семантических типов основывается на онтологической информации о моделируемой предметной области. Дополнительные ограничения позволяют более точно выбрать библиотечную процедуру.

Онтологический подход в имитационном моделировании

Известно, что онтология – это описание типов сущностей, существующих в предметной области, их свойств и отношений. Каждая предметная область (некая часть реального мира) может быть описана с помощью онтологий. Онтологии создаются и используются во множестве областей знаний, в том числе, известны примеры их успешного применения в имитационном моделировании. Однако создание онтологий для моделирования является достаточно сложной задачей, поскольку этот метод используют для исследования самых разнообразных систем, относящихся к различным предметным областям (химическим, физическим, транспортным и т.д.). Кроме того, методы имитационного моделирования основаны на математических, вероятностных и статистических расчетах, и, таким образом, онтологии для этих областей должны служить основой для всех остальных. Онтологии используют на различных этапах имитационного моделирования, начиная с этапа сбора информации о моделируемой системе и заканчивая этапом валидации модели [Fishwick 2004].

Примерами использования онтологий моделирования могут служить управляемые онтологиями среды моделирования, а также подходы к объединению различных федератов, разрабатываемые для HLA. Подход, разрабатываемый для HLA, использует онтологии для описания требований, которым должны удовлетворять интерфейсы федератов для успешного взаимодействия в федерации, а так же для разработки этих требований, с учётом знаний о моделируемой предметной области.

В работе [Liang, 2003] представлена онтология портов, рассматриваемая как средство автоматизации построения моделей из компонентов. Порты описывают интерфейс, определяющий границы компонентов или подсистем в конфигурации системы. Система представлена как конфигурация подсистем или компонентов, соединенных друг с другом через четко определенные интерфейсы. Онтологии успешно применяются и в других работах по имитационному моделированию [Benjamin, 2006].

Прежде, чем представить архитектуру подсистемы доопределения модели, описать критерии выбора подходящих библиотечных процедур, реализующих функционирование некоторого элемента модели, опишем особенности представления имитационной модели в Triad.Net.

Имитационная модель в Triad

Имитационная модель в Triad [Mikov, 1995] представляет собой совокупность объектов, которые действуют по определённым сценариям и обмениваются информацией друг с другом и может быть представлена тройкой: μ = {Str, Rout, Mes}.{Str), (Rout), (Mes) – это слой структур , рутин и сообщений соответственно.

Слой структур предназначен для описания моделируемых объектов и связей между ними, слой рутин представляет собой набор алгоритмов поведения моделируемых объектов, а слой сообщений даёт возможность описывать сообщения сложной структуры. Моделируемые объекты часто имеют иерархическую структуру. Имитационная модель также является иерархической. Каждый из уровней можно описать как граф с полюсами P = {U,V,W}, где V – множество вершин графа, каждая вершина представляет собой моделируемый объект, который находится на конкретном уровне иерархии. W – набор дуг, связывающих вершины графа (моделируемые объекты). U – набор внешних полюсов. Внутренние полюса используют для передачи сообщений на одном уровне иерархии. Их разделяют на входные In(V) и выходные Out(V). Набор внешних полюсов служит для передачи информации объектам, находящимся на различных (смежных) уровнях иерархии. Структура модели представляется иерархически, для этого вводится операция расшифровки объекта структурой. При выполнении этой операции некоторой вершине v структуры ставится в соответствие граф, описывающий внутреннюю структуру объекта системы, представленного вершиной v. Устанавливается так же соответствие между полюсами вершины и внешними полюсами расшифровывающего графа, таким образом, граф «общается» с окружением через интерфейс, предоставленный вершиной.

Рутина представлена множествами событий (E), состояний Q, моментов времени. Каждое состояние определяется набором значений локальных переменных (множество Var) каждой конкретной рутины. Поведение объекта системы, представляемого некоторой вершиной структуры, определяется путём наложения на неё соответствующей рутины. Операция наложения рутины во многом схожа с операцией расшифровки объекта. Рутина таким же образом представляет внутреннее устройство объекта структуры. Так же устанавливается соответствие между полюсами вершины и полюсами наложенной на неё рутины. Сообщения, приходящие на входные полюса вершины, мгновенно передаются на соответствующие полюса рутины, и наоборот: сообщения исходящие через выходные полюса рутины, передаются через соответствующие полюса вершины. Наложение рутины возможно только на терминальную вершину модели (т.е. не расшифрованную структурой), т.к. поведение расшифрованной вершины определяется поведением вершин её внутренней структуры.

Для сбора статистических данных о ходе моделирования, для анализа и представления результатов имитационного эксперимента в Triad используют специальные средства – информационные процедуры и условия моделирования. Информационные процедуры и условия моделирования реализуют алгоритм исследования. Алгоритм исследования отделён от модели. Пользователь имеет возможность изменить алгоритм исследования в ходе моделирования, при этом модель остаётся неизменной, нет необходимости вносить в неё какие-либо изменения, чтобы указать алгоритму исследования те элементы модели, за поведением которых надо вести наблюдение. Управление имитационным экспериментом осуществляется в условиях моделирования. В условиях моделирования указывают условия завершения моделирования, определяют набор информационных процедур, которые осуществляют сбор информации об имитационной модели.

Необходимо отметить особенность имитационных моделей в Triad: модель не является статической. В Triad определены операции над моделями в каждом из трёх слоёв [Mikov 1995]. Это операции в слое структуре: добавление и удаление вершины, добавление и удаление полюсов, добавление и удаление дуг, рёбер, объединение графов (модель представлена в виде графа), пересечение графов и т.д. В слое рутин – это добавление и удаление событий из графа событий. В слое сообщений – добавление и удаление типов и т.д. Кроме того, в языке Triad определен оператор наложения слоя сообщений. Заменив слой сообщений новым, исследователь имеет возможность (не изменяя модели) провести эксперимент с той же моделью, но с другими правилами преобразования данных.

Подсистема доопределения имитационной модели в Triad.Net

Как уже было сказано ранее, на начальных этапах проектирования исследователь может описать модель частично, опустив описание поведения какого либо элемента модели (μ_r* = {STR,ROUT*,MES}), не указав информационные потоки, воздействующие на модель (μ_s* = {STR*,ROUT*,MES}), не определив правила преобразования сигналов в слое сообщений (μ_m* = {STR,ROUT*,MES}). Однако для запуска модели и последующего её анализа все эти элементы должны быть так или иначе (пусть приближенно) описаны. В Triad.Net доопределение выполняется подсистемой доопределения модели. На рис.1. представлен процесс обработки имитационной модели.

Следует различать автоматическое и полуавтоматическое доопределение модели.

Полуавтоматическое доопределение моделей предполагает использование условий моделирования и погружение в среду моделирования. При полуавтоматическом доопределении исследователь вводит в часть initial условий моделирования: (а) операторы наложения рутин на вершину; (б) операторы наложения слоя сообщений; (в) операторы расшифровки вершины подструктурой; (г) операторы, реализующие операции над структурой модели (добавление и удаление вершин, дуг, входов и выходов и т.д.). При выполнении процесса имитации по оператору simulate симулятор вначале выполняет доопределение модели (при обработке операторов, записанных в части initial условий моделирования). Полуавтоматическое доопределение позволяет только на один акт имитации изменить воздействие информационных потоков (расшифровка вершины подмоделью, наложение рутины на вершину) или

условия преобразования сигналов (наложение слоя сообщений). Для того, чтобы провести исследования с другими информационными потоками или с другими правилами преобразования сигналов надо запустить процесс моделирования с другими условиями моделирования: simulate M on condition of simulation New_Condition (M.N1.a,M.N2.b) M.N1.a,M.N2.b – фактические параметры – переменные модели, за которыми ведется наблюдение в системе моделирования Triad.Net.

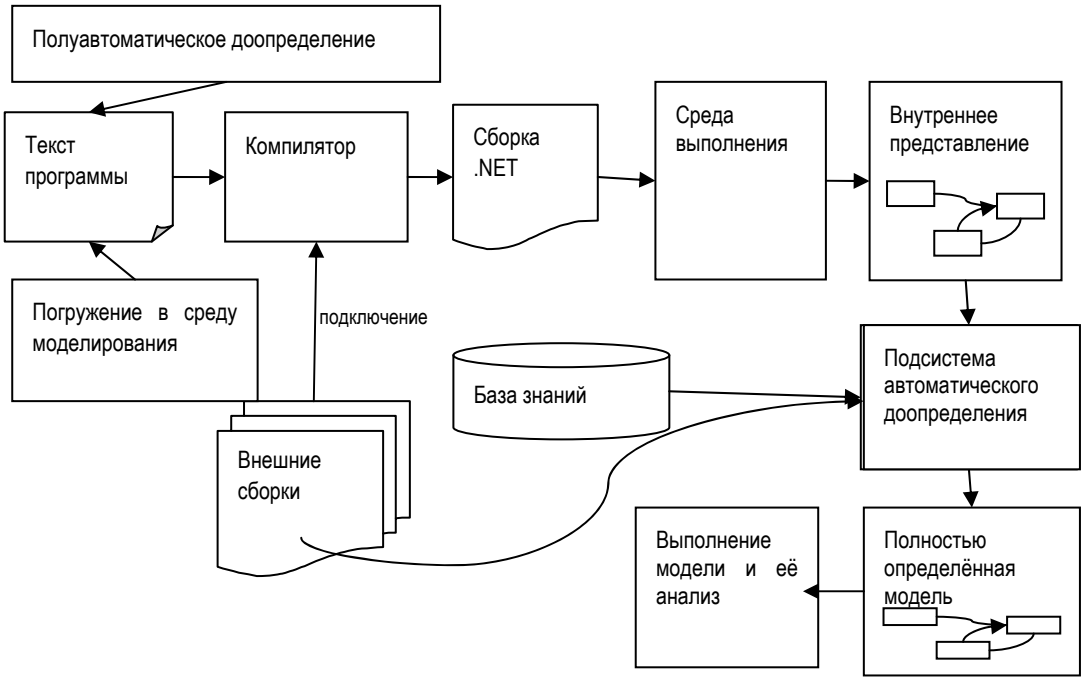


Рис.1. Структура системы доопределения моделей в Triad.Net

Погружение в среду моделирования позволяет оперативно изменить информационные потоки, поступающие на внешние полюса имитационной модели и выполняется с помощью оператора расшифровки вершины v графом, который представляет собой внутреннюю структуру моделируемого объекта, представленного вершиной v.

Автоматическое доопределение модели предполагает, что пользователь работает с частично описанной моделью μ_r^*, в которой не определены алгоритмы поведения некоторых элементов. Во время компиляции компоненты подсистемы автоматического доопределения моделей выявляют вершины v_i, для которых исследователем не определены рутины $ri = f(v_i)$, i=1..n. Задача подсистемы автоматического доопределения– найти по определенным критериям подходящие рутины в базе экземпляров рутин и достроить модель.

Подсистема автоматического доопределения модели

Рассмотрим пример моделирования компьютерной сети, представленной на рис. 2. и описание этого фрагмента на языке Triad. Каждая рабочая станция имеет два соседних узла: рабочую станцию и маршрутизатор. Сообщение должно быть передано от одной рабочей станции другой (не соседней). При передаче сообщений компьютерная сеть использует маршрутизатор. Точный сценарий поведения маршрутизаторов (Router) исследователю неизвестно. Задача системы автоматического доопределения модели состоит в том, чтобы для каждой вершины Router подобрать подходящую рутину из базы экземпляров рутин и выполнить действия, определенные оператором наложения рутины.

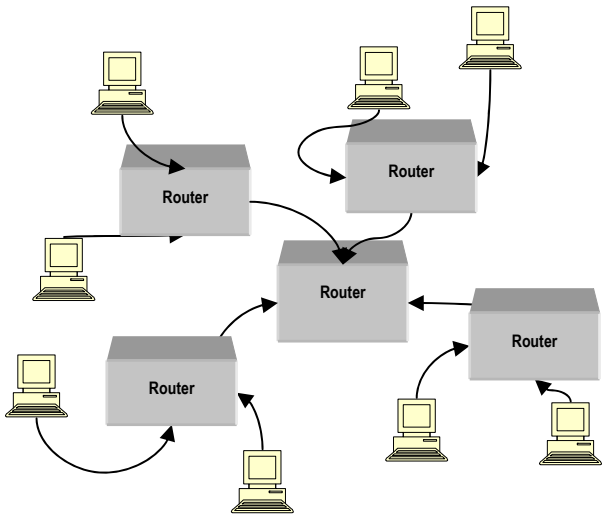


Рис.2. Фрагмент компьютерной сети

```
Type Router,Host; integer i;
   M:=dStar(Rout[5]<Pol[4]>);
   M:=M+node Hst[8]<Pol>;
   M.Rout[0]=>Router;
   for i:=1 by 1 to 4 do
       M.Rout[i]=>Router;
       M:=M+edge(Rout[i].Pol[1]—Hst[2*i-2]);
       M:=M+edge(Rout[i].Pol[2]—Hst[2*i-1]);
   endf;
   for i:=0 by 1 to 7 do
       M.Hst[i]=>Host;
   endf;
```

Рис.3. Описание фрагмента компьютерной сети на языке Triad с использованием семантических типов

Автоматическое доопределение в Triad выполняется на основании дополнительной семантической информации. Семантическая информация включает такое понятие как семантический тип. Семантический тип вводится для того, чтобы сгруппировать ряд объектов по некоторому смысловому, структурному, поведенческому типам. Так, для обозначения множества процессорных устройств при моделировании вычислительных систем, может быть введён семантический тип Процессор, для обозначения элементов памяти – тип ***МодульПамяти***. При моделировании систем массового обслуживания уместны будут семантические типы Очередь, ***ГенераторЗаявок*** и т.п. Для того, чтобы причислить объект к тому или иному семантическому типу, в тексте программы употребляется специальный оператор: <имя объекта> => <имя типа>. Семантические типы объявляются специальным оператором type <имя типа>. На рис. 3. приведен фрагмент программы, использующей семантические типы. В результате выполнения этой программы будет построена структура, описывающая небольшую сеть, терминальным вершинам которой будет присвоен семантический тип Host, а промежуточным – Router (Рис.4).

Семантические типы определяют смысловую нагрузку того или иного объекта модели. Для поиска экземпляра рутины используют базу знаний, представленной в виде онтологий. В этих онтологиях описывают семантические типы, отношения наследования между ними, а так же множества соответствующих этим типам экземпляров рутин, и семантической информации, необходимой для проверки условий доопределения. Семантических типы представлены в виде иерархии классов онтологии. Использование такого подхода предполагает, что дочерние семантические типы будут описывать понятия, конкретизирующие понятия, соответствующие родительским типам. Например, при моделировании вычислительных систем, на верхние уровни иерархии будут помещены семантические типы, соответствующие наиболее базовым понятиям, например Устройство. Дочерние типы будут представлять более конкретные понятия моделируемой предметной области: Устройства разделятся на ***Процессор, МодульПамяти*** и т.д., процессоры могут быть классифицированы в зависимости от их архитектуры. Таким образом, семантический тип представляется в базе знаний как класс объектов, являющийся подклассом общего типа Object, соответствующего всему множеству вершин. При определении семантических типов возможно указание нескольких семантических типов для одного объекта.

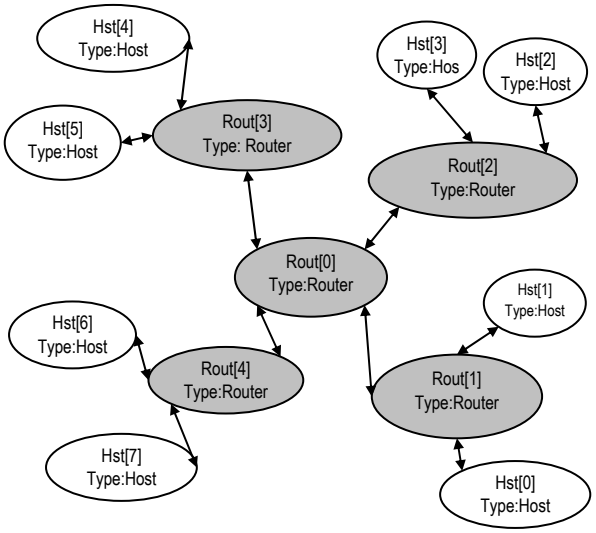


Рис.4. Внутреннее представление фрагмента модели, описанной программой на рис.3.

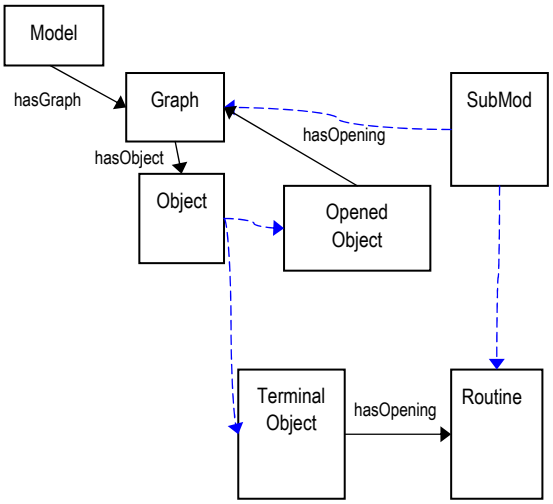


Рис.5. Фрагмент базовой онтологии

В системе Triad.Net выделены условия доопределения терминальной вершины экземпляром рутины: условия специализации, условия конфигурации и условия декомпозиции:

-- ***Условия специализации***. Пусть v – терминальная вершина, а r – экземпляр рутины, который соответствует этой вершине. Введём функцию equtype(v,r), определяющую выполнение условия специализации. Эта функция будет считаться истинной, если семантический тип Type(v), приписанный вершине, соответствует семантическому типу Type(r), соответствующему экземпляру рутины r, найденному в базе знаний. Семантический тип T1 соответствует семантическому типу T2, если T1 является суперклассом T2 (т.е. $T_2 \subset T_1$). Таким образом, условие специализации выполняется, если найденный экземпляр рутины соответствует семантическому типу вершины, или более частному типу.

-- ***Условия конфигурации***. Условие конфигурации предполагает проверку количества входных и выходных полюсов вершины и экземпляра рутины. При наложении рутины r на вершину v определяются отношения:

$$L_i : In(v) \to In(r) \tag{1}$$

$$L_o : Out(v) \to Out(r) \tag{2}$$

Отметим, что эти отображения не являются функциональными. Они задают множество связанных пар $(p_1; p_2)$, таких что $p_1 \in v; p_2 \in Pol(r)$, при этом каждый полюс может входить в любое количество пар, или не участвовать в отношении вообще. Пусть D(Li/o) – мощности множеств входных и выходных полюсов вершины, участвующих в отношениях Li и Lo соответственно. В зависимости от этих величин, вершина на которую предполагается наложить рутину должна иметь определённые количества входных и выходных полюсов, а именно, необходимо выполнение следующих условий:

$$\left|In(v)\right| \geq \left|D(L_i)\right| \tag{3}$$

$$\left|Out(v)\right| \geq \left|D(L_o)\right| \tag{4}$$

Использование нестрогого равенства позволяет накладывать экземпляры рутин на вершины, имеющие избыточное количество полюсов. При этом часть «лишних» входов и выходов вершины останется «висячими», т.е. не расшифрованными полюсами рутины, а сообщения, приходящие на эти полюса, не будут обрабатываться рутиной.

-- ***Условия декомпозиции***. Для определения условий декомпозиции, введём понятие графа окружения вершины v. Пусть $G = \{V; W; U\}$ - граф, которому принадлежит вершина ($v \in V$). Отношение S определяет смежность вершин в графе G, т.е.:

$$\forall v_1, v_2 \in V : (v_1; v_2) \in S \leftrightarrow \exists p_1 \in v_1, \exists p_2 \in v_2 : ((p_1; p_2) \in W \vee (p_2; p_1) \in W)$$

Функция Sub(w,v) определяет множество полюсов вершины w, соединённых с полюсами v:

$$Sub(w, v) = \{p \in w \mid \exists p_0 \in v : (p; p_0) \in W \vee (p_0; p) \in W\} \quad (5)$$

Тогда граф $GG(v) = \{V'; W'; \varnothing\}$ - граф окружения v, если выполняются следующие условия:

$$V' = \{v\} \cup \{w' \mid w' = Sub(w, v); (w; v) \in S\} \quad (6)$$

$$\forall w : (w; v) \in S \rightarrow Sub(w, v) \in V' \quad (7)$$

$$\forall p_1, p_2 : [(p_1; p_2) \in W] \wedge [p_1 \in v \vee p_2 \in v] \rightarrow (p_1; p_2) \in W' \quad (8)$$

$$W' \subset W \quad (9)$$

(6) ограничивает множество вершин графа окружения только самой вершиной v, и подмножествами вершин, смежных с ней. При этом учитываются только те полюса смежных вершин, которые, согласно (5), непосредственно связаны с полюсами v.(7) указывает, что такое подмножество включается для каждой из смежных с v вершин.(8) говорит, что каждая дуга, входящая или исходящая из полюсов v, будет включена в граф окружения. Из (6) и (7), очевидно, что все соответствующие полюса будут включены в граф. (9) утверждает, что никаких дополнительных дуг в графе окружения нет, т.е. включаются только дуги, соответствующие условию (8). Выполнение условия декомпозиции определяется значением функции iso(GG'(r),GG(v)), которое является истинным, если в графе окружения вершины v, есть изоморфный графу GG'(r) подграф. Функция проверяет также выполнение некоторых дополнительных требований, которым должен удовлетворять граф окружения. Граф GG' должен храниться в базе знаний вместе с экземпляром рутины. Таким образом, работа системы доопределения модели заключается в том, чтобы по сохранённым в базе знаний информации об условиях специализации, конфигурации и декомпозиции, для всех выявленных компилятором вершин без сценария поведения, найти соответствующие экземпляры рутины из базы знаний.

Представление знаний и базовая онтология

Для представления семантических знаний, необходимых для доопределения моделей, были выбраны онтологии, для представления онтологий – язык OWL[Dean, 2002], поскольку существует большое количество инструментальных средств работы с онтологиями OWL, поддерживающих возможность публиковать созданные онтологии в сети Internet и объединять информацию из различных источников, как локальных, так и находящихся в глобальной сети. Для работы с онтологиями используется инструментарий Jena OWL API. На данном этапе разработки онтологии сохраняются в виде текстовых файлов в формате представления Notation 3 (N3).

Семантические знания, которые необходимы для проверки условий автоматического доопределения, необходимо онтологическое описание слоя структур системы Triad.Net. В качестве такого описания используется базовая онтология, импортируемая всеми создаваемыми онтологиями. В этой онтологии определены следующие классы: Model (класс, описывающий множество моделей языка Triad), SubMod (класс, описывающий множество всех экземпляров рутин и структур), Graph (класс, описывающий множество структур моделей, является подклассом SubMod), Routine (множество экземпляров рутин, является подклассом SubMod), Object (множество всех вершин структуры модели, является суперклассом для всех семантических типов) и т.д. Фрагмент базовой онтологии представлен на рис.5. Для работы с самими онтологиями реализован класс OntoManager, поддерживающий загрузку нескольких онтологий, создание и сохранение онтологий. В классе TypeManager описаны функции, используемые классом, сохраняющим экземпляры рутин и классом доопределения для работы с семантическими типами вершин и экземпляров рутин.

Заключение

Подсистема доопределения модели позволяет автоматизировать процесс генерации моделей. Особенно это полезно в условиях неопределенности, когда исследователю неизвестны некоторые характеристики имитационной модели. В Triad.Net можно выполнить полуавтоматическое и автоматическое доопределение модели. Автоматическое доопределение выполняют на основе онтологий. Онтологический подход дает возможность доопределить модель, описанную пользователем лишь частично, а именно, с помощью онтологий найти в базе знаний фрагмент программы, который заменит рутину, описывающую наиболее точно сценарий поведения некоторого элемента модели.

Благодарности

Работа выполнена при финансовой поддержке грантов РФФИ 08-07-90005-Бел_а и 08-07-90006-Бел_а.

Библиографический список

[Fishwick, 2004] Fishwick P.A.. Ontologies For Modeling And Simulation: Issues And Approaches /Paul A. Fishwick, John A. Miller // Proceedings of the 2004 Winter Simulation Conference. pp. 259-264

[Benjamin, 2006]Benjamin P., Patki M., Mayer R.. Using Ontologies For Simulation Modeling. Proceedings of the 2006 Winter Simulation Conference/ L. F. Perrone, F. P. Wieland, J. Liu, B. G. Lawson, D. M. Nicol, and R. M. Fujimoto, eds. – pp.1161-1167

[Mikov, 1995] Mikov A.I. Simulation and Design of Hardware and Software with Triad// Proc.2nd Intl.Conf. on Electronic Hardware Description Languages, Las Vegas, USA, 1995. pp. 15-20.

[Dean, 2002] Dean M., Connolly D., van Harmelen F., et al. 2002. Web Ontology Language (OWL) Reference Version 1.0. W3C. //www.w3.org/TR/2002/owl-ref/

[Liang, 2003] Liang Vei-Chung. A Port Ontology For Automated Model Composition / Vei-Chung Liang, Christiaan J.J. Paredis // Proceedings of the 2003 Winter Simulation Conference, - pp. 613-622

Сведения об авторах

Александр Миков *– АНО «Институт компьютинга», директор; Россия, г. Краснодар, ул. Аксайская, 40/1-28; e-mail: alexander_mikov@mail.ru*

Елена Замятина *– Пермский государственный университет, доцент кафедры математического обеспечения вычислительных систем, Россия, г. Пермь, 614017, ул.Тургенева, .33–40; e-mail: e_zamyatina@mail.ru*

Евгений Кубрак *– Пермский государственный университет, выпускник кафедры математического обеспечения вычислительных систем, Россия, г. Пермь, ул. Леонова, .20-5; e-mail: g_brick@mail.ru*

ПОДХОД К ПРОГРАММИРОВАНИЮ АГЕНТОВ В МУЛЬТИАГЕНТНЫХ СИСТЕМАХ

Дмитрий Черемисинов, Людмила Черемисинова

Аннотация. *Рассмотрены методы спецификации протоколов взаимодействия агентов в мультиагентных системах на языке ПРАЛУ параллельных алгоритмов логического управления, который обладает средствами для представления последовательности состояний диалога, приема и отправки сообщений. Показано, что описание поведения агентов на языке ПРАЛУ позволяет моделировать поведение мультиагентной системы целиком. Предложена методология программирования агентов ПРАЛУ, использующая двухблочную архитектуру: блок синхронизации и функциональный блок. Оригинальной компонентой этой методологии является средство автоматической трансляции блока синхронизации по описанию на ПРАЛУ.*

Ключевые слова*: протокол взаимодействия, BDI агент, онтология, параллельный алгоритм.*

ACM Classification Keywords*: I.2.11 [Computer Applications]; Distributed Artificial Intelligence, Multiagent systems; D.3.3 [Programming Languages]: Language Constructs and Features – Concurrent programming structures.*

Conference*: The paper is selected from XIVth International Conference "Knowledge-Dialogue-Solution" KDS 2008, Varna, Bulgaria, June-July 2008*

Введение

Мультиагентные системы (МАС) состоят из множества искусственных агентов, которые работают совместно, чтобы достигнуть некоторых целей [1]. Агент представляет собой открытую систему, помещенную в некоторую среду, причем агенты обладают собственным поведением, удовлетворяющим определенным правилам. Примерами искусственных агентов служат роботы. МАС можно рассматривать как организацию агентов (по аналогии с человеческой организацией) или, другими словами, как некоторое искусственное сообщество. Технология программирования на основе использования взаимодействующих агентов считается наиболее перспективным инструментом современного программирования. Наиболее известной промышленной системой, построенной на основе концепции агентов и предназначенной для управления процессом производства изделий на предприятии, является ARCHON [1].

Агенты МАС характеризуются процессами, которые происходят во время их работы и определяются описанием их потенциального поведения. Взаимодействие агентов предполагает обмен сообщениями между ними. Множество взаимосвязанных сообщений образует переговоры в МАС, которые основаны на протоколах взаимодействия, определяющих все возможные течения переговоров.

В большинстве моделей МАС поведение агентов описывается в терминах убеждений, желаний и намерений (beliefs, desires and intentions – BDI) [2], то есть на основе понятий из социологии, а коммуникация задается в терминах протоколов, которые не имеют прямой связи с первыми. Одна из проблем в связи таким подходом состоит в том, что чрезвычайно трудно разрабатывать и моделировать коммуникацию между агентами. В большинстве моделей МАС автономное поведение агентов описывается с использованием формализмов высокого уровня абстрактности, а коммуникация задается в понятиях, близких к реализации. Разница в уровнях описания не позволяет моделировать коммуникацию между агентами на том уровне, на котором описано их автономное поведение. Эта проблема возникает вследствие отсутствия модели агента, объединяющей аспекты внутреннего состояния и коммуникации. Главная причина отсутствия общей модели состоит в том, что отсутствует общее концептуальное основание, объединяющее все абстракции, связанные с агентами.

Для преодоления этих методологических трудностей используются такие языки представления теории агентов, в основе которых лежит некий формализм и система программирования, задающие семантику

языка программирования агентов. Хотя в литературе предлагаются все новые языки программирования агентов, но немногие из них полностью понятны с семантической точки зрения.

В работе предлагается методология программирования агентов МАС, в основе которой лежит язык ПРАЛУ описания параллельных алгоритмов логического управления [3]. ПРАЛУ как язык программирования агентов имеет то преимущество, что он имеет в качестве семантики логический формализм и допускает простую реализацию. ПРАЛУ обладает средствами для представления последовательности состояний диалога, приема и отправки сообщений. Компактность представления программ и простота синтаксиса являются факторами, значительно упрощающими реализацию языка. Показано, что описание поведения агентов на языке ПРАЛУ позволяет моделировать поведение МАС целиком. В основе предлагаемой методологии лежит двухблочная архитектура организации описания и реализации МАС, состоящая из блока синхронизации и функционального блока.

Формализмы задания протоколов взаимодействия агентов

Протокол – это набор правил, которым соответствует взаимодействие, имеющее место при координации работы нескольких агентов. Формальные модели протоколов изучались в рамках теории распределенных вычислений. Фундаментальный признаком, по которому отличаются эти модели, является степень синхронизации поведения участников взаимодействия. Если абстрагироваться от назначения агентов, то единственной целью взаимодействия является синхронизация поведения взаимодействующих агентов, так как ненадлежащая синхронизация полностью разрушает целесообразность совместной работы агентов. Достижение синхронизации требует специальной организации взаимодействующих процессов.

Причинно-следственные и временные отношения являются главными характеристиками протоколов. Анализ этих зависимостей позволяет установить возможные последовательности возникновения событий при функционировании протокола, что дает возможность выявить, реализуются ли при выполнении протокола желательные события и обнаружить ошибки, вызывающие нежелательные события.

Когда агенты вовлечены во взаимодействие, где параллелизм не допустим, протоколы традиционно задаются детерминированными конечными автоматами. Самым простым из других представлений протокола является диаграмма потока сообщений, такая как используется в стандарте FIPA [4]. Для более сложных протоколов лучшим представлением является диаграмма взаимодействия таких языков, как UML (Uniffied Modelling Language – универсальный язык моделирования) [5], AUML [6] и цветных сетей Петри (CPN – Coloured Petri Nets) [7]. UML – один из наиболее популярных в настоящее время графических языков проектирования, который де-факто является стандартом для описания систем программного обеспечения. Язык AUML – это расширение UML для представления асинхронного обмена сообщениями между агентами. Представление протоколов практической сложности на языках автоматов, UML или AUML, к сожалению, требует существенных усилий для реализации агента, его отладки и понимания правил поведения.

Модель системы в виде CPN описывает множество состояний, в которых может находиться система, и переходы между этими состояниями. Формализм CPN обеспечивает математический базис для описания, реализации и анализа распределенных и параллельных систем, может выражать взаимодействия в графической форме и обладает строгой семантикой, что позволяет автоматизировать формальный анализ и преобразования описаний [7]. Используя CPN, протокол взаимодействия агентов может быть задан сетью компонентов, задающей структуру протокола. В CPN состояния взаимодействия агентов представляются местами CPN, передаваемые при взаимодействии данные – символами, значения которых указываются их цветами. Последовательность взаимодействий задают переходы и связанные с ними дуги. Переход допустим, если все его входные места имеют символы и цвета этих символов удовлетворяют ограничениям, которые определены на дугах. Если переход допустим, то он может быть запущен, а определяемые им действия выполнены. После выполнения перехода, состояние (маркировка) сети изменяется, и работа протокола заканчивается, когда нет допустимых или запущенных переходов.

Имеется множество работ по использованию обычных [8] или цветных сетей Петри для представления протоколов взаимодействия агентов, считается [7], что CPN – это один из лучших способов представления протоколов взаимодействия агентов. Однако понятие выполнения действия агентом в сети Петри не имеет явного представления [9], каждую роль агента нужно задавать отдельной сетью,

некоторые ситуации в поведении агентов не могут быть выражены стандартной сетью Петри. Все это значительно усложняет проектирование всего протокола в целом.

Спецификация протоколов взаимодействия агентов на ПРАЛУ

Язык ПРАЛУ удобно использовать для описания систем, характеризующихся сложным взаимодействием, асинхронностью и параллелизмом. Он (основан на расширенных сетях свободного выбора [10]) объединяет возможности моделей «если–то» с возможностями сетей Петри и обладает средствами для представления последовательности текущих состояний диалога, приема и отправки сообщений. С помощью языка ПРАЛУ [3] возможно описание временной упорядоченности событий, возникающих при реализации протокола, абстрагируясь от всех деталей, кроме тех, что выражаются причинно-следственными и временными отношениями. Алгоритмы на ПРАЛУ представляются в виде причинно-временных зависимостей между событиями, происходящими в МАС.

Основными операциями языка ПРАЛУ являются *операции ожидания и действия*. Операция ожидания «$-p_i$» сводится к ожиданию наступления некоторого события p_i, представленного конъюнкцией логических переменных, и ограничивается проверкой условия его истинности, завершаясь после ее выполнения. Операция действия «$\rightarrow A_i$» приводит к наступлению некоторого события, представленного также конъюнкцией логических переменных, в описываемом объекте (каким-то изменениям его состояния) и выполняется в течение некоторого промежутка времени после ее инициализации.

Алгоритм управления представляется *неупорядоченной совокупностью предложений*. Каждое предложение состоит из одной или нескольких одинаково помеченных цепочек «μ_i: $l_i \rightarrow \nu_i$»; через l_i обозначен некоторый линейный алгоритм, составленный из операций языка; μ_i и ν_i – начальная и конечная метки, которыми служат непустые подмножества из множества $M = \{1, 2, \ldots, m\}$ целых чисел. Допускается $\nu_i = \varnothing$, что обозначается как «$\rightarrow$.» и является концом реализации алгоритма.

Порядок выполнения цепочек алгоритма управления в процессе его реализации определяется множеством N запуска, его текущие значения $N_t \subset M$. Среди предложений алгоритма выделяется одно – начальное, его метка заносится в N перед реализацией алгоритма.

В процессе реализации алгоритма управления цепочки запускаются независимо друг от друга. Если в некоторый момент времени для некоторой цепочки «μ_i: $l_i \rightarrow \nu_i$» выполняется условие $\mu_i \subseteq N_t$ и реализуется событие p_i, с ожидания которого начинается цепочка l_i, то она запускается. При этом N_t заменяется на $N_t \setminus \mu_i$, а после завершения цепочки N_t становится равным $(N_t \setminus \mu_i) \cup \nu_i$. Предложенный механизм достаточен для отображения *альтернативного ветвления и распараллеливания* процессов. Синтаксически параллельный алгоритм характеризуется наличием меток $|\mu_i| > 1$, $|\nu_i| > 1$. Альтернативное ветвление обеспечивается ограничением $(i \neq j) \wedge (\mu_i \cap \mu_j \neq \varnothing) \rightarrow (p_i \wedge p_j = 0)$.

Язык ПРАЛУ поддерживает *иерархическое описание алгоритмов*, которое является особенно важным в случае описания сложных систем. Для обеспечения реакции устройства управления на некоторые особые события, происходящие в системе, в язык введена операция гашения в трех модификациях: «$\rightarrow *$», «$\rightarrow *\gamma$» и «$\rightarrow *\gamma'$», где $\gamma \subseteq M$. Ее действие заключается в прекращении реализации всех или некоторых (из множества γ или $\gamma' = N_t \setminus \gamma$) активных цепочек алгоритма. Наряду с булевыми в ПРАЛУ допустимо использование и арифметических переменных, в частности введены операции: задержки "$-n$" – выдержки n единиц времени; счета событий: «$\rightarrow (x = n)$» – присвоения многозначной переменной x натурального значения n; «$\rightarrow (x +)$» и «$\rightarrow (x -)$» – единичного положительного и отрицательного «$\rightarrow (x -)$» приращения значения; «$-(x = n)$» – ожидания наступления события: значение x равно n.

В работе [11] приведен пример описания на языке ПРАЛУ протокола мультиагентной системы, представляющей английский аукцион [4].

BDI агенты

Любая реальная МАС открыта – агенты работают в изменяющейся окружающей среде, которая воздействует на агентов и изменяется вследствие их работы. Система должна принимать решения, чтобы достичь поставленных перед ней целей, для этого она должна иметь модель среды, в которой развивается ее поведение. В области разработки моделей и архитектур агентов доминирующую роль

играет архитектура BDI, в которой агент рассматривается как социальное существо, общающееся с другими агентами посредством некоторого языка и играющее в обществе определенную роль, зависящую от убеждений (Beliefs), желаний (Desires) и намерений (Intentions) [2]. Считаются, что эти три компоненты полностью задают состояние «ума» социального агента.

В терминах программирования, убеждения BDI-агента представляют собой знания (информацию), которые имеет агент о состоянии окружающей среды и которые обновляются после каждого его действия. Желания обозначают цели, к которым стремится агент, включая их приоритеты. Намерения определяют действия, которые должны быть выполнены, чтобы достичь цели (образцы поведения). Протоколы взаимодействия позволяют агенту сократить пространство поиска возможных решений, определяя ограниченный диапазон ответов на сообщения, возможные для данной ситуации.

Онтология архитектуры BDI для языка ПРАЛУ

В практическом программировании агент – это оформленная в оболочку компьютерная система, которая расположена в некоторой окружающей среде и предназначена для гибких, автономных действий в этой среде с целью достижения заданных целей. Агенты отличаются от обычного программного обеспечения сложностью сценариев взаимодействия и коммуникации.

Онтологии – это явные формальные спецификации терминов предметной области и отношений между ними. Этот термин социальных наук, используемый в теории агентов, почти эквивалентен принятому в программировании понятию семантики языка в определенной предметной области. Задачей онтологии в нашем случае является определение терминов архитектуры BDI в понятиях языка ПРАЛУ.

Можно считать, что агент состоит из множества убеждений *B*, планов *P*, ситуаций *E*, действий *A* и намерений *I*. Когда агент замечает изменение в окружающей среде, он считает, что произошло событие, представляющее собой некоторую ситуацию внешней среды из *E*. Регистрация события агентом состоит в изменении состояния его «ума»: выбора некоторого убеждения из *B*. В соответствии с ним и желанием (определяемым некоторым планом из *P*) агент намеревается выполнить некоторые намерения из *I*, представляющие последовательности действий из *A*. Эти действия составляют план достижения поставленной цели. Таким образом, планируемое действие определяется выбранным планом из *P*. Затем планируемые действия осуществляются, изменяя текущую ситуацию в окружающей среде.

В традиционных параллельных языках программирования основными понятиями являются данные и управление вычислениями. Данные представляются значениями переменных, а управление задается набором процессов, которые трансформируют локальные состояния их памяти и задаются отображением переменных в их значения. Концепция интеллектуального агента вводит в параллельном программировании новые представления о манипуляции данными. Информационные состояния агента вместо отображений переменных в значения задаются более сложными информационными структурами логики предикатов первого порядка или модальной логики. Вычисления как трансформации локальных состояний памяти процессов представляют трансформации состояний «ума» агента. При программировании агентов это имеет два важных последствия. Во-первых, понятие присваивания значения переменной, на котором основаны традиционные языки программирования, нужно заменить операторами информационного обмена; во-вторых, механизм рассылки сообщений в MAC должен быть заменен механизмом, который обеспечивал бы обмен информацией между агентами в соответствии с некоторым протоколом. Таким образом, вычисления как трансформации состояний памяти заменяются протоколами коммуникации агентов.

Поведение агента (т.е. его взаимодействие с окружающей средой как внешнее поведение и процессы как внутреннее поведение) диктуется программой агента. Убеждения, желания и намерения агента в программе на ПРАЛУ не представляются явно как модальные формулы. Вместо этого проектировщик программы должен описать эти понятия теории агента внутри этой программы. Текущее состояние агента, которое объединяет состояние его «ума», окружающей среды и состояния других агентов, можно рассматривать как текущее состояние его убеждений. Состояния, которые агент собирается осуществить, основываясь на внешних или внутренних стимулах, можно рассматривать как его желания. Выбор плана действий для достижения поставленных целей можно рассматривать как намерения агента. Это изменение во взгляде на конструкции языка ПРАЛУ делает его не только языком спецификации, но и обеспечивает выполнимость описания поведения агента. Кроме того, мышление о поведении агента в

терминах убеждений, желаний и намерений, вероятно, дает хороший шанс для применения ПРАЛУ для программирования агентов, объединяя понятия теории МАС и практики проектирования параллельных алгоритмов управления.

Цель действия агента – это состояние окружающей среды, которое он хочет вызвать. На ПРАЛУ намерение достичь цели описывается операцией действия «$\rightarrow g$» (*DONE g*). Действия – основные единицы программы на ПРАЛУ, их выполнение является средством достижения цели. Действие можно выполнить, если некоторое убеждение оказалось верным. На ПРАЛУ проверка такого условия описывается операцией ожидания «– *a*» (*HAPPENS a*) события *a*. Предполагается, что агент выполняет действие, следующее за операцией «– *a*», только после того момента, когда «убеждение *a*» становится верным.

Программа агента является набором планов, определяющих средства, с помощью которых агент должен достичь конечной цели функционирования. План состоит из головных меток, тела и хвостовых меток. Тело плана – это последовательность действий, которая определяет цели, которые агент должен достигнуть, и условия, которые агент должен проверить. Головные и хвостовые метки плана символизируют намерения. Критическим понятием для задания поведения агента является понятие активного намерения. Предполагается, что агент имеет предопределенный список намерений. В каждый момент, когда агент выбирает некоторый план выполнения, намерение может быть активным или несущественным. В начале выполнения активно специальное начальное намерение *Init*. План будет выполняться только тогда, когда все намерения из числа его головных будут активны. Агент переводит эти намерения в несущественные, когда план выполнен, и делает активными все хвостовые намерения плана. Текущий набор активных намерений всегда не пуст. Тело плана и набор хвостовых намерений могут быть пустыми. В теории агентов существует понятие рациональности поведения агента (относительно достижения своих целей). Агент, заданный на языке ПРАЛУ, рационален в пределах заложенных в него планов и намерений.

Методология программирования агентов на ПРАЛУ

Программируя на языке ПРАЛУ, проектировщик агента указывает наборы его убеждений, планов и намерений. Такой стиль описания напоминает логическое программирование, в котором программа представляет собой спецификацию фактов и правил. Можно выделить следующие главные различия между логической программой и программой агента на языке ПРАЛУ.

– В логической программе нет различия между целью в теле правила и локальной переменной, описывающей условие ее применения. В программе агента на ПРАЛУ условия применения плана состоят из намерений, а не целей. Это обеспечивает планам на ПРАЛУ более выразительную форму, допуская управление выполнением планов как данными (используя добавления/удаления убеждений), так и намерениями.

– Правила в логической программе не контекстно-зависимы в отличие от планов на ПРАЛУ.

– Правила в логической программе указывают связывание переменных; в то время как выполнение планов имеет результатом последовательность действий, которые затрагивают окружающую среду.

– Если цель в логической программе становится не актуальной, выполнение доказательства не может быть прервано. Выполнение плана в программе агента на языке ПРАЛУ может быть остановлено операцией гашения этого языка, если при выполнении параллельных планов выяснилась неактуальность цели данного плана.

Процесс проектирования программы агента состоит из следующих этапов.

1. Анализ и разработка ПРАЛУ-описания агента, основываясь на неформальной спецификаци агентов МАС на языке ПРАЛУ.
2. Верификация логической непротиворечивости поведения МАС путем проверки корректности и моделирования ПРАЛУ-описания. Устранение обнаруженных ошибок.
3. Разработка программ, реализующих поведение каждого агента МАС, путем трансляции ПРАЛУ-описания поведения каждого агента в программу на языке программирования.
4. Тестирование сгенерированных программ.

Благодаря использованию языка ПРАЛУ как средства представления спецификаций агентов, а также выполнимости этих спецификаций, процесс проектирования становится структурированным. В нем разрешимы все формальные задачи, и для этих задач имеются программные средства их решения.

Предлагаемая стратегия проектирования естественным образом диктует разбиение программы на два блока: блок *синхронизации* и функциональную часть – блок *сопряжения*. Блок синхронизации координирует совместное выполнение параллельных процессов программы агента. Функциональная часть управляет данными и выполняет требуемые программой вычисления.

Это разбиение выполняется на уровне исходного описания: в спецификации на ПРАЛУ функциональная часть представлена предикатами, описывающими состояния памяти программы и внешней среды или предписывающими выполнение определенных действий. В ПРАЛУ-описании каждому предикату соответствует своя логическая переменная, установка единичного значения которой запускает процесс вычисления предиката. На этапе проверки логической непротиворечивости удобно интерпретировать предикаты как независимые логические переменные.

Такой подход позволяет отделить разработку синхронизирующей части ПРАЛУ-описания от функциональной. Отодвинув разработку функциональной части на более поздние стадии проектирования, можно добиться значительного упрощения проверки логической непротиворечивости поведения агентов.

Для трансляции ПРАЛУ-описания используется транслятор ПРАЛУ на промежуточный язык, применяемый в программе моделирования [12]. Этот транслятор строит символические представления программ вычисления реакций и управления датчиками и исполнительными механизмами агента. Управляющую структуру программы вычисления реакций составляет бесконечный цикл, заключающийся во вводе в оперативную память сигналов датчиков агента, формирующих его убеждения, вычислении по значениям этих сигналов реакции агента и выводе сигналов на его исполнительные механизмы.

Все связи по управлению и по данным между блоком вычисления реакций и блоком управления датчиками и исполнительными механизмами заложены в сгенерированном представлении ПРАЛУ-описания на промежуточном языке.

Чтобы реализовать параллельный алгоритм на одном процессоре [12], нужно упорядочить операции промежуточного языка с помощью планировщика промежуточного языка ПРАЛУ, который имеет свойства и методы. Свойства планировщика – две очереди: ждущих и готовых ветвей. Методы планировщика составляют: запуск, остановка и приостановка подпроцесса. Программа автоматически упорядочивает выполнение операций в процессе выполнения, и в предварительном планировании нет необходимости. Начальное состояние планировщика – очередь готовых содержит первую операцию алгоритма, а очередь ждущих пуста. Если планировщик обнаруживает, что очередь готовых пуста, он переносит содержимое очереди ждущих в очередь готовых, очередь ждущих становится пустой. Информационный обмен с внешней средой выполняется, когда очередь готовых пуста. Это гарантирует, что параллельные операции исходного алгоритма имеют равную продолжительность. Таким образом, программная реализация ПРАЛУ имеет семантику измеряемого времени с выполнением условия рандеву [13].

В этой модели программы предикаты функциональной части рассматриваются как «дополнительное оборудование» агента, формирующее логические сигналы или запускаемое по сигналу. Если при управлении требуется учитывать результаты выполнения расчетных операций, то «дополнительное оборудование» должно специально для этой цели вырабатывать логические сигналы. В программе управления датчиками и исполнительными механизмами это «дополнительное оборудование» описывается в виде выражений обычного языка программирования разработчиком программы агента.

Заключение

Язык ПРАЛУ как язык программирования агентов имеет то преимущество, что он имеет в качестве семантики логический формализм и допускает простую реализацию. Компактность представления программ и простота синтаксиса являются факторами, значительно упрощающими реализацию языка.

Для языка ПРАЛУ имеются программные инструменты, выполняющие имитационное моделирование системы по описанию ее поведения на ПРАЛУ. Имитационное моделирование позволяет объединить высокое быстродействие ЭВМ с гибкостью человеческого мышления. Проводя эксперимент и интерпретируя его результаты, проектировщик системы может контролировать неформальные проектные

операции. Использование ПРАЛУ для моделирования поведения протоколов позволяет получать информацию о логической корректности непосредственно, а не по результатам оценки статистики сеансов имитационного моделирования. При моделировании с целью анализа логической корректности важно рассматривать таймауты, т.е. учитывать параллелизм и асинхронность выполнения операций. Без учета таймаутов опасность ошибочных ситуаций значительно увеличивается.

Поддержка

Работа поддержана РФФИ НАН Беларуси (Проект F07-125).

Библиография

1. Lesser V. Cooperative Multiagent Systems: A Personal View of the State of the Art // IEEE Trans. Knowledge and Data Engineering, 1999. – Vol. 11. –No 1. – P. 133–142.
2. Burmeister B., Sundermeyer K. Cooperative problem-solving guided by intensions and perception, edited by E. Werner and Y. Demazean, Decentralized A.I. 3. – Amsterdam, The Netherlands, North Holland, 1992.
3. Закревский А.Д. Параллельные алгоритмы логического управления. – Мн.: Ин-т техн. кибернетики НАН Беларуси, 1999. – 202 с.
4. Foundation for Intelligent Physical Agents (FIPA). Communicative Act Library Specification, 2002. Available at http://www.fipa.org/specs/fipa00037/.
5. Booch G., Rumbaugh J., Jacobson I. The Unified Modeling Language User Guide – Addison Wesley, 1999.
6 Bauer B., Müller J.P., Odell J. Agent UML: A Formalism for Specifying Multiagent Interaction // Agent-Oriented Software Engineering , edited by P. Ciancarini and M. Wooldridge, Springer-Verlag, Berlin, 2001. – P – 91–103.
7 Quan Bai, Minjie Zhang Khin, Than Win A Colored Petri Net Based Approach for Multi-agent Interactions // 2nd Intern. Conf. on Autonomous Robots and Agents, Palmerston North, New Zealand, Dec. 13–15, 2004. – P. 152–157.
8 Nelson R.A., Haibt L.M., Sheridan P.T. Casting Petri nets into programs // IEEE Trans. Software Eng., 1983. – V. 9. – No 5. – P. 590–602.
9. Paurobally S., Cunningham J. Achieving Common Interaction Protocols in Open Agent Environments // AAMAS '02, Melbourne, Australia, 2002.
10. Hack M. Analysis of production schemata by Petri nets – Project MAC-94, Cambridge, 1972.
11. Cheremisinov D., Cheremisinova L. Developing Agent Interaction Protocols with PRALU // Information Theories & Applications (IJ ITA), 2006. – V. 13. – No 3. – P. 239–246.
12. Черемисинов Д.И. Реализация параллельных алгоритмов управления на одном микропроцессоре // Программирование, 1986. – № 1. – С. 37–45.
13. Cheremisinov D., Cheremisinova L. The specification of agent interaction in multi-agent systems // Proc. of the 5th Intern. Conf. «Information research and application (i.tech)», June 26–30, 2007, Varna, v. 2, Sofia: ITHEA, p. 428–434.

Информация об авторах

Дмитрий Иванович Черемисинов, Людмила Дмитриевна Черемисинова *– Объединенный институт проблем информатики Национальной академии наук Беларуси, ул. Сурганова, 6, Минск, 220012, Беларусь, e-mail: cher@newman.bas-net.by, cld@newman.bas-net.by*

ОПТИМИЗАЦИЯ ПОКАЗАТЕЛЕЙ ЖИВУЧЕСТИ СЕТЕЙ С ТЕХНОЛОГИЕЙ MPLS

Юрий Зайченко, Мохаммадреза Моссавари

***Аннотация**: В статье выполнен анализ живучести и оптимизация MPLS сетей. Введен индекс выживаемости и предложен метод ее оценки. Сформулирована задача оптимизации структуры сети MPLS, исходя из ее живучести, и разработан алгоритм ее решения. Рассмотрена задача реконфигурации сети в случае отказа ее элементов и предложен метод ее решения*

***Ключевые слова:** MPLS сеть, анализ живучести, оптимизация, реконфигурация маршрута*

***ACM Classification Keywords:** C.2. Computer-communication networks*

***Conference**: The paper is selected from XIV[th] International Conference "Knowledge-Dialogue-Solution" KDS 2008, Varna, Bulgaria, June-July 2008*

Введение

В последние годы в связи с резким увеличением объемов передаваемой информации в компьютерных сетях, необходимостью передачи аудио и видеоинформации, а также мультимедийной информации, возникла потребность в разработке новой коммуникационной технологии, способной обеспечить передачу разных видов информации (аудио, видео и данных) с заданным качеством обслуживания на высоких и сверхвысоких скоростях.

Первой технологией, обеспечивающей интегрированную передачу аудио, видео информации и данных стала технология ATM (Asynchronous Transfer Mode). Однако жесткое ограничение на размер передаваемых ячеек- 53 байта, а также высокая дороговизна используемого оборудования, в частности коммутаторов ATM, препятствуют ее широкому применению. Поэтому в конце 90-х годов 20 века на смену ей пришла технология многопротокольной коммутации меток MPLS (Multiprotocol Label Switching).

Эта технология предоставляет единый транспортный механизм для сетей, которые используют протоколы TCP/IP, Frame Relay, X.25,ATM. Она базируется на введении потоков различных классов обслуживания (CoS), установлении приоритетов в обслуживании различных классов и обеспечении требуемого качества обслуживания (Quality of Service – QoS) для соответствующих классов [Олвейн, 2004].

Важной задачей, возникающей при проектировании сетей MPLS, является задача анализа и оптимизации показателей живучести сетей MPLS. В работах [Зайченко, 2005; Зайченко, 2006] были предложены показатели живучести для сетей с технологией MPLS и предложен алгоритм их анализа. Целью настоящей работы является разработка и исследование метода оптимизации показателей живучести сетей с технологией MPLS и алгоритма реконфигурации сетей в случае отказов ее элементов.

Постановка и модель задачи анализа живучести

Следуя работе [Зайченко, 2005], под живучестью системы будем понимать её способность сохранять своё функционирование и обеспечивать выполнение основных функций (в уменьшенном объеме) при заданных показателях качества обслуживания.

Поскольку основное назначение сети с технологией MPLS является передача заданных величин входящих потоков различных классов, то живучесть сети MPLS будем оценивать величиной максимального потока, который возможно передать в сети при отказах ее элементов-каналов и узлов при сохранении заданных показателей качества.

Пусть имеется сеть MPLS, которая описывается орграфом $G = \{X, E\}$, где $X = \{x_j\}$ множество узлов сети (УС), $E = \{(r,s)\}$ -множество каналов связи (КС); μ_{rs} - пропускные способности КС.

Допустим, что в сети передается K классов потоков (K=$\overline{1,6}$) (CoS) в соответствии с матрицами требований $H(k)=\left\|h_{ij}(k)\right\|$ $i=\overline{1,N}$, $j=\overline{1,N}$ (Мбит/с). Для каждого класса k введен показатель качества (QoS) в виде заданной величины средней задержки $T_{cp,k}$, которая оценивается следующим выражением [Зайченко, 2005]:

$$T_{cp,k}=\frac{1}{H_{\Sigma}^{(k)}}\sum_{(r,s)\in E}\frac{f_{rs}^{(k)}\sum_{i=1}^{k}f_{rs}^{(i)}}{\left(\mu_{rs}-\sum_{i=1}^{k-1}f_{rs}^{(i)}\right)\cdot\left(\mu_{rs}-\sum_{i=1}^{k}f_{rs}^{(i)}\right)}, \quad (1)$$

где $H_{\Sigma}^{(k)}=\sum_{i=1}^{n}\sum_{j=1}^{n}h_{ij}^{(k)}$, μ_{rs}-пропускная способность канала связи (r,s), $f_{rs}^{(k)}$- величина потока k-го класса в канале (r,s).

Требуется определить для данной сети показатели живучести.

В работе [Зайченко, 2005] для анализа живучести сетей MPLS был введен следующий комплексный показатель;

$$P\left\{H_{\Sigma}^{\Phi}(1)\geq r\%H_{\Sigma}^{0}(1)\right\},P\left\{H_{\Sigma}^{\Phi}(2)\geq r\%H_{\Sigma}^{0}(2)\right\},...,P\left\{H_{\Sigma}^{\Phi}(k)\geq r\%H_{\Sigma}^{0}(k)\right\}, \quad (2)$$

где $H_{\Sigma}^{0}(k)$ - величина потока k-го класса в безотказовом состоянии сети; $H_{\Sigma}^{\Phi}(k)$ - фактическая величина потока класса k в случае действия отказов, r=(50÷100)%, $k=\overline{1,K}$. Таким образом для оценки живучести сетей с технологией MPLS используется векторный показатель вида (2).

Алгоритм оценки показателей живучести сети MPLS

Рассмотрим сеть MPLS G=(X,E), состоящую из элементов (каналов и узлов), подверженных воздействию внешней среды, в результате которого они выходят из строя. Предполагается, что заданы надежностные характеристики элементов сети - коэффициенты готовности каналов $k_{\Gamma r,s}$ и узлов $k_{\Gamma,i}$, $(r,s)\in E$, $i=\overline{1,n}$. Используя модель активной внешней среды, можно определить вероятность каждого состояния $P\{Z_0\}$. Например, если Z_i - это выход из строя КС $(r_i s_i)$, то

$$P(Z_i)=(1-K_{\Gamma,r,s})\prod_{(r,s)\neq(r_i,s_i)}K_{\Gamma r,s}, \quad (3)$$

где $K_{\Gamma r,s}$ – вероятность исправного состояния КС, $(r,s)\neq(r_i,s_i)$, 1-$K_{\Gamma r,s}$- вероятность вывода из строя КС (r,s).

В работе [2] был предложен алгоритм оценки показателей живучести сети MPLS, суть которого состоит в следующем.

1. Вычисляем общую величину потока в безотказном состоянии для всех классов сервиса: $H_{\Sigma}^{(0)}(1),H_{\Sigma}^{(0)}(2),...,H_{\Sigma}^{(0)}(K)$.

2. Моделируем различные отказовые состояния: Z_1,Z_2,Z_3,Z_4,Z_5. Для каждого из них рассчитываем вероятности $P(Z_i)$ согласно (3).

3. Находим величину максимального потока всех классов в состоянии Z_j : $H_{\Sigma}^{\Phi}(k,z_j)$, $k=\overline{1,K}$. Для этого используется специально разработанный алгоритм нахождения максимального потока

4. Вычисляем комплексный показатель живучести для каждого класса сервиса:

для первого класса

$$P\left\{H_{\Sigma}^{\Phi}(1) \geq r\% H_{\Sigma}^{0}(1)\right\} = \sum_{Z_j} P(Z_j) \tag{4}$$

где суммирование в (4) происходит по всем Z_j таким что $H_{\Sigma}^{\Phi}(1) \geq r\% H_{\Sigma}^{0}(1)$;

для k-го класса

$$P\left\{H_{\Sigma}^{\Phi}(k) \geq r\% H_{\Sigma}^{0}(k)\right\} = \sum_{Z_i} P(Z_i) \tag{5}$$

где суммирование в (5) происходит по всем состояниям Z_i таким, что:

$H_{\Sigma}^{\phi}(k) \geq rH_{\Sigma}^{(0)}(k)$; $H_{\Sigma}^{0}(k)$ - величина потока k-го класса в безотказовом состоянии сети; $H_{\Sigma}^{\Phi}(k)$ - фактическая величина потока класса k в случае действия отказов, r =(50÷100)%, $k = \overline{1,K}$;

Z_i: $H_{\Sigma}^{\phi}(r) \geq kH_{\Sigma}^{(0)}(r)$.

Полученные зависимости $P\left\{H_{\Sigma}^{\Phi}(1) \geq r\% H_{\Sigma}^{0}(1)\right\}, P\left\{H_{\Sigma}^{\Phi}(2) \geq r\% H_{\Sigma}^{0}(2)\right\}, \ldots, P\left\{H_{\Sigma}^{\Phi}(k) \geq r\% H_{\Sigma}^{0}(k)\right\}$ построим в координатах $P\{H_{\Sigma}^{\phi}(k)\}$ - $r\% H_{\Sigma}^{0}$.

Постановка задачи оптимизации сети MPLS по показателям живучести

В ходе проектирования сетей по результатам анализа ее показателей живучести возникает проблема обеспечения требуемого уровня живучести. Естественно, что эта задача может быть решена путем резервирования ее каналов и узлов, структурной оптимизации сети и требует дополнительных затрат материальных средств. Поэтому далее в работе рассматривается постановка задачи структурной оптимизации сети по показателям живучести.

Пусть имеется сеть MPLS, которая описывается орграфом $G = \{X, E\}$, где $X = \{x_j\}$ множество узлов сети (УС), $E = \{(r,s)\}$ - множество каналов связи (КС); μ_{rs} - пропускные способности КС.

Допустим, что в сети передается K классов потоков (К=$\overline{1,6}$) (CoS) в соответствии с матрицами требований $H(k) = \left\| h_{ij}(k) \right\|$ $i = \overline{1,N}$, $j = \overline{1,N}$ (Мбит/с). Для каждого класса k введен показатель качества (QoS) в виде заданной величины средней задержки $T_{ср,k}$. Пусть, исходя из функционального назначения сети, установлены следующие значения показателей живучести для потока k-го класса; $P_{0зад}^{(k)}, P_{1зад}^{(k)}, \ldots, P_{5зад}^{(k)}$.

Требуется определить такую структуру сети, для которой для всех классов K будут обеспечиваться следующие ограничения по уровню живучести:

$$P\left\{H_{\Sigma}^{\phi}(k) \geq r\% H_{\Sigma}^{(0)}(k)\right\} \geq P_{кзза}, \; r\text{=(50÷100)\%}, \; k = \overline{1,K} \tag{6}$$

а дополнительные затраты средств будут при этом минимальными.

Достижение требуемого уровня живучести будем обеспечивать путем введения соответствующего резервирования наиболее ответственных элементов сети (КС и УС).

Для оценки эффективности резервирования каналов и узлов вводится следующий показатель

$$\alpha_{r_i s_i} = -\frac{\Delta P(Z_i)}{C_{r_i s_i}} \tag{7}$$

где Z_i - состояние выхода из строя КС $(r_i s_i)$; $\Delta P(Z_i)$ - изменение вероятности состояния Z_i в случае резервирования, $C_{r_i s_i}$ – стоимость такого резервирования. Величина $\Delta P(Z_i)$ оценивается по следующей формуле:

$$P_{рез}(Z_i) - P(Z_i) = (P^2_{отк\, r_i s_i} \cdot \prod_{(r,s)\neq(r_i,s_i)} K_{\Gamma\, r,s} - P_{отк\, r_i s_i} \prod_{(r,s)\neq(r_i,s_i)} K_{\Gamma\, r,s} =$$
$$= -(1 - P_{отк\, r_i s_i}) \cdot P_{отк\, r_i s_i} \prod_{(r,s)\neq(r_i,s_i)} K_{\Gamma\, r,s} = -(1 - P_{отк\, r_i s_i}) \cdot P(Z_i) \quad (8)$$

Аналогичные соотношения используем и для оценки резервирования УС.

Показатель $\alpha_{r_i s_i}$ используется для выбора первоочередных элементов (КС и УС) для резервирования.

Предложим следующий алгоритм оптимизации сети MPLS по показателям живучести.

Оптимальная реконфигурация маршрутов в сетях MPLS при отказах в задаче обеспечения заданного уровня живучести

Выше была сформулирована задача анализа живучести сети при отказах её элементов и предложен алгоритм оценки показателей живучести, основанный на решении задачи нахождения максимального потока (НМП) в сети при отказах её элементов.

Для вычисления величины $H_\Sigma(z_j)$ при отказе, например, КС (r_j, s_j) предполагалось, что в этом состоянии находится полностью новое распределение потоков всех требований и новые маршруты, которые устанавливаются в сети. Однако такой случай является идеальным, и не соответствует реальным условиям функционирования маршрутов сети MPLS, так называемых LSR (Label Switching Routers). На практике, в случае отказа некоторого КС, соседние маршрутизаторы реконфигурируют потоки всех соединений, проходившие через КС (r_j, s_j) на другие маршруты, причем так, чтобы по возможности не нарушить другие соединения в сети и сохранить установленные показатели качества сервиса (QoS) для них.

Постановка задачи. Пусть задана сеть MPLS со структурой $G = \{X, E\}$, $X = \{x_j\}$ $j = \overline{1,n}$ - множество узлов сети (УС), $E = \{r, s\}$ - множество каналов связи (КС) сети, заданы также пропускные способности (ПС) всех КС μ_{rs}, $(r,s) \in E$, матрицы требований $H(k) = \|h_{ij}(k)\|$ $i, j = \overline{1,n}$, $h_{ij}(k)$ - интенсивность потока k-го класса, который необходимо передавать из узла i в j (Кбит/с), распределение потоков всех классов $F(k) = [f_{rs}(k)]$, где $f_{rs}(k)$ - величина потока класса k, передаваемого по КС (r,s), соответствующая матрице $H_\Sigma(k)$. Введены также ограничения на значения показателя качества (QoS) для всех классов в виде $T_{ср}^{(k)} \le T_{зад,k}$,

Известны также маршруты коммутации меток LSP $\{T_{ij}(k)\}$ для каждого соединения (пары) (i,j), которые устанавливаются с помощью протокола RSVP или SNMP.

Допустим, что произошел отказ КС (r_j, s_j), обозначим это отказовое состояние z_j. Требуется реконфигурировать все маршруты отказавшего соединения КС (r_j, s_j) таким образом, чтобы в максимальной степени удовлетворить соответствующие требования, получившие отказ в обслуживании при сохранении остальных соединений по объему трафика и заданному качеству QoS - $T_{ср}$. Назовём эту задачу оптимальной реконфигурацией сети MPLS при отказах**.**

Математическая модель данной задачи имеет следующий вид:

Требуется найти такое распределение потоков $[f_{rs}(k)]$, при котором обеспечивается:

$$H_\Sigma = \sum_{(i,j):(r_j,s_j)\in \Pi_{ij}} h_{ij}^{(корр)} \to \max, \quad (9)$$

при условиях

$$T_{ср}\left(F_{корр}^{(k)}\right) \le T_{зад},\ k=\overline{1,K}, \tag{10}$$

$F_{кор}^{(k)}$ - скорректированный поток в КС (r,s) после реконфигурации k -го класса сервиса.

Приведем алгоритм оптимальной реконфигурации сети MPLS, состоящий из 2-х этапов.

На первом этапе определяются все требования (соединение) (i,j), которые использовали КС $\left(r_j,s_j\right)$ и отключаются временно от сети и пересчитываются потоки в КС $F^{(k)}=\left[f_{rs}^{(k)}\right],\ (r,s)\in E$.

На втором этапе определяются резервы по ПС всех КС и оптимальным образом перераспределяются потоки отказанных требований так, чтобы обеспечить достижение критерия $H_\Sigma \to \max$ (назовём их отказовыми требованиями).

1 ЭТАП

1. Находим все требования (i,j) проходившие через отказанные соединения КС $\left(r_j,s_j\right)$. Обозначим их

$$P_{r_j,s_j}=\left\{(i,s):\left(r_j,s_j\in \Pi_{ij}\right)\right\}. \tag{11}$$

2. Временно отключаем передачу информации для требований множества P_{r_j,s_j} и вычисляем новые значения потоков - $F^{н}(k)=\left[f_{rs}^{н}(k)\right]$:

$$f_{rs}^{н}=\begin{cases} f_{rs}-\sum\limits_{(i,j):\Pi_{ij}\Rightarrow(r,s)} h_{ij}, & \text{где}\quad (i,j)\in P_{r_j,s_j} \\ f_{rs}, & \text{в остальных случаях} \end{cases}.$$

Это выполняется следующим образом:

1. Находим первое требование $(i_1,j_1)\in P_{r_j,s_j}$.
2. Полагаем $h_{i_1,j_1}=0$ и вычисляем новое распределение потоков:

$$f_{rs}^{н}(k)=\begin{cases} f_{rs}(k)-h_{i_1,j_1}, & \text{если}\quad (r,s)\in \Pi_{i_1,j_1} \\ f_{rs}(k), & \text{в остальных случаях} \end{cases}. \tag{12}$$

3. Проверка условия: $P_{r_j,s_j}\setminus(i_1,j_1)\neq 0$, если ДА, то на шаг 1 и повторяем шаги 1-3 до исчерпания множества P_{r_j,s_j}. В результате получим новое распределение потоков всех классов $F^{н}(k)=\left[f_{rs}^{н}(k)\right]$, включающее потоки только от неотказанных требований $(i,j)\setminus P_{r_j,s_j}$.
4. Определяем резервы по ПС всех КС:

$$Q_{рез,r,s}=\mu_{rs}-\sum_{k=1}^{K} f_{rs}^{н}(k). \tag{13}$$

Переходим ко второму этапу.

2 ЭТАП

Для требований множества P_{r_j,s_j} находим новые маршруты (реконфигурируем маршруты) так, чтобы обеспечить выполнение условий:

$$T_{ср}\left(F_{корр}^{(k)}\right) \le T_{зад},\ k=\overline{1,K}, \tag{14}$$

и при этом $\sum_{(i,j)\in P_{r_j,s_j}} h_{ij}^{(корр)} \to \max$.

Второй этап состоит из k подэтапов, на каждом из которых осуществляем реконфигурацию маршрутов и переопределение потоков для k -го класса.

Подэтап 1

1. $k=1$. Сначала распределяем отказанные требования класса 1.

2. Находим условную метрику $\frac{\partial T_{ср}}{\partial f_{rs}^{(1)}} \mid f_{rs}^{н}(1)$.

3. Находим кратчайшие пути $П_{ij}^{\min}(1)$ для всех отказанных требований класса $k=1$.

4. Выбираем первое требование $(i_1, j_1) \in P_{r_j,s_j}$ такое, что $l(П_{i_1,j_1}^{\min}(1)) = \min\limits_{(i,j)} l(П_{i,j}^{\min}))$.

5. Проверяем возможность передачи его в полном объеме по пути $П_{i_1,j_1}^{\min}$

$$h_{i_1 j_1} < Q_{рез}\left(П_{i_1,j_1}^{\min}\right), \tag{15}$$

где $Q_{рез}\left(П_{i_1,j_1}^{\min}\right)$ - свободная ПС маршрута $П_{i_1,j_1}^{\min}$

$$Q_{рез}\left(П_{i_1,j_1}^{\min}\right) = \min_{(r,s)\in П_{i_1,j_1}} \{\mu_{rs} - f_{rs}\}. \tag{16}$$

Если условие (15) выполняется, то распределяем полностью поток требования $h_{i_1 j_1}$ по маршруту $П_{i_1,j_1}^{\min}$ и находим скорректированное распределение потоков (РП):

$$f_{rs}^{н}(k) = \begin{cases} f_{rs}^{н}(1) + h_{i_1,j_1}, & \text{если} \quad (r,s) \in П_{i_1,j_1}^{\min} \\ f_{rs}^{н}(1), & \text{в остальных случаях} \end{cases}. \tag{17}$$

Иначе - на шаг 6.

6. Полагаем $h_{i_1,j_1}^{(a)} = Q_{рез}\left(П_{i_1,j_1}^{\min}\right) - \Delta$. Здесь $h_{i_1,j_1}^{(a)}$ - доля требования h_{i_1,j_1}, передаваемая по маршруту $П_{i_1,j_1}^{\min}$, Δ - некоторая заданная величина.

7. Находим скорректированное распределение потоков:

$$f_{rs}^{корр}(k) = \begin{cases} f_{rs}^{н}(1) + h_{i_1,j_1}^{(a)}, & \text{если} \quad (r,s) \in П_{i_1,j_1}^{\min} \\ f_{rs}(1), & \text{в остальных случаях} \end{cases}. \tag{18}$$

Проверяем, выполняется ли ограничение на $T_{ср,1}$:

$$T_{ср}\left(F_{корр}^{(1)}\right) \le T_{зад,1}. \tag{19}$$

Если условие (19) выполняется, то $P_{r_j,s_j}^{н} = P_{r_j,s_j} \setminus (i_1, j_1)$ и переход на шаг 9.

8. Проверка условия: $P_{r_j,s_j}^{н} \neq 0$? Если ДА, то переходим к следующей итерации и распределяем очередное требование класса $k=1$. Иначе - на шаг 9.

9. Конец первого подэтапа.

Далее для всех $k = \overline{1, K}$ следующие подэтапы выполняются по вышеприведенной схеме. На каждом из них выполняем перераспределение отказанных требований второго класса, причем так, чтобы не нарушалось условие:

$$T_{ср}\left(F_{корр}^{(k)}\right) \le T_{зад,k}\,. \tag{20}$$

Последовательность этих подэтапов заканчивается либо полным распределением всех отказанных требований (что маловероятно), либо при выходе на границу по всем ограничениям:

$$T_{ср}\left(F_{корр}^{(k)}\right) \ge T_{зад,k}\ \text{, для всех } k = \overline{1, K}\,. \tag{21}$$

Это означает полное исчерпание всех свободных ресурсов (свободной полосы) каналов связи.

В результате работы алгоритма находятся оптимальные реконфигурированные пути (LSP) для соединений (i, j), получивших отказ в обслуживании из-за отказа соответствующего КС или УС. Естественно, что в силу ограниченной ПС сети при этом некоторые соединения обслуживаться не будут. При этом, учитывая очередность реконфигурации, отказ в обслуживании получат наименее приоритетные соединения, а величина общего потока, передаваемого в сети после реконфигурации – минимальной.

Выводы

Сформулирована задача оптимизации показателей живучести сети с технологией MPLS в случае отказов её элементов-каналов и узлов.

1. Введены показатели живучести сетей и предложен алгоритм их оценки, учитывающий специфику сетей с технологией MPLS.
2. Сформулирована задача оптимизации сетей по показателям живучести и предложен алгоритм ее решения, позволяющий достичь заданных значений показателей живучести при минимальных дополнительных затратах
3. Предложен алгоритм оптимальной реконфигурации сети MPLS в случае отказов, позволяющий максимально использовать коммуникационные ресурсы сети и максимизировать величину потока передаваемого через сеть в случае отказов.

Литература

[Олвейн, 2004] Олвейн Вивьен. Структура и реализация современной технологии MPLS. Перевод с английского. Изд. дом «Вильямс», 2004. – 480 с.

[Зайченко, 2005] Зайченко Ю.П., Мохаммадреза Моссавари. Анализ показателей живучести компьютерной сети с технологией MPLS // Вісник національного технічного університету “КПІ”. Інформатика, управління та обчислювальна техніка. - Вип. 43. – 2005. - С. 73-80.

[Зайченко, 2006] Зайченко Ю.П., Мохаммадреза Моссавари. Оптимизация компьютерных сетей с технологией MPLS по показателям живучести в случае активной внешней среды // Вісник національного технічного університету “КПІ”. Інформатика, управління та обчислювальна техніка. Вип. 45. – 2006. - С.163-172.

[Зайченко, 2006] Зайченко Ю.П., Аникиев А.С., Мохаммедреза Моссавари, Ашраф Абдель-Карим Хилал Абу-Аин. Синтез структуры глобальных компьютерных сетей с технологией MPLS при ограничениях на показатели качества и живучести // Электроника и связь. - 2006. - № 2. - С. 68-71.

Информация об авторе

Зайченко Юрий Петрович, *профессор, д.т.н., декан факультета, «Институт прикладного системного анализа». Киев, НТУУ «КПИ», ул. Политехническая 14. тел: +8(044)241-86-93, e-mail: zaych@i.com.ua*

Мохаммадреза Моссавари *(Иран), аспирант кафедры «Прикладная математика» НТУУ «КПИ», проспект Победы 37, тел +380677099063*

ПСИХОЛОГИЧЕСКИЕ ПРОБЛЕМЫ И ПЕРСПЕКТИВЫ РАЗВИТИЯ ДИАЛОГА „ЧЕЛОВЕК - КОМПЬЮТЕР”

Ирина Сергиенко

***Аннотация**: Статья посвящена проблеме влияния диалога "человек-компьютер" на формирование коммуникативных качеств личности пользователей и разработчиков. В статье рассмотрены психологические проблемы взаимодействия человека с техническими интеллектуальными системами. Проанализированы новые психологические эффекты такие, как "сужение диапазона эмоций", "размывание (или наоборот - повышение жесткости) коммуникативных границ", "трудности в понимании психологического контекста диалога человека с человеком", "монологический стиль взаимодействия с людьми" пользователя. В статье показаны перспективные направления в развитии систем "человек – система искусственного интеллекта" для преодоления названных психологических эффектов может стать такие пути, как: во-первых, качественное преобразование интерактивных функций интеллектуальных технических систем, во-вторых, интенсификация непосредственного межличностного взаимодействия разработчиков, программистов на основе принципов оптимального диалогического взаимодействия (в стиле взаимодействия - "сотрудничество"). Представлены психологические рекомендации для развития коммуникативного потенциала личности, длительно взаимодействующей с система искусственного интеллекта.*

***Ключевые слова**: Диалог "человек – компьютер", системы искусственного интеллекта, диалог "человек – человек".*

***ACM Classification Keywords**: H.1.2 User/Machine - Software psychology*

***Conference**: The paper is selected from XIVth International Conference "Knowledge-Dialogue-Solution" KDS 2008, Varna, Bulgaria, June-July 2008*

Введение

Психология предоставляет научное и практическое обеспечение исполнения задач социального развития. Одна из них – исследование психологических аспектов взаимодействия человека с системами искусственного интеллекта. В этом контексте может быть рассмотрено несколько вопросов: во-первых, психологическое содержание диалога "человек – компьютер (здесь и далее, система искусственного интеллекта)"; во-вторых, влияние особенностей психики человека (в том числе, индивидуально-психологических особенностей пользователя-программиста и пользователя-непрограммиста) на взаимодействие с технической интеллектуальной системой; в-третьих, влияние диалога пользователя с системами искусственного интеллекта на его психику.

Психологические проблемы взаимодействия "человек – система искусственного интеллекта"

В рамках исследования влияния диалога ***"человек - система искусственного интеллекта"*** на психику человека мы предлагаем обратить внимание разработчиков систем искусственного интеллекта и прикладных психологов на такой аспект как изменение коммуникативных способностей пользователя в результате длительного взаимодействия человека с компьютерными системами, в частности, на его диалог в системе ***«человек - человек»***. В этой проблеме выделяем для рассмотрения три момента: усечение структуры и средств общения человека; фиксация способа организации совместно-распределенной деятельности в системе «человек – искусственный интеллект» и перенос ее сферу отношений «человек – человек»; личностные деформации у профессионалов, работающих в сфере взаимодействия с техническими интеллектуальными системами.

Для анализа первого аспекта влияния диалога «человек – система искусственного интеллекта» на коммуникативные способности личности пользователя рассмотрим сущность и структуру общения, его средства; условия формирования способности человека к взаимодействию в другими людьми и внутриличностной ядерной концепции «Я» в условиях диалога со значимыми людьми.

Процесс развития личности и ее навыков диалога с другими людьми не прекращается всю жизнь. В ходе жизни коммуникативная сфера человека может обогащаться или сужаться, ценностно переориентироваться, развиваться и совершенствоваться или стереотипизироваться и регрессировать. Тенденции актуального развития личности зависят от ведущей деятельности и основного типа взаимодействия человека со средой.

В психологии под общением мы понимаем социально-психологическое взаимодействие людей, которые вступают в психологический контакт между собой с целью решения тех или иных актуальных жизненных задач. Высшим уровнем взаимодействия людей, с нашей точки зрения, является диалогическое взаимодействие как творческая интегративная субъект-субъектая система, которая имеет потенциал саморазвития и самоусовершенствования участников [Сергиенко, 2005]. В этом случае, партнер по взаимодействию воспринимается субъектом как уникальная, неповторимая личность, обладающая ценностью сама по себе, являющаяся равноправным субъектом в выдвижении встречных инициатив в отношении потребностей, желаний, целей, идей и т.п.

Если провести психологический анализ взаимодействия человека и системы искусственного интеллекта, то можно обнаружить, что принцип "субъект-субъектности" в этом "диалоге" нарушается, поскольку "система искусственного интеллекта" всегда является "объектом" (средством) для исполнения потребностей, целей и задач человека, который является "субъектом". Взаимодействие "человека с системой искусственного интеллекта" обладает развивающим потенциалом для пользователя, однако специальной цели личностного развития партнера по диалогу в этой диаде не выдвигается.

Общение людей имеет сложную многослойную систему взаимодействия человека с человеком. В своей целостной форме оно включает такие уровни: ценностно-смысловой (презентация и обмен ценностями, личностными смыслами, установление личностных границ, выдвижение личных нравственно-этических норм, предъявление и защита личностной позиции, экзистенциальных установок и т. д.); когнитивный (получение, сохранение, передача, обмен, переработка, создание информации как предметной, так и социально-психологической и т. д.), эмотивный (выражение, передача, получение, обмен, рефлексия, торможение или активация эмоций и чувств и т. д.) и поведенческий (обмен действиями, средства установления, поддержания контакта для осуществления взаимодействия и выхода из него, организация совместной деятельности, система управления совместной деятельностью, распределение ролей и функций в совместной деятельности т. д.).

Пример А.П. Журавлева и Н.А. Павлюка, приведенный в их работе «Язык и компьютер» [Журавлев,1989] позволяет нам проиллюстрировать наличие разных уровни взаимодействия в общении людей между собой посредством языка (как вербального диалогического средства). Они приводят историю о баснописце Эзопе, который был рабом у философа Ксанфа: «Однажды хозяин Эзопа велел приготовить самое лучшее блюдо. Эзоп приготовил говяжий ЯЗЫК. В ответ на недоумение Ксанфа он сказал:»Что может быть на свете лучше языка? При помощи языка люди могут объясняться…, выражать ласку, признаваться в любви. Поэтому нужно думать, что нет ничего лучше языка». В следующий раз Ксанф велел приготовить самое худшее. Эзоп опять приготовил блюдо из языка и сказал: «Ты велел мне сыскать самое худшее. А что на свете хуже языка? Посредством языка люди огорчают и разочаровывают друг друга, посредством языка можно лицемерить, лгать, обманывать, хитрить, ссориться. Язык может сделать людей врагами, … он может вносить в нашу жизнь горе и зло, предавать, оскорблять. Может ли быть что-нибудь хуже языка?» [Журавлев, 1989, с. 85]. Хотя авторы обращают в своей работе больше внимания на семантику слова, эмоциональные заряды слов, меняющиеся в зависимости от этапа развития социума, но наглядно показывают, что *диалог человека с человеком* содержит в себе огромную палитру ценностно-смысловых, эмоциональных аспектов, которые играют решающую роль в организации жизнедеятельности человека, в формировании позитивных отношений с другими людьми, в достижении ими счастья и успеха.

При взаимодействии человека с техническими интеллектуальными у пользователя часто возникает иллюзия общения с реальной личностью, поскольку диалог происходит на языке, понятном человеку,

производится обмен информацией. Таким образом, включение когнитивного уровня взаимодействия, аналогичного тому, который имеет место в системе "человек-человек", создает основу для незаметной сознанию пользователя подмены реального партнера по общению виртуальной когнитивной моделью, которую представляет техническая интеллектуальная система. Другие слои общения человека при этом могут постепенно редуцироваться, упрощаясь и сужаясь по диапазону и глубине переживания и внешнего выражения. Психическая функция, которая не востребована во взаимодействии человека с системой искусственного интеллекта, будет постепенно регрессировать. Эта тенденция выражается в таких эффектах, как "сужение и уплощение эмоций, участвующих в непосредственном общении человека с человеком", "размывание нормативно-ценностной сферы личности в сфере взаимодействия с другими людьми". С другой стороны, усечение структуры общения, обеднение эмоциональных и ценностных слоев взаимодействия с другим человеком, приводит к возникновению психологических трудностей в общении с реальными партнерами. Личность, стремясь к облегчению своего эмоционального состояния и снижения психологической нагрузки от реального "трудного" общения. неосознанно удаляется от реального взаимодействия с реальными людьми и укрепляется в своем центрировании на "общении" с виртуальным "простым и понятным" партнером. Этот психологический эффект мы определяем как "защитный уход в виртуальную реальность", сопровождающийся явлением "повышения жесткости и непроницаемости коммуникативных границ для реальных собеседников".

Отражение в другом человеке своего «Я» обеспечивает основу для формирования личности. Многие социально-психологические знания и навыки общения человека имеют происхождение в способе взаимодействия с ребенком родителей и других близких и значимых людей в детстве, из той инфорамации, которую осознанно и неосознанно предоставляли личности окружающие люди. Этот процесс реализуется через механизм "обратной связи", который активируется в процессе общения. Специалисты в области эргономических исследования диалога "человек – мащина" пишут следующее: "Социальная взаимозависимость начинается с момента оплодотворения зародышевой клетки и продолжается в течение всего жизненного пути. При кибернетическом подходе к пониманию этой взаимозависимости социальные взаимодействия описываются с помощью понятия "обратная связь" и соответствует моделям слежения и управления. Самые разнообразные формы социального поведения – подражание, прослеживание взглядом, ухаживание, речь, социальная координация в процессе работы или игры, спортивная или художественная деятельность – могут быть проанализированы как следящие системы, регулируемые социальной обратной связью" [Кристенсен, 1991, с. 547]

В интерпространстве субъект ежесекундно получает информацию о реакциях других людей на него как партнера по взаимодействию (на когнитивном, эмотивном и поведенческом уровнях). Эта информация селективно принимается субъектом для дальнейшей обработки и проверки на ее релевантность собственной сознательной Я-концепции. Часть сведений остается в оперативной памяти, обеспечивающей накопление и применение жизненного опыта в каждый момент взаимодействия с людьми.

Личность, центрированная на взаимодействии с "системой искусственного интеллекта", значительно уменьшает свою возможность получать личностно-ориентированную обратную связь от окружающих людей, которая побуждала бы его к реалистической оценке себя, а также к личностному и межличностному самосовершенствованию. Техническая интеллектуальная система не дает пользователю негативной личностно-ориентированной обратной связи, которая могла бы в устах людей иметь "болезненную" форму, поэтому появляется дополнительный "аргумент" для подсознания человека в пользу выбора виртуального, простого во взаимодействии "партнера". Человек, достигающий успеха в компьютерных играх, написании компьютерных программ и т.д., ощущает повышение самоуважения В связи с сужением непосредственного общения и уходом от реального взаимодействия с живыми людьми человек (пользователь) не накапливает нового жизненного опыта. Личность может "застревать" на том этапе своего личностного и социально-психологического развития, на котором произошла фиксация на "виртуальном партнере" (компьютере).

При обычном повседневном общении люди используют множество средств для контакта на всех четырех уровнях взаимодействия: вербальные и невербальные, которые в свою очередь имеют свои разновидности и тонко дифференцируются внутри себя. "В процессе социального слежения движения первого субъекта генерируют входные стимулы для второго субъекта, который регулирует эти входные стимулы с помощью движений и создает соответствующие входные стимулы для первого субъекта и т.д."

[Кристенсен, 1991, с. 547]. Большую роль в общении играют такие средства, как жесты, мимика. пантомимика, паралингвистические компоненты речи, которые создавая тонкие контексты, дают человеку дополнительную уточняющую информацию о смысле, заложенном в вербальном сообщении, или об отношении к собеседнику, с которым ведется диалог, об отношении к информации, которая передается и т. д. Можно также учесть, что невербальный диалог в общении людей может иметь место сам по себе без соединения с вербальным обменом сообщениями.

«Ореол» взаимодействия в виде «фоновых», незаметных для сознания, но существенных для подсознания латентных сигналов-средств общения, помогает субъекту при взаимодействии с людьми точнее понимать потребности и мотивы другого человека, его нормы и границы, его личностную позицию, его чувства и эмоции. Обработка вторичных сигналов-средств общения происходит в основном на подсознательном уровне в правом полушарии человека и реализуется в сознании в виде интуитивного результата переработанной информации – мысли или чувства (эмоции). Для развития интуитивного способа обработки информации необходимо накопление информационного багажа – базы контекстных данных. Эта информация на неосознанном уровне психики человека собирается в ходе его непосредственного взаимодействия с людьми. Вторичные сигналы-средства общения являются знаками для субъекта, их воспринимающего, но не согласовываются социумом, их применяющим, и не имеют однозначно установленных связей между сигналом-средством и его значением.

Вербальные и символьные средства общения (язык, слова, знаки) представляют собой часть общепринятой (социально-консенсусной) системы знаков, комбинация которых создает новые знаки и тексты, имеющие общепринятое значение. Смысл и значение их доступно для понимания всем участникам общения в границах группы, овладевшей информационным наполнением (содержанием) выбранной системы знаков. Обработка информации в вербальной и общепринятой символьной форме осуществляется в основном в левом полушарии мозга. Работа оператора, программиста, инженера-системотехника связана в основном с обработкой информации вербального и абстрактно-символьного вида, что ведет в свою очередь в перегрузке левого полушария мозга и ослаблении функций правого полушария. В этом случае правое полушарие человека недополучает информацию о сфере организации отношений с людьми образного, эмоционального, телесно-кинестетического и кинетического типа, что ухудшает возможности для интуитивного решения задач социально-перцептивного типа, а также осложняет понимание контекста, и, следовательно, дополнительного, личного и скрытого смыслов в высказываниях партнера по общению. Незагруженность правого полушария мозга образной и эмоционально-насыщенной информацией приводит к усилению потребности личности в "сильных" и "ярких" ощущениях, порождающих в свою очередь необходимость в таких видах активности, которые предоставляют человеку подобные состояния. Для некоторых личностей, такими видами деятельности, становятся занятия опасными видами спорта, компьютерные и "азартные" игры, для других – потребление психоактивных веществ и т.д. Зависимости, которые могут сформироваться в этом случае, носят характер компенсирующих механизмов, поэтому, для избавления личности от их влияния необходимо определение первичной психологической проблемы, которая привела к их возникновению. Может иметь место остановка социально-психологического развития личности и ее снижаться ее способность успешно взаимодействовать с людьми в формате современного психологического пространства и времени. В результате у пользователя могут наблюдаться такие психологические эффекты, как: "сужение диапазона эмоций", "размывание (или наоборот - повышение жесткости) коммуникативных границ", "трудности в понимании психологического контекста диалога человека с человеком", "редукция интуиции в коммуникативной сфере", "монологический стиль взаимодействия с людьми" пользователя.

Сравнительный анализ психологической структуры взаимодействие в системе «человек – человек» и «человек- система искусственного интеллекта» показывает следующее:

Во-первых, в результате фиксации личности на диалоге с искусственным интеллектом происходит усечение целостной структуры общения и используемых личностью средств общения (паралингвистических, кинестетических, кинетических и др.), нарушение понимание контекста сообщения партнера и дополнительных смыслах, заложенных во «вторичных» сигналах-средствах общения, редукция интуиции, размывание значимости ценностно-нормативной и эмоциональной функций общения.

Вторым существенным моментом может быть изменение типа взаимодействия человека-оператора с другими людьми. Диалог в системе „человек – система искусственного интеллекта” лишен партнерской

функции (демократического стиля общения), при которой каждый из участников взаимодействия вносит равный вклад в инициативность, выдвижение предложений для решения проблемы, принятие окончательного решения о выборе способа решения актуальной задачи, взаимпродолжаемое интеллектуальное действие (стиль сотрудничества), выполнение решения. Субъектом в системе взаимодействия „человек – система искусственного интеллекта" всегда является одна сторона (человек), а исполняющим агентом – другая сторона «система искусственного интеллекта». Наблюдается значительный перевес субъектной функции со стороны человека, что проявляется в виде разработки программ, выдачи «команд» машине, которая «должна" исполнить их. Проблема такого взаимодействия для человека возникает лишь в том, чтобы он точно, на языке системы искусственного интеллекта дал ей нужную программу, команду, указание. Такой стиль взаимодействия относится к типу «авторитарный» или «манипулятивный».

Третий момент заключается в том, что результатом длительного взаимодействия человека с техническими интеллектуальными системами может явиться личностная профессиональная деформация. Человек, работающий постоянно в данной системе взаимодействия с технической интеллектуальной системой, может неосознанно переносить монологический когнитивно-деятельный паттерн на построение отношений с людьми. Это в свою очередь может приводить к нарушению демократического стиля общения с людьми, возникновению конфликтный ситуаций, потере психологического контакта с другими людьми, профессиональной деформации личности оператора, программиста. Возникает «порочный круг», когда человек, получивший психологическую травму в отношениях с людьми, находит во взаимодействии с «системой искусственного интеллекта» возможность сбалансировать свое эмоциональное состояние. Прогнозируя дальнейшее развитие личности, постоянно взаимодействующей с техническими интеллектуальными системами, можно проследить тенденции: к одностороннему (левополушарному) развитию психики; уходу от реальных трудностей непосредственного межличностного общения во взаимодействие в виртуальном пространстве и с виртуальным собеседником; сужению коммуникативной среды; консервации или регрессу навыков общения; эмоциональной депривации; возникновению эмоционально-замещающих деятельностей и формированию психологических зависимостей от них и т. д.

Психологические рекомендации по оптимизации диалога "человек-система искусственного интеллекта"

Перспективными для преодоления названных психологических эффектов в развитии систем "человек – система искусственного интеллекта" могут стать такие пути, как: во-первых, качественное преобразование интерактивных функций технических интеллектуальных систем, во-вторых, интенсификация непосредственного межличностного взаимодействия разработчиков, программистов на основе принципов оптимального диалогического взаимодействия (в стиле взаимодействия - "сотрудничество").

Психопрофилактическими мерами первого направления, по нашему мнению, могут быть:

1) создание нового типа диалога „человек - компьютер" путем введения в функции интеллектуальной технической системы инициативной, активаторной или блокирующей, экспертно-оценочной и консультативной диалогической функции;

2) предоставление технической интеллектуальной системой пользователю обратной связи в человеческих формах «образного и эмоционального» типов реагирования информации с мимическими и пантомимическими коммуникативными элементами;

3) учет не только технических, эргономических, психологических требований при разработке технических интеллектуальных систем, но и социально-психологических, педагогических, этических норм, обуславливающих влияние диалога человека с компьютером на формирование социально-перцептивных, эмотивных и коммуникативных свойств личности пользователя.

Психопрофилактическими мерами второго направления, по нашему мнению, могут быть:

4) разработка проектов, компьютерных программ при непосредственном общении участников лицом к лицу в диаде и/или в «команде»;

5) обучение программистов навыкам сотрудничества при работе в диаде и группе «лицом к лицу»;

6) психологическая диагностика стиля взаимодействия, навыков общения, уровня развития умений осуществлять диалог с людьми, личностных свойств операторов, программистов, инженеров и других лиц, активно задействованных в диалоге с техническими интеллектуальными системами;

7) развитие у лиц, работающих в системах «человек-система искусственного интеллекта», социально-перцептивного интеллекта и навыков общения, диалогического взаимодействия в группах личностного роста, психодраматических, арт-терапевтических, гештальт-группах и других личностно- и коммуникативно-центрированных группах;

8) коррекция личностных качеств в случае развития профессиональной деформации путем индивидуальных психологических консультаций и групповой психокоррекционной работы [Яценко, 2002] .

Выводы

В результате психологического анализа диалога "человек – система искусственного интеллекта" было установлено, что при длительном взаимодействии человека с компьютером у пользователя может сформироваться ряд психологических феноменов, которые осложнят налаживание отношений субъекта с другими людьми. В психике человека могут появиться такие психологические эффекты, как: "сужение диапазона эмоций", "размывание (или наоборот - повышение жесткости) коммуникативных границ", "трудности в понимании психологического контекста диалога человека с человеком", "редукция интуиции в коммуникативной сфере". Испытывая коммуникативные трудности, пользователь склонен скорее на неосознанном уровне зафиксироваться на взаимодействии с "системой искусственного интеллекта", нежели предпринимать попытки построения удовлетворяющих отношений с людьми. Таким образом, круг контактов человека замыкается, жизненное взаимодействие превращается в "хождение по ленте Мёбиуса", что способствует уходу личности от реального диалогически-партнерского, целостного в эмоциональном и ценностном аспектах общения с людьми, в виртуальное монологически-когнитивное взаимодействие с системой искусственного интеллекта. Достижения современной практической психологии в сфере коррекции и развития коммуникативных качеств человека открывают новые возможности для разработки психопрофилактических мер в направлении оптимизации диалога человека с системами искусственного интеллекта.

Литература

[Кристенсен, 1991] Человеческий фактор. В 6-ти тт. Т.1. Эргономика – комплексная научно-техническая дисциплина: Пер. с англ. / Ж. Кристенсен, Д. Мейстер, П. Фоули и др. – М.: Мир, 1991. – 599 с.

[Журавлев,1989] Журавлев А. П., Павлюк Н. А. Язык и компьютер. – М. Просвещение, 1989. – 159 с.

[Сергиенко, 2005] Сергиенко И. М. Характеристика диалогического взаимодействия в системе "психолог – клиент" // Матеріали Міжнародної науково-практичної конференції "Дні науки 2005". Т. 31. Психологія та соціологія. – Дніпропетровськ: Наука і освіта, 2005. – С. 51 – 54.

[Яценко, 2002] Яценко Т.С., Кмит Я.М., Алексеенко Б.Н. Активное социально-психологическое обучение: теория, процесс, практика: Учебное пособие. – Хмельницкий: Издательство НАПВУ; – Москва: Издательство СИП РИА, 2002. – 792 с.

Информация об авторе

Ирина Сергиенко *– доцент кафедры практической психологии Черкасского национального университета. Рабочий адрес: Кафедра практической психологии, Черкасский национальный университет, 18031, г. Черкассы, бул. Шевченко, 81. Домашний адрес: ул. Хрещатик, д.55, кв.107, г.Черкассы, 18031. e-mail: serhyenko@ukr.net*

КОМПЬЮТЕРНАЯ СИСТЕМА ВИРТУАЛЬНОГО ОБЩЕНИЯ ЛЮДЕЙ С ПРОБЛЕМАМИ СЛУХА

Юрий Крак, Александр Бармак, Александр Ганджа, Антон Тернов, Николай Шатковский

***Аннотация**: В статье представлена комплексная информационная технология для безсловесной коммуникации между людями с проблемами слуха, основанная на жестомимическом языке.*

***Ключевые слова**: симулация, язык жестов, компютерная система.*

***ACM Classification Keywords**: I.2.8 Problem Solving, Control Methods, and Search H.1.1 Systems and Information.*

***Conference**: The paper is selected from XIVth International Conference "Knowledge-Dialogue-Solution" KDS 2008, Varna, Bulgaria, June-July 2008*

Введение

В мире количество полностью глухих людей и людей с тяжелыми формами потери слуха составляет около 1,5% от общей численности населения – это десятки миллионов людей, для которых необходимо создавать средства равноценного общения в обществе. Пункт 7 Правила 5 Приложения к Резолюции ООН 48/96 "Стандартные правила обеспечения равных возможностей для инвалидов" гласит: "Необходимо обеспечить, чтобы язык жестов применялся для обучения глухих детей, в их семьях и сообществах. Необходимо также обеспечить услуги сурдоперевода для того, чтоб способствовать общению глухих с другими людьми".

В Украине больше полумиллиона детей с недостатками слуха, а количество глухих людей, для которых необходимо разрабатывать современные средства обучения и общения соответственно к мировому научно-техническому развитию – составляет миллионы. Развитие современной науки, компьютеризация общества, использование мультимедийных и Интернет технологий создали достаточные условия для разработки компьютерных систем коммуникации этих людей в формах и образах близких и понятных для них и для окружающего мира.

Основной формой общения глухих есть язык жестов. Язык жестов имеет национальные особенности (например, английский, французский, украинский, русский и другие языки), причем люди с недостатками слуха используют при общении два жестовых языка, которые имеют разную грамматику и разный набор жестов:

- разговорный язык жестов, который используется в повседневном общении и имеет собственную грамматику, достаточно отличающуюся от естественного разговорного языка;
- калькулирующий язык жестов, который используется в официальной и деловой обстановке и содержит в себе как знаки разговорного языка жестов, так и знаки дактильной азбуки, которая служит для отображения слов по буквах, причем калькулирующий язык жестов не имеет своей грамматики, он подчиняется грамматическим правилам национального языка.

Важной проблемой общения глухих с другими людьми есть умение распознать разговорный язык по губам. С этой точки зрения задачи визуального синтеза языка и распознавания по губам есть альтернативой языкового общения для людей с проблемами слуха. Кроме этого, развитие направления автоматического чтения по губам поможет улучшить показатели существующих систем распознавания

языка благодаря получению дополнительного независимого канала информации, а синтезированное озвучивание текстов позволит включить в коммуникационный процесс и людей с проблемами зрения.

В статье рассматривается комплексная информационная технология невербального общения людей с проблемами слуха, как между собой, так и с другими людьми.

Информационная технология невербального общения людей с недостатками слуха

После проведенного анализа структуры языка жестов (на основе анализа американского жестового языка (ASL) [Stokoe, 1960 , Stokoe, 2001]), современного состояния проблемы общения людей с ограниченным слухом, была разработана концепция комплексной информационной технологии.

Комплексная информационная технология включает реализацию следующих возможностей:

- модуль перевода обычного текста на язык жестов глухих, который содержит: анимацию процесса проговаривания разговорного и калькулирующего языка жестов с использованием виртуальных трехмерных моделей людей;
- анимацию мимики лица модели (с учетом эмоциональных составляющих) при проговаривании;
- озвучивание (синтез) обычного текста в его звуковой аналог (с использованием разных голосов, с реализацией функций «громче/тише», «дальше/ближе»);
- модуль распознавания по изменению мимики губ текста, который проговаривается.

Концепция экспериментальной технологии виртуального общения людей с проблемами слуха представлена на схеме (рис. 1).

Рис. 1. Схема концепции экспериментальной технологии виртуального общения глухих людей

В рамках этих исследований была реализована модель синтеза украинского языка жестов с использованием технологии Microsoft Agent. Был создан агент, который продуцирует язык жестов, используя некоторое множество жестов. Полученный агент может быть интегрирован как в офисные, так и в Web-приложения (Рис. 2).

Рис. 2. Демонстрация моделирования языка жестов с помощью технологии Microsoft Agent

С использованием определенного множества жестов было реализовано приложение для мобильного устройства (рис. 3)

Рис.3. Приложение для мобильного устройства

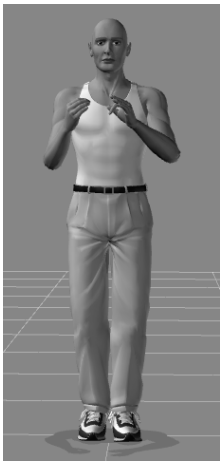


Рис.4. Трехмерная модель генерирующая украинский язык жестов

Для синтеза трехмерной анимации языка жестов построены геометрические классы векторов-образов жестов. Построение этих классов базировалось на использовании технологии motion capture [Menache, 2000]. Motion capture – это технология представления движений, позволяющая перейти от характеристик движения в реальном мире, к фиксации изменений параметров для математической модели. Для автоматического получения необходимых ключевых координат жеста использовалась технология трекинга [Avidan, 2001].

Для представления жеста использовался формат BVH, с последующим экспортом полученных данных на скелетную трехмерную модель (например, в модуле Character Studio для 3D studio MAX или в Poser) .

Предложенная реализация технологии motion capture для фиксации движений жестового языка включает в себя (рис. 5):

1. Получение видео-потока жестикуляции в двух ракурсах: фронт, профиль (все необходимые параметры съемки известны).
2. Обработка видео-потока с выделением следов и положений (координат) рук жестикуляции в пространстве.
3. На основе полученных координат жестикуляции - формирование BVH-файла для синтеза трехмерной анимации.
4. Применение BVH-файла для создания анимации (в Character Studio для 3D studio MAX или в Poser).

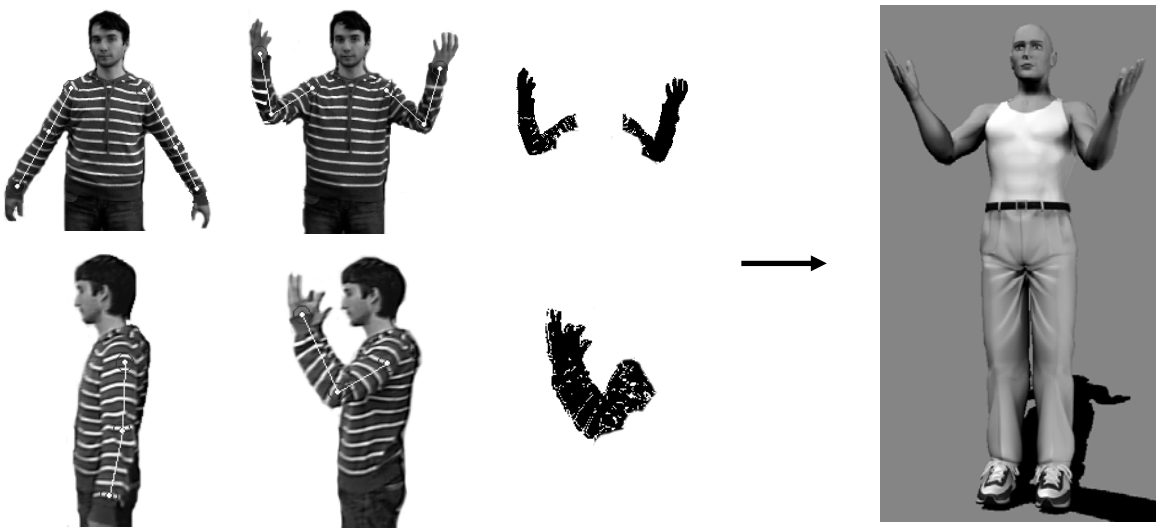


Рис. 5. Построение анимации жеста

Для обработки входного текста (расстановки ударений, выделения инфинитивов, поиска синонимов и типовых фраз языка) разработана информационная модель украинского языка. Модель представлена в виде таблиц реляционной базы данных с набором сохраненных процедур, которые реализуют необходимую, для рассматриваемой технологии, функциональность (рис. 6).

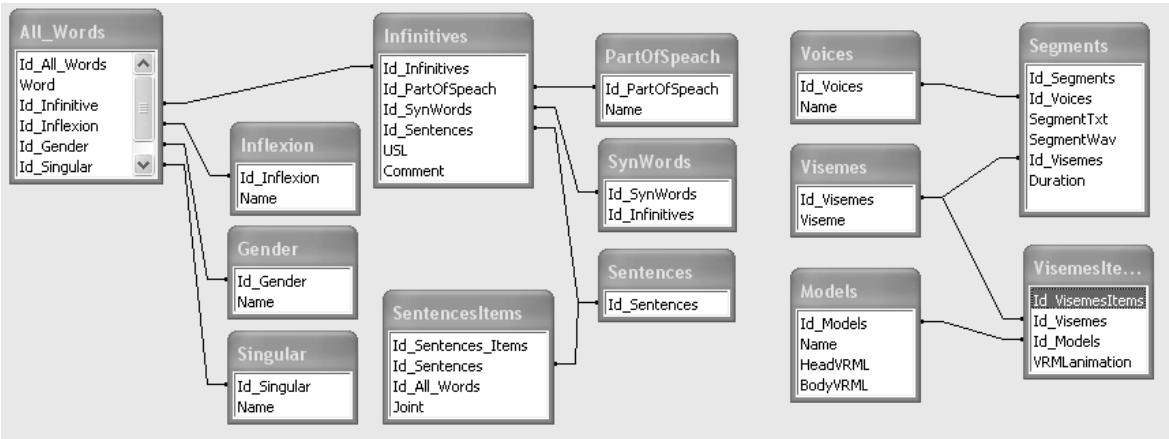


Рис. 6. Логическая схема базы данных – информационная модель украинского языка.

Информационная модель украинского языка содержит более двух с половиной миллионов слов (все возможные словоформы), ударения, синонимы, идиомы, фрагменты слов для синтеза разными голосами, вектора-образы жестов.

Для реализации функции визуализации озвучивания текста разработан синтезатор украинского языка. Синтезатор позволяет создавать голосовой аналог произвольного текста различными голосами с возможностью управления характеристиками голоса (громкость, дальше/ближе). Синтезатор также дает возможность построить визуальное представление процесса проговаривания (как с помощью двумерных визем, так и на трехмерной модели) (рис. 7,8).

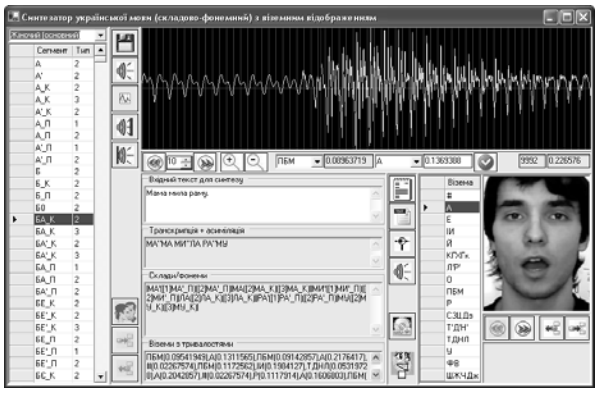


Рис. 7. Синтезатор украинского языка

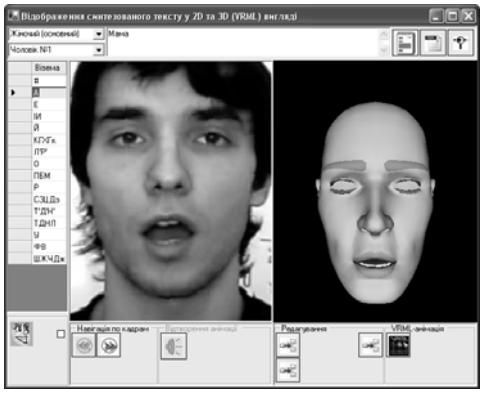


Рис. 8. Синтез визем украинского языка

Выводы

Дальнейшие исследования направлены на реализацию всего множества жестов украинского языка жестов с возможностью визуализации на трехмерных моделях (рис. 4). Создается информационная технология и соответствующее программное обеспечение для реализации приведенной концепции компьютерной системы виртуального общения людей с проблемами слуха.

Библиография

[Stokoe, 1960] Stokoe, W. Sign language structure: An outline of the visual communication systems of the American Deaf. Studies in linguistics, occasional papers 8. Silver Spring, MD: Linstok Press., 1960. - 94 p.

[Stokoe, 2001] Stokoe W. C. Language in Hand: Why Sign Came Before Speech. Washington, DC: Gallaudet Univ.Press., 2001. - 227p.

[Menache, 2000] Menache A.,Understanding Motion Capture for Computer Animation and Video Games,Morgan Kaufmann, 2000. - 238 p.

[Avidan, 2001] S. Avidan, “Support vector tracking,” in Proc. IEEE Conf. on Computer Vision and Pattern Recognition, Kauai, Hawaii, volume I, 2001. - P. 184–191.

Authors' Information

Yuriy Krak – *The Institute of Cybernetics of National Academy of Science of the Ukraine, the senior scientist, address: 40 Glushkov ave., Kiev, Ukraine, 03680; e-mail: krak@unicyb.kiev.ua*

Alexander Barmak – *The Institute of Cybernetics of National Academy of Science of the Ukraine, the senior scientist, address: 40 Glushkov ave., Kiev, Ukraine, 03680; e-mail: barmak@svitonline.com*

Alexander Gangha – *The Institute of Cybernetics of National Academy of Science of the Ukraine, the engineer, address: 40 Glushkov ave., Kiev, Ukraine, 03680*

Anton Ternov – *The Institute of Cybernetics of National Academy of Science of the Ukraine, the junior scientist, address: 40 Glushkov ave., Kiev, Ukraine, 03680*

Nikolai Shatkovskii – *The Institute of Cybernetics of National Academy of Science of the Ukraine, the junior scientist, address: 40 Glushkov ave., Kiev, Ukraine, 03680*

МУЛЬТИАГЕНТАЯ СИСТЕМА ДЛЯ ИНТЕЛЛЕКТУАЛЬНОГО АНАЛИЗА ДОКУМЕНТОВ

Вячеслав Ланин

***Аннотация**: В статье представлены промежуточные результаты реализации комплексного подхода к разработке подсистемы управления электронными документами в CASE-системе METAS, предназначенной для создания распределенных информационных систем, допускающих динамическую настройку на меняющиеся условия эксплуатации и потребности пользователей. Предлагается значительно увеличить эффективность работы с электронными документами за счет их автоматизированного интеллектуального анализа. В предлагаемом решении для анализа документов используются агентный и онтологический подходы. Онтологии позволяют в явном виде представить семантику и структуру документа. Использование агентов позволяет упростить процесс анализа, сделать его расширяемым и масштабируемым. Результаты интеллектуального поиска и обработки документов, получаемых из гетерогенных источников, могут быть использованы не только для автоматической классификации и каталогизации документов в информационной системе в удобной для пользователя форме, но и для снижения трудоемкости выполнения этапа анализа предметной области информационной системы, ее проектирования, а также для интеллектуализации процессов создания отчетных документов на основе информации, размещенной в базе данных системы.*

***Ключевые слова**: онтология, агент, мультиагентные системы, интеллектуальный поиск, анализ документов, адаптируемые информационные системы, CASE-технология.*

***ACM Classification Keywords**: D.2 Software Engineering: D.2.2 Design Tools and Techniques – Computer-aided software engineering (CASE); H.2 Database Management: H.2.3 Languages – Report writers; H.3.3 Information Search and Retrieval – Query formulation.*

***Conference**: The paper is selected from XIVth International Conference "Knowledge-Dialogue-Solution" KDS 2008, Varna, Bulgaria, June-July 2008*

Введение

К настоящему времени разработано большое количество CASE-систем, автоматизирующих наиболее трудоемкие этапы разработки информационных систем (ИС), связанные с программированием бизнес-операций и созданием интерфейса. Самым продолжительным и трудоемким становится *этап анализа предметной* области, который обычно не автоматизируется CASE-системами. Таким образом, одним из перспективных направлений развития CASE-систем является автоматизация этого процесса. В CASE-системах, ориентированных на создание ИС с динамической адаптацией во время их использования, где стадия анализа предметной области «растягивается» на все время функционирования системы, эта задача становится особенно актуальной. Если учесть, что на стадии эксплуатации таких систем задача реинжиниринга возложена (хотя бы частично) на пользователей – специалистов в предметных областях, но не в области информационных технологий, то средства автоматизации анализа становятся важнейшими компонентами. Другими словами, если ставить задачу динамической настройки информационной системы на меняющиеся условия, то основа реализации средств ее динамической адаптации – средства реструктуризации данных в базе данных (БД) ИС. А эти средства позволяют вносить изменения в модель данных на основе результатов анализа предметной области, нормативно-справочных и распорядительных документов, регламентирующих деятельность в этой области. Отсюда следует необходимость поддержки в динамически адаптируемых системах одного из самых сложных и

трудоемких этапов разработки ИС – этапа анализа. Источником информации для анализа могут служить документы различного вида, т.к. деятельность любой бизнес-системы строится именно на основе нормативных документов. Поддержка бизнес-операций средствами ИС требует отражения в модели данных системы норм, закрепленных в нормативно-справочных данных, распорядительных документах, в виде ограничений, налагаемых на данные (атрибуты, свойства объектов предметной области, информация о которых хранится в БД, а также связи между ними) и операции, выполняемые над ними [1].

В результате анализа должна быть построена *система взаимосвязанных документов*:

- относящихся к определенным направлениям деятельности бизнес-системы (к определенным понятиям, объектам предметной области);
- отражающих связи между этими понятиями (с каждым понятием может быть связан документ или совокупность документов, связи между документами отражают связи между понятиями);
- содержащих нормативную информацию, которая также может быть выделена на основе анализа содержания документов.

На основе построенной системы взаимосвязанных документов можно частично автоматизировать процесс анализа изменений предметной области и внесения изменений в модель предметной области ИС (т.е. реализовать поддержку процесса разработки и адаптации ИС). Таким образом, система управления документами становится не только «надстройкой» над ИС и ее БД, позволяющей получать результаты обработки данных, хранящихся в БД ИС, в удобной для пользователей форме, но и становится основой средств разработки ИС – средств реструктуризации данных.

Описание документов с помощью онтологий

Для повышения эффективности обработки электронный документ требует наличия метаданных, описывающих структуру и семантику данных. Одним из возможных подходов к описанию информации, заложенной в документе, является подход на основе онтологий. Под онтологией понимается база знаний специального типа, которая может «читаться» и пониматься, отчуждаться от разработчика и/или физически разделяться ее пользователями [4]. Онтологический подход обладает такими преимуществами, как

- удобство восприятия человеком;
- отсутствие необходимости в специальной квалификации пользователя при разработке онтологии;
- возможность описания одного документа различными онтологиями.

В качестве подхода к решению описанной выше задачи был выбран онтологический подход [1], в котором онтология описывает как структуру, так и содержание документа. В соответствии с предлагаемым подходом *онтология используется для описания семантики данных документа и его структуры*. Учитывая специфику решаемых в данной работе задач, конкретизируем понятие онтологии: будем считать, что *онтология – это спецификация некоторой предметной области*, которая включает в себя словарь терминов (понятий) предметной области и множество связей между ними, которые описывают, как эти термины соотносятся между собой в конкретной предметной области.

Для построения иерархии понятий онтологии используются следующие базовые типы отношений:

- “is_a” («экземпляр – класс», гипонимия);
- “part_of” («часть – целое», меронимия);
- “synonym_of” (синонимия).

Следует учесть, что данные типы отношений являются базовыми и не зависят от онтологии, но необходимо предоставить пользователю возможность добавления новых отношений, которые бы учитывали специфику описываемой предметной области.

Ректору Иванову И.И.
студентки Сидоровой А.А.

заявление.

Прошу освободить меня от занятий 10.02.08 для участия в спортивных соревнованиях.

10.02.2008 Сидорова А.А.

а)

б)

Рис. 1. Пример простого документа «Заявление» (а) и онтологии, описывающей класс документов «Заявление» (б)

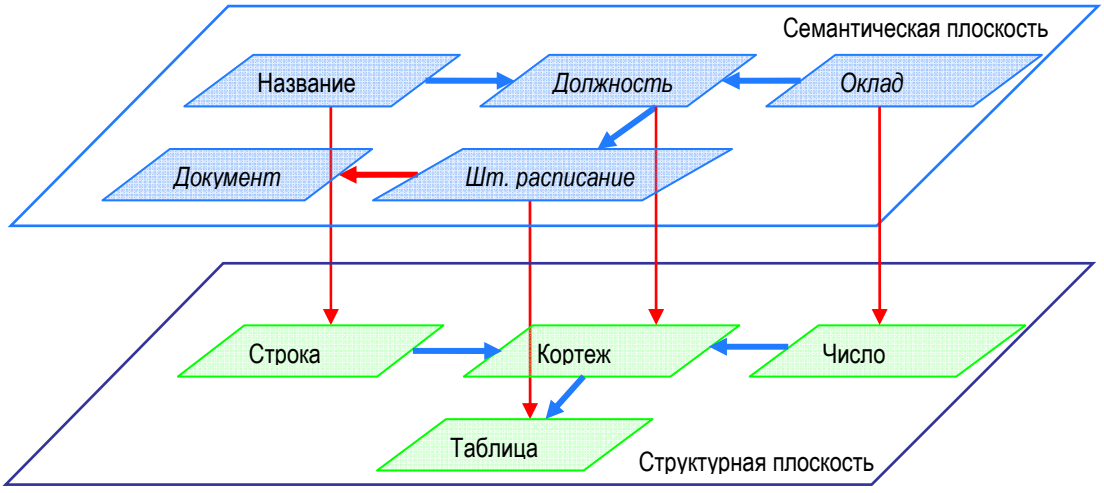
Приказ № 1
от 01.11.2005

Установить с 01.12.2005 следующее штатное расписание:

Ассистент	4 000 рублей
Старший преподаватель	6 000 рублей
Доцент	10 000 рублей
Заведующий кафедрой	15 000 рублей

Ректор Иванов И.И.

а)



б)

Рис. 2. Пример документа «Приказ» (а) и разбиение вершин онтологии для документа на две плоскости (б)

Кроме отношений онтология включает в себя два типа вершин. К первому типу отнесем вершины, описывающие структуру документа. Например: таблица, дата, должность и т.д. (они представляют собой общие понятия, не зависящие от конкретной предметной области). Другим типом будут являться вершины, содержащие понятия документа. Первый тип вершин будем называть *структурные вершины*, второй тип – *семантический вершины*. На рис. 1 структурные вершины имеют темный оттенок, а семантические вершины изображены более светлым оттенком.

Фактически в данном контексте *онтология – это иерархическая понятийная основа рассматриваемой предметной области*. Онтология документа используется для анализа документа, благодаря ей из документа можно получить требуемые данные: известно, где искать данные и как они могут быть интерпретированы.

Если представлять документ с использованием онтологий, то задача сопоставления онтологии и имеющегося документа сводится к задаче поиска понятий онтологии в документе. Как следствие, системе необходимо ответить на вопрос: описывает ли данная онтология документ или нет. На последний вопрос можно ответить утвердительно, если в процессе сопоставления в документе были найдены все понятия, включенные в онтологию. Прежде, чем производить поиск вершин, содержащих понятия документа, необходимо провести поиск вершин, описывающих структуру документа. Таким образом, исходная задача сводится к задаче поиска в тексте документа общих понятий на основе формальных описаний.

В приведенном примере (рис. 2, б) вершины онтологии разбиты на две плоскости, что учитывается при сопоставлении документа (рис. 2, а) и его онтологии.

Агентный подход к анализу документов

К процессу поиска документов предъявляется ряд требований:

- высокая скорость обработки больших объемов данных;
- отказоустойчивость;
- масштабируемость и
- настраиваемость на потребности пользователей и меняющиеся условия.

Для решения проблемы выделения общих понятий на основе формальных описаний предлагается агентный подход [2]. Здесь *под агентом понимается система, направленная на достижение определенной цели, способная к взаимодействию со средой и другими агентами* [3]. Данный подход будет удовлетворять требованиям, предъявляемым к процессу поиска, если при построении системы будут реализованы все преимущества мультиагентных систем.

При использовании данного подхода *для каждой вершины онтологии, содержащей общее понятие, создается агент*, *который проводит поиск данного конкретного понятия*. Для признания агента интеллектуальным необходимым условием является *наличие у него базы знаний*. Таким образом, чтобы определить агентов, действующих в системе, необходимо выбрать способ для описания базы знаний (БЗ), характер взаимодействия со средой и сотрудничества. Средствам представления базы знаний агентов посвящен следующий раздел статьи.

Одним из важнейших свойств агентов является *социальность или способность к взаимодействию* [2]. Как было сказано ранее, для каждой вершины онтологии, содержащей общее понятие (семантическая вершина), создается агент. Согласно принятой классификации агентов он является *интенциональным*.

Данный агент нацелен на решение двух задач:

1. Весь имеющийся список шаблонов понятия он разбивает на отдельные компоненты и запускает более простых агентов для поиска структурных вершин.
2. Производит сборку результатов из всех списков, полученных агентами более низкого уровня.

Упомянутые выше агенты более низкого уровня являются *рефлекторными*. Они получают шаблон, и их целью становится отыскание в тексте фрагментов, попадающих под этот шаблон.

Важным вопросом становится коммуникация агентов. *Механизмы коммуникации агентов* делятся на непосредственные и опосредованные. Примером реализации *непосредственной коммуникации* может служить модель взаимодействия «*заказчик – подрядчик*» (*contract network*). Механизм *опосредованной коммуникации* реализуется с помощью архитектуры «*доски объявлений*» (*blackboard*):

- Модель «заказчик – подрядчик». Данная модель предполагает деление всего множества агентов системы на два класса – класс заказчиков и класс подрядчиков. Суть данной модели взаимодействия заключается в решении различных задач путем направления их на выполнение наиболее подходящим для этого агентам. За распределение задач ответственны агенты – заказчики. Потенциальные подрядчики анализируют выставленные заказчиками заявки, анализируют их на предмет возможности реализации и, в случае положительного результата анализа, подают заявку заказчику.
- Модель «доска объявлений». Blackboard-архитектура основана на модели классной доски, на которой представлено текущее состояние системы, в рамках которой оперируют агенты. Агенты постоянно анализируют информацию на доске, пытаясь найти применение своим возможностям. В случае если в некоторый момент времени агент обнаруживает возможность внесения своего вклада в процесс решения текущих задач, он оставляет на доске информацию о начале работы в данном направлении, а по окончанию работы помещает результат на доску.

Учитывая особенности решаемой задачи, реализована комбинация двух моделей коммуникации «заказчик – подрядчик» и «доски объявлений».

Архитектура мультиагентной системы и процесс анализа документа представлены на рис. 3.

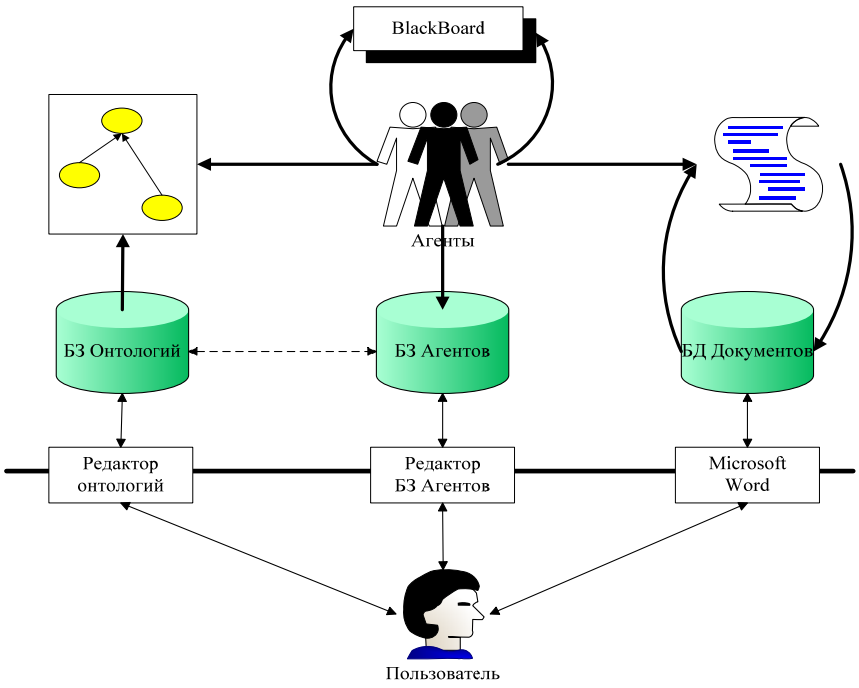


Рис. 3. Архитектура системы SemanticDoc

Представление базы знаний агентов

Одним из наиболее важных вопросов в системе является вопрос представления БЗ агента. К настоящему моменту представление БЗ агента возможно тремя различными способами: с использованием онтологий, с помощью регулярных выражений и на базе продукций.

Преставление знаний агента с помощью онтологии – наиболее выразительный способ, использующий все преимущества явного представления знаний (рис. 4). Достоинством данного способа является то, что для «доказательства» вершины онтологии мы можем применить различные средства. Например, это может быть простое совпадение ключевой фразы или обращение к БД ИС. Онтологии позволяют описать различные ситуации в случае, если не удается найти точное соответствие. Мы может найти обобщающее или конкретизирующее понятие и т.п.

Содержимое анализируемого документа представлено в виде специальной *объектной модели*, за основу которой была взята объектная модель документа Microsoft Word. Для доступа к этой объектной модели разработаны API-функции, позволяющие оперировать одинаковыми понятиями при работе с документами в различных форматах. В состав API-функций включены функции по синтаксическому разбору приложений, функции для вычисления различных метрик между понятиями, функции для извлечения информации о структуре документа. Если для поиска понятия вершины необходимы дополнительные действия, они могут быть описаны с помощью скрипта с использованием упомянутых выше API-функций. В скрипте также могут быть использованы обращения к объектной модели самой ИС.

Вторым подходом является подход с использование *регулярных выражений*. Последние позволяют легко учитывать различные формы слова и работать с большими объемами информации [5]. Однако необходимо учитывать, что иногда, особенно для неквалифицированных пользователей, задача правильного построения регулярного выражения становится достаточно сложной. С целью ее упрощения предполагается наличие в системе специального редактора, позволяющего работать с регулярными выражениями на естественном языке. Например, эквивалентом к «\d{5}» является «пятизначное число» и т.д. Кроме того, желательна реализация функции построения регулярного выражения «по образцу». Это означает, что по примерам, приведенным пользователем, возможно автоматическое построение регулярного выражения. Например, пользователь в качестве примеров предложил две даты: «1.12.08» и «15.07.2006». Система должна построить регулярное выражение, которое бы соответствовало обоим форматам представления дат: «(\d{1,2}).(\d{1,2}).((\d{4})|(\d{2}))».

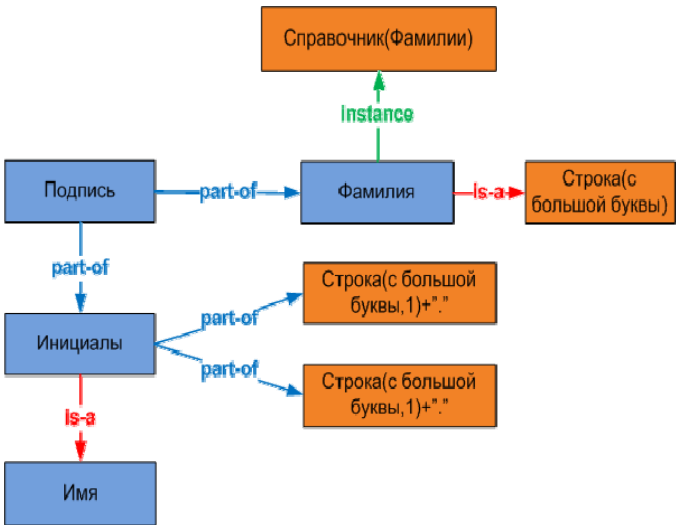


Рис. 4. Представление базы знаний агента с помощью онтологии

Недостатком регулярных выражений является то, что при поиске они не позволяют учитывать местонахождение искомого слова/фразы. Для устранения данного недостатка возможно совместное использование регулярных выражений и *правил продукционного типа*, которые являются третьим способом представления БЗ агента.

Продукции в основном используются для анализа структуры документа. Введены специальные понятия, которые могут быть использованы при задании условий. Например, правило находящее заголовки в тексте может быть сформулировано следующим образом:

> ***Если*** *(шрифт абзаца отличен от абзаца до и абзаца после) и (абзац выровнен по центру),*
> ***то*** *данный абзац является заголовком.*

Заключение

На данный момент результатом работы стала реализация на платформе .NET системы SemanticDoc, представляющей собой мультиагентную систему, которая проводит сопоставление документа и онтологии.

В информационном поиске для сравнения качества результатов были введены две характеристики: *точность* (*precision*) и *полнота* (*recall*) [6]. Подобные характеристики можно ввести и для системы сопоставления документа и онтологии. Под *точностью* (P) будем понимать долю правильно проведенных соответствий документа и онтологии по отношению ко всем сделанным системой соответствиям. Под *полнотой* (R) – долю правильно проведенных соответствий по отношению ко всем соответствиям документа и онтологии.

Пусть N – число существующих соответствий между документом и онтологией, M – число проведенных системой сопоставлений, A – число правильно проведенных системой сопоставлений. Тогда:

$$P = A/M \quad \text{и} \quad R = A/N.$$

Обычно эти два критерия «конфликтуют» и на практике стопроцентная точность и полнота недостижимы.

Работы по оценке пока не проводились, следующим этапом исследование станет оценка величин P и R при проведении экспериментов на реальных документах.

Средства анализа документов могут быть использованы как для снижения трудоемкости работы пользователей с документами, так и для поддержки решения задачи анализа предметной области разработчиками. В данном случае предлагается глубокая интеграция функциональных подсистем, включающих как средства разработки, так и средства, с которыми работают «конечные пользователи». Это дает возможность создания CASE-технологии, предназначенной для создания динамически настраиваемых ИС, обладающих уникальными возможностями адаптации к меняющимся условиям эксплуатации на основе «обратной связи» и интеллектуального анализа документов.

В рамках данной работы разрабатывается также формальная модель электронного документа и онтологии применительно к решаемой задаче, а на ее основе уточняется существующая объектная модель ИС, метаданных и алгоритмы управления документами.

Благодарности

Работа выполнена при поддержке гранта РФФИ № 08-07-90006-Бел_а.

Библиографический список

[1] Ланин В.В. Интеллектуальное управление документами как основа технологии создания адаптируемых информационных систем // Труды международной научно-технической конференций «Интеллектуальные системы» (AIS'07). Т. 2 / М.: Физматлит, 2007. С. 334-339.

[2] Тарасов В.Б. От многоагентных систем к интеллектуальным организациям: философия, психология, информатика. М.: Эдиториал, УРСС, 2002.

[3] Рассел С. Искусственный интеллект: современный подход. М.: Издательский дом «Вильямс», 2006.

[4] Хорошевский В.Ф., Гаврилова Т.А. Базы знаний интеллектуальных систем. СПб.: Питер, 2001.

[5] Фридл Дж. Регулярные выражения. СПб.: Питер, 2003.

[6] Weal M.J., Kim S., Lewis P.H., Millard D.E., Sinclair P.A.S., De Roure D.C., Nigel R. Ontologies as facilitators for repurposing web documents / Shadbolt. Southampton, 2007.

Сведения об авторе

Вячеслав Ланин *– Пермский государственный университет, аспирант кафедры математического обеспечения вычислительных систем; Россия, г. Пермь, 614990, ул. Букирева, д. 15; e-mail: lanin@psu.ru*

МОДЕЛИРОВАНИЕ МНОГОМЕРНЫХ ДАННЫХ В СИСТЕМЕ METAS BI-PLATFORM

Павел Мальцев

***Аннотация**: Представлено формальное описание многомерной модели данных, реализованной в программном комплексе METAS BI-Platform. В статью включено описание объектов многомерной модели (измерений и множеств измерений и т.д.), их свойств и организации, а также операций, выполняемых над ними. Описаны методы агрегации многомерных данных, позволяющие эффективно агрегировать массивы числовых показателей. Программный комплекс METAS BI-Platform предназначен для многомерного анализа данных, получаемых из гетерогенных источников, и позволяет упростить разработку BI-приложений. Программный комплекс представляет собой многоуровневое приложение с архитектурой «Клиент-сервер». Каждый уровень комплекса соответствует степени абстракции данных. На самом низком уровне расположены драйверы доступа к специфическим физическим источникам данных. Следующий уровень – уровень виртуальной СУБД, позволяющей осуществлять унифицированный доступ к данным, что избавляет от необходимости учитывать специфику конкретных СУБД при разработке BI-приложений. Реализован программный интерфейс комплекса (API). В распоряжение разработчиков предоставляется набор готовых компонентов, которые могут быть использованы при создании BI-приложений. Это позволяет разрабатывать на основе комплекса BI-приложения, отвечающие современным требованиям, предъявляемым к подобным системам.*

***Ключевые слова**: Business Intelligence, BI, бизнес-анализ, OLAP, системы поддержки принятия решений, DSS, модель многомерных данных, многомерный анализ.*

***ACM Classification Keywords**: H.2 Database Management: H.2.1 Logical Design – Data models, H.2.4 Systems – Distributed databases; H.4 Information Systems Applications: H.4.2 Types of Systems – Decision support (e.g., MIS).*

***Conference**: The paper is selected from Sixth International Conference on Information Research and Applications – i.Tech 2008, Varna, Bulgaria, June-July 2008*

Введение

В настоящее время всё более широкое применение находят так называемые средства Business Intelligence (BI), которые позволяют облегчить процесс принятия решений за счёт получения необходимых количественных характеристик, являющихся результатом обработки больших объёмов данных, и применения математических методов анализа этих характеристик с целью выявления закономерностей. Естественно, решения принимаются человеком, средства Business Intelligence способны лишь «дать рекомендации», помочь обосновать принимаемые решения.

Работа BI-приложений, как правило, основана на анализе больших объёмов информации, чем больше объём анализируемых данных, тем выше доверие к результатам. Данные для анализа зачастую берутся из реляционных баз данных, но работа с данными в BI-средствах обладает определённой спецификой и реляционная модель данных не всегда отвечает им. Одной из наиболее подходящих для Business Intelligence моделью данных на сегодняшний день является многомерная модель.

В данной статье приводится формальное описание многомерной модели данных, реализованной в программном комплексе METAS BI Platform.

Программный комплекс METAS BI-Platform

Программный комплекс METAS BI-Platform предназначен для проведения многомерного анализа данных, получаемых из гетерогенных источников. Использование данного средства позволит облегчить процесс создания BI-приложений за счёт того, что в комплексе уже реализованы модули сбора и многомерного анализа данных, и разработчик может свободно использовать функции данных модулей.

С точки зрения архитектуры программный комплекс представляет собой многоуровневое клиент-серверное приложение (рис. 1). Это позволяет разрабатывать на основе комплекса BI-приложения, отвечающие современным требованиям, предъявляемым к подобным системам. С другой стороны, архитектуру METAS BI-Platform можно охарактеризовать как иерархическую – модули комплекса разбиты на уровни и модули более высоких уровней используют функции модулей более низких уровней. На рис. 1 схематично представлена архитектура комплекса.

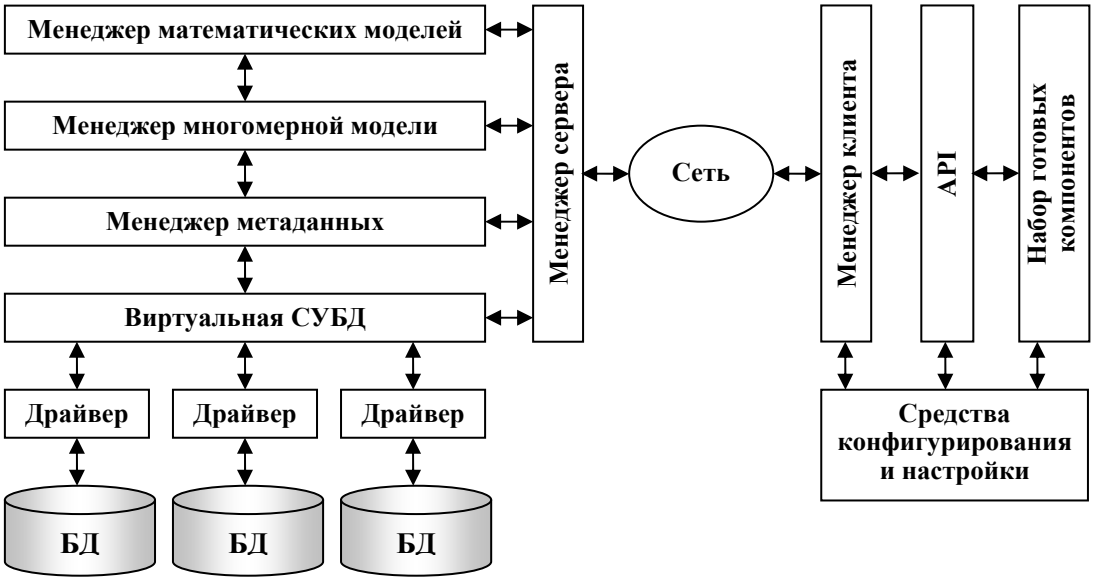


Рис. 1. Архитектура комплекса METAS BI-Platform

Каждый уровень комплекса соответствует степени абстракции данных. На самом низком уровне расположены *драйверы доступа к специфическим физическим источникам данных*. Следующий уровень реализован в виде *виртуальной СУБД*, которая позволяет осуществлять унифицированный доступ к реляционным данным и избавляет модули вышестоящих уровней от необходимости учитывать специфику конкретных СУБД.

Менеджер метаданных осуществляет ведение *базы метаданных* (*репозитория*) комплекса и разработанных на его основе приложений. Метаданные в репозитории представляются с позиции объектного подхода.

В основе работы комплекса лежит технология OLAP. В соответствии с современными требованиями к средствам, реализующим эту технологию, данные должны представляться в многомерной модели. За реализацию этого представления отвечает *менеджер многомерной модели.* Описание многомерной модели, реализованной в METAS BI Platform, приведено ниже.

Иногда возникает потребность в построении некоторых математических представлений данных, получаемых из базы, с целью их анализа. Для решения этой задачи в комплексе реализован *менеджер математических моделей*, частично упрощающий реализацию математических методов анализа данных в конечных приложениях.

Для того чтобы разработчики BI-приложений могли использовать комплекс, реализован программный интерфейс комплекса (API), а также набор готовых компонентов, которые могут быть использованы при разработке BI-приложений.

Описание многомерной модели данных в METAS BI-Platform

Можно сказать, что в многомерной модели данные представляются в виде *вектор-функций* многих переменных: $f_i(x_1,...,x_n)$, где f_i – некоторые числовые показатели (например: объём продаж, сумма сделки), а x_i – параметры. *Параметры* организуются в виде *измерений*, имеющих иерархическую структуру. Будем считать, что *элемент измерения* является некоторым подмножеством множества допустимых значений соответствующего параметра. Мощность многомерной модели заключается в агрегации. На пример, если мы запросили у OLAP-средства общую сумму контрактов за 2006 г., то OLAP-средство само просуммирует суммы всех контрактов из БД, дата которых принадлежит 2006 г.

Объекты многомерной модели

Будем называть *множествами точек измерения* непустые множества строковых значений, а элементы этих множеств, соответственно, – *точками измерения*. Множества точек измерения будем обозначать X^i, где $i \in N$, а точки измерения из множества X^i будем обозначать x_j^i, где $j \in N$.

Рассмотрим некоторое множество точек измерения X, будем обозначать через $\widetilde{X}$ такое подмножество $\mathscr{P}(X)$, что:

$$\varnothing \in \widetilde{X} \tag{1}$$

$$X \in \widetilde{X} \tag{2}$$

$$(\forall x \in X) : \{x\} \in \widetilde{X} \tag{3}$$

$$((\forall X^1, X^2 \in \widetilde{X}) : (X^1 \bigcap X^2 = \varnothing) \vee (X^1 \bigcap X^2 = X^1) \vee (X^1 \bigcap X^2 = X^2) \tag{4}$$

Замечание: Из (4) видно, что подмножества X в $\widetilde{X}$ организованны в виде иерархии. Приведём пример: на рис. 2 изображена схема организации подмножеств измерений.

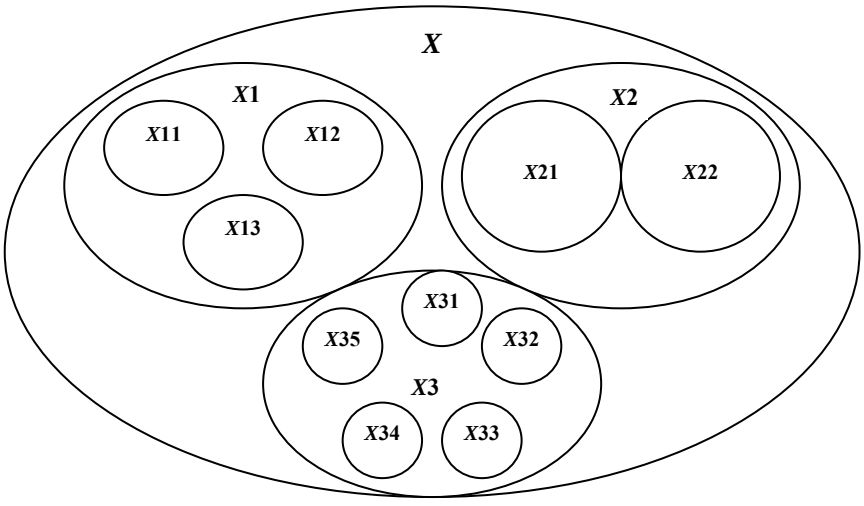


Рис 2. Схема организации подмножеств измерении

Такую организацию подмножеств можно интерпретировать как иерархию, которая изображена на рис. 3.

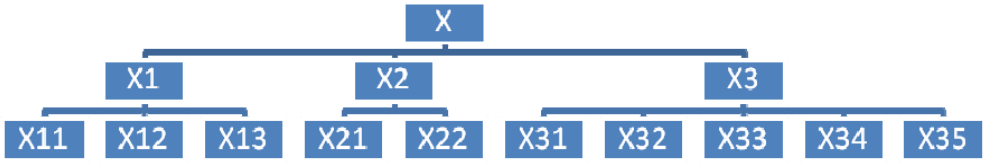


Рис. 3. Иерархия измерений

Пусть дано некоторое множество точек измерения X, для него построено измерение $\widetilde{X}$. Веткой иерархии измерения $\widetilde{X}$ для некоторой точки $x \in X$ будем называть такое подмножество $\widetilde{X}$ (обозначим его $Branch(\widetilde{X}, x)$), что

$$((\forall X^1 \in Branch(\widetilde{X}, x)) : x \in X^1) \wedge ((\forall X^2 \in \widetilde{X} \setminus X^1) : x^1 \notin X^2).$$

Длиной ветви иерархии $Branch(\widetilde{X}, x)\ x \in X$ будем называть величину *Card Branch*($\widetilde{X}$, *x*), т.е. количество подмножеств из $\widetilde{X}$, входящих в *Branch*($\widetilde{X}$, *x*).

Измерение $\widetilde{X}$ будем называть регулярным, если $(\forall x \in X) : Card\ Branch(\widetilde{X}, x) = k$, где $k \in N$. Договоримся понимать под глубиной иерархии некоторого измерения $\widetilde{X}$ величину $Depth(\widetilde{X}) = \max_{x \in X} Card\ Branch(\widetilde{X}, x)$. Очевидно, что если измерение регулярно, то глубина его иерархии равна длине любой его ветки.

Рассмотрим некоторое измерение $\widetilde{X}$ и построим группу множеств подмножеств:

$$L_0(\widetilde{X}) = \{\varnothing\}$$

$$L_i(\widetilde{X}) = \{X^1 \mid X^1 \in \widetilde{X} \setminus \bigcup_{j=0}^{i-1} L_j(\widetilde{X}), (\forall X^2 \in \widetilde{X} \setminus \bigcup_{j=0}^{i-1} L_j(\widetilde{X}) \mid X^1 \neq X^2) : (X^2 \subset X^1) \vee (X^1 \bigcap X^2 = \varnothing)\}$$

$$i = \overline{1, m}, m = Depth(\widetilde{X})$$

Множества $L_i(\widetilde{X})$ будем называть уровнями измерения $\widetilde{X}$, а i – номером уровня.

Будем называть набором измерений упорядоченное множество m измерений. Обозначать его будем следующим образом: $\breve{X} = \langle \widetilde{X}^1, \widetilde{X}^2, ..., \widetilde{X}^m \rangle$, $m \in N$. Элементом набора измерений, соответственно, будем называть m-ку $\breve{x} = \langle X^{11}, X^{21}, ..., X^{m1} \rangle$, где $X^{i1} \in \widetilde{X}^i$, $i = \overline{1, m}$.

Рассмотрим некоторую предметную область (ПрО). Пусть для этой предметной области у нас разработана некоторая OLTP система, будем обозначать её $OLTP($ ПрО, $data)$, где $data$ – набор данных. Объединим все возможные для ПрО наборы данных в множество D. Далее построим множество S, такое, что $Card\ S = Card\ D$ и каждому элементу из D поставим в соответствие один элемент из S. Элементы множества S будем называть состояниями предметной области ПрО, а само множество S – множеством состояний ПрО. При измерении набора данных будет меняться и состояние ПрО.

Событием будем называть изменение состояния предметной области. Понятно, что событие вызывается изменением набора данных. *Фактами* будем называть те события, записи о которых заносятся в базу данных (возможно, это те записи, которые и привели к событию), и мы можем их проанализировать. Факты будем обозначать строчными буквами греческого алфавита, кроме того, выделим особый факт,

назовём его нулевым фактом и договоримся обозначать его буквой o, договоримся также ни один другой факт так не обозначать.

Назовём *показателем факта* его числовую характеристику, т.е. по правилам нашей предметной области каждому факту поставлено в соответствие одно или несколько чисел. Обозначать показатели фактов будем следующим образом: $f_i(\alpha) \in R$, где α – некоторый факт, а $i \in N$. Договоримся, что если это специально не оговорено, то все показатели нулевого факта равны 0. Адресной функцией Adr, будем называть такое биективное отображение, что:

1. Прообразом этого отображения является множество $\left\{\left\langle x^1, x^2, ..., x^m \right\rangle \mid x^i \in X^i\right\}$, где $m \in N$, X^i – множество точек измерения.
2. Образом этого отображения является некоторое множество фактов.

В одной и той же предметной области могут рассматриваться факты различной природы. Факты, на возникновение которых по правилам данной предметной области следует реагировать одинаковым образом, будем объединять в *группы* (*классы фактов*). Классом фактов назовём такое множество фактов A, что:

1. $(\exists R)(\forall \alpha \in \mathrm{A}) : \alpha \in D(R)$, где R – некоторая реакция предметной области на некоторую группу фактов, $D(R)$ – область определения реакции предметной области (множество фактов которые составляют ту группу фактов, на которую распространяется определённая реакция предметной области).
2. Существует такая адресная функция, что её образ совпадает с A.
3. У фактов в классе одинаковый набор показателей, т.е. показатели фактов в рамках класса это отображения класса на множество действительных чисел.

Таким образом, класс фактов составляют не только те факты, записи о которых имеются в базе данных, но и все те, которые по правилам предметной области могут иметь место, и реагировать на них следует одинаково. Договоримся классы фактов обозначать прописными буквами греческого алфавита. Пусть имеется некоторый класс фактов A, через $\overline{\mathrm{A}}$ будем обозначать подмножество A такое, что в него входят только те факты, записи о которых уже имеются в БД.

Кубом будем называть множество $Cube(\overline{A}) = \overline{\mathrm{A}} \bigcup \{o\}$, где A – некоторый класс фактов. Кроме того, договоримся, что o имеет тот же набор показателей, что и факты из A.

Агрегирующей функцией назовём отображение, которое совокупности действительных значений M ставит в соответствие действительное значение: $\varphi(M) \in R$. Пару «показатель факта – агрегирующая функция» будем называть *параметром факта*. Параметры факта будем обозначать следующим образом: $F_i^{\alpha} =< f_j(\alpha), \varphi_k >$, где $i, j, k \in N$, α – некоторый факт, $f_j(\alpha)$ – показатель факта, φ_k – некоторая агрегирующая функция.

Адресная функция не позволяет получить агрегированные данные, а позволяет получить лишь конкретный факт, поэтому построим такое отображение, которое позволяло бы по элементу набора измерений получить агрегированные данные.

Пусть у нас имеется некоторый куб $Cube(\overline{\mathrm{A}})$, для которого мы построили адресную функцию $Fact_{\overline{\mathrm{A}}}$. Рассмотрим прообраз этой функции: $D(Fact_{\overline{\mathrm{A}}}) = \left\{< x^1, x^2, ..., x^m > \mid x^i \in X^i\right\}, m \in N$, по определению X^i – некоторое множество точек измерения. Для каждого множества точек измерения X^i построим измерение $\widetilde{X}^i$, получим набор измерений $\breve{X} = \left\langle \widetilde{X}^1, \widetilde{X}^2, ..., \widetilde{X}^m \right\rangle$; данный набор измерений будем называть *системой координат куба* $Cube(\overline{A})$, а элемент этого набора измерений – *координатой*

ячейки куба; *нулевой координатой* будем называть такую координату, в которой хотя бы один элемент является пустым множеством и обозначать её будем $\breve{0}$.

Таким образом, адресная функция с возможностью агрегации имеет вид

$$Fact_{\overline{\mathrm{A}}}(\breve{x}) = \begin{cases} Adr(< x^1, x^2, ..., x^m >), < x^1, x^2, ..., x^m > \in \overline{D}(Adr) \\ o, < x^1, x^2, ..., x^m > \notin \overline{D}(Adr) \\ o, \breve{x} = \breve{0} \end{cases}$$

$$\breve{x} =< x^1, x^2, ..., x^m >$$

Осталось ввести понятие самой *ячейки куба*. Выберем один из показателей для фактов куба $Cube(\overline{\mathrm{A}})$, пусть это будет f_i. Возьмём в качестве агрегирующей функции φ_k. Построим для куба систему координат (построим измерения, которые составляют систему координат по множествам точек измерения из прообраза адресной функции). Пусть это будет $\breve{X} = \left\langle \widetilde{X}^1, \widetilde{X}^2, ..., \widetilde{X}^m \right\rangle$, $m \in N$. Возьмём произвольно координату ячейки куба $\breve{x} = \left\langle X^{11}, X^{21}, ..., X^{m1} \right\rangle$, $X^{i1} \in \widetilde{X}^i$, $i = \overline{1,m}$ и построим для этой координаты множество $A = \left\{ < x^1, x^2, ..., x^m > | \, x^i \in X^i \right\}$. Построим теперь на основании множества A множество $B = \left\{ f_i(\alpha) \mid \alpha = Fact_{\overline{\mathrm{A}}}(< x^1, x^2, ..., x^m >), < x^1, x^2, ..., x^m > \in A \right\}$. Значение $\varphi_k(B)$ и будет ячейкой куба.

Таким образом, ячейкой куба $Cube(\overline{A})$ для показателей фактов f_i, агрегирующей функции φ_k с координатой $\breve{x} = \left\langle X^{11}, X^{21}, ..., X^{m1} \right\rangle$ в системе координат $\breve{X} = \left\langle \widetilde{X}^1, \widetilde{X}^2, ..., \widetilde{X}^m \right\rangle$ будем называть значение:

$$Cell^{f_i, \varphi_k}_{Cube(\overline{\mathrm{A}})}(\breve{x}) = \varphi_k \left(\left\{ f_i(Fact_{\overline{A}}(< x^1, x^2, ..., x^m >)) \mid x^i \in X^{i1} \right\} \right).$$

Агрегация показателей

Построим некоторое конечное, упорядоченное множество $M =< x_1, ..., x_n >$. Будем его называть *массивом агрегатов*. Договоримся, что для любого массива агрегатов существует отображение $\chi : M \to Y \subseteq R$. *Агрегатом* будем называть число $M[i] = \chi(x_i)$, $x_i \in M$, а i *– номером агрегата*.

При формализации многомерной модели данных мы договорились, что агрегирующей функцией будем называть отображение $\varphi : \mathscr{P}(R) \to R$. Уточним определение: будем под агрегирующей функцией понимать такое отображение $\varphi : \mathscr{P}(M) \to R$, где M – массив агрегатов. Кроме того, сформулируем такую *аксиому агрегирующей функции*: её результат не зависит от выбранного порядка в *M*. Автор предлагает несколько *методов агрегации* (*схем*).

Первой такой схемой является *итерационная схема агрегации*. Договоримся, что *Card M* = *n*, коме этого пусть *n* > 1, полагая $\varphi(\{x\}) = \chi(x)$. Для $\varphi(M)$ найдём такую функцию $\psi(y, x, i) \in [R]$, $y, x \in R, i \in N$, что $\varphi(M) = y_n$, где

$$\begin{cases} y_1 = M[1] \\ y_i = \psi(y_{i-1}, M[i], i) \end{cases}$$

Функцию ψ будем называть *итерирующей функцией* для агрегирующей функции φ.

Что же даёт нам использования итерационной схемы агрегации? Не секрет, что в целях анализа могут понадобиться различные агрегирующие функции и все их предусмотреть невозможно, поэтому при

разработке OLAP-средства необходимо учесть потребность в использовании пользовательских агрегирующих функции и предоставить средства для их разработки. Конечно, можно реализовать средство для разработки всей агрегирующей функции, но при этом оно было бы достаточно сложным, как для реализации, так и для использования, а можно воспользоваться итерационной схемой агрегации, при этом достаточно предоставить возможность проектирования только итерирующей функции, для проектирования которой вполне достаточно средств элементарной математики.

В итерационной схеме агрегации мы за каждый из n шагов обрабатываем всего один элемент массива агрегатов M. С целью повысить производительность построим такую схему, при которой за каждый шаг агрегировалось хотя бы два элемента.

Для начала, для агрегирующей функции φ построим итерирующую функцию второго порядка: $\psi^2(y, x_1, x_2, i) \in [R],\ y, x_1, x_2 \in [R],\ i \in N$, такую, что

$$\varphi(M) = \psi^2(y_{n-2}, M[n-1], M[n], n)$$

$$\begin{cases} y_1 = M[1] \\ y_i = \psi^2(y_{i-2}, M[i-1], M[i], i) \end{cases}$$

Такая схема, выдвигает два требования к множеству M:

1) $n > 2$;
2) *n* – нечётно.

Со вторым требованием можно «справиться» таким образом: если n – чётно, то добавить в качестве $M[n+1]$ ноль, так же можно поступить в случае невыполнения первого требования, т.е. дополнить M нулями до $CardM = 3$, единственное условие, что это необходимо будет учесть при разработке итерирующей функции.

Основную сложность в разработке итерирующих функций второго порядка составляет учёт добавленных нами нулей. В ряде случаев это может составить очень серьёзную проблему, поэтому предложим ещё один путь решения второй проблемы: будем использовать итерирующие функции как первого, так и второго порядков. При этом до ближайшего нечётного номера шага, не превышающего n, будем пользоваться итерирующей функцией второго порядка, а потом, если понадобится, воспользуемся итерирующей функцией первого порядка.

Обобщим теперь всё сказанное на случай *k*-го порядка итерирующей функции. Итерирующей функцией k-го порядка будем называть функцию $\psi^k(y, \bar{x}, i) \in [R]$, $y \in [R], i \in N$, $\bar{x}$ – вектор k-го порядка элементы которого – из R.

Построим следующую схему агрегации:

$$\varphi(M) = \psi^k(y_{n-k}, (M[n-k+1], ..., M[n]), n)$$

$$\begin{cases} y_1 = M[1] \\ y_i = \psi^k(y_{i-k}, (M[i-k+1], ..., M[i]), i) \end{cases}$$

Назовём эту схему *усовершенствованной итерационной схемой агрегации*. Её применение позволяет в разы снизить количество шагов, сохранив при этом преимущества итерационной схемы агрегации.

Часто при ответе на запрос системе приходится агрегировать уже агрегированные показатели (например, несколько ячеек куба). Эту задачу не получится решить, пользуясь итерационной схемой агрегации. Автор предлагает следующую схему агрегации, при которой возможна агрегация как простых агрегатов, так и уже агрегированных показателей.

Пусть имеется два массива агрегатов $M^1 =< x_1^1, x_2^1, ..., x_{n_1}^1 >, M^2 =< x_1^2, x_2^2, ..., x_{n_2}^2 >$ и агрегирующая функция φ; мы знаем значения $y_1 = \varphi(M^1)$, $y_2 = \varphi(M^2)$, $n_1 = Card\ M^1$ и $n_2 = Card\ M^2$. Требуется найти значение $\varphi(M^1 \bigcup M^2)$. Полагаем при этом, что $\varphi(\{x\}) = \chi(x)$. Будем обозначать $\psi(y_1, y_2, n_1, n_2)$ функцию агрегации агрегированных показателей y_1 и y_2; заметим, что при $n_2 = 1$ получаем итерирующую функцию первого порядка.

Итерационная схема агрегации с использованием функции агрегации агрегированных показателей выглядит следующим образом:

$$\begin{cases} y_1 = M[1] \\ y_i = \psi(y_{i-1}, M[i], i-1, 1) \end{cases}$$

Чтобы агрегировать более одного агрегированного показателя, достаточно построить суперпозицию функций агрегации двух агрегированных показателей.

Заключение

В данной статье приведено формальное описание реализации многомерной модели данных в программном комплексе METAS BI-Platform на языке теории множеств и теории функций. Были описаны методы агрегации данных, позволяющие эффективно агрегировать массивы числовых показателей.

Благодарности

Работа выполнена при поддержке гранта РФФИ № 08-07-90006-Бел_а.

Библиографический список

[1] Codd E.F. , Codd, S.B., Salley C.T. Providing OLAP (on-line analytical processing) to user-analysts: An IT mandate. Technical report, 1993 [PDF] (www.olap.ru).

[2] Мальцев П.А. Разработка компонента визуализации многомерных данных для CASE-технологии METAS // Технологии Microsoft в теории и практике программирования. Материалы конференции. Нижний Новгород: Изд-во Нижегородского университета, 2006. С. 202-204.

[3] Мальцев П.А., Лядова Л.Н. Формализация многомерной модели данных // Математика программных систем: Межвузовский сб. науч. тр. / Перм. ун-т. Пермь, 2006. С. 74-87.

Сведения об авторе

Павел Мальцев *– Пермский государственный университет, аспирант кафедры математического обеспечения вычислительных систем; Россия, г. Пермь, 614990, ул. Букирева, 15;*
e-mail: pavel_maltsev@mail.ru