AN APPLICATION OF BUSINESS PROCESS MODELING SYSTEM ILNET

Dimitar BLAGOEV, George TOTKOV, Elena SOMOVA
Department of Computer Science, Faculty of Mathematics and Computer Science, The University of Plovdiv “Paisii Hilendarski”,
24 Tzar Assen st., 4000 Plovdiv, Bulgaria, tel.: +359 886 401345,
e-mail: gefrix@gmail.com, totkov@uni-plovdiv.bg, eledel@uni-plovdiv.bg

ABSTRACT

In this paper we present a system for accompanying of the process of Bachelor and Master thesis administration, including all stages: publication of thesis themes, thesis choice, documents fulfillment, thesis writing, thesis submission, thesis review, thesis publication, etc. The system is developed on the base of business process modeling system ILNET and its basic model for describing and executing workflow models. The basic model allows the creation and usage of both high-level constructs that are available in all popular business modeling standards as well as low-level programming language constructs that allow the implementations of new building blocks. Amongst the main features of the system are the capability of the blocks to implement their own custom visualization using the same workflow model, the centralized independent hosting and execution of the models, the ability to hot-swap a compiled and running model with another (newer) one and the automatic wrapping of web services and .NET libraries into ILNET building blocks.

Keywords: *workflow, business process modeling, administration of education*

1. INTRODUCTION

In this paper we introduce a novel process modeling environment and present an approach towards the automation of administrative processes. Still big portion of the administrative work at learning institutions is being carried out manually on paper. Some of the processes are short and can be done fast, while others spread over a long period of time, involve collecting and processing data from multiple members and is more difficult to monitor and administer. The notion of modeling and executing business processes using workflows or other techniques has been around for decades. Initially in the late 1970s it was focused on office automation systems (Officetalk-Zero [2]), shortly after systems supporting organizational processes emerged (SCOOP [23], which used Petri Nets). Since then many new tools and approaches have been introduced and new standards emerged trying to help different tools exchange process definitions. Using Petri Nets [16, 18] and in particular Colored Petri Nets [8, 10] is a good choice in cases of computer algorithm analysis and improvement [9]. Presenting a good base for simulation of process models is probably one of the reasons some business process modeling systems [3, 6] to choose CPN as their underlying model representation language, while some of the others focus on model visualization and visual description, not execution or analysis.

Amongst the vast number of tools and models two standards have become widely accepted – the BPMN (Business process modeling notation) [4, 20] and EPC (Event-driven process chain) [11, 14]. These two seemingly completely different process modeling approaches are used by most of the recently developed tools in the field. The execution of the BPMN models has been possible with the help of the BPEL (Business process execution language) standard [13], while XPDL (XML process definition language) [17, 21] focuses more on the visual representation of the models (also known as Workflow process description language in its first revision from 1998, before it began supporting the BPMN constructs). The latter also works as an exchange format for process definitions between different systems (including those based on the EPC approach). The second version of the BPMN standard [19] also supports process execution.

The “Workflow Patterns” initiative [22] under the supervision of Prof. Wil van der Aalst begins in 1999. The goal of the initiative is to present a thorough study of the different perspectives (workflow control, data, resources and exception handling) that should be supported by a process management or business process modeling standard. Currently there are 43 patterns for workflow control, 43 for working with resources, and 40 for data and 108 (derived) for exception handling. In Table 1 it is shown a comparison of the number of supported patterns in some of the most widely used standards for process modeling (BPMN, UML [5, 12], EPC) and process execution and management (BPEL, XPDL) in each of the Van der Aalst classification’s categories. The values (Table 1) display that different standards support different number (and type) of patterns for workflows, which creates the following complications: one standard is not always possible to transform lightly to another; before starting business process modeling with some of the existing standards, it is necessary a research and careful choice of the standard, which is better to use.

Table 1 Supported workflow patterns by some of the business process modelling standards

Patterns	BPEL	BPMN	XPDL	UML	EPC
Workflow Control	21	33	33	30	12
Data	19	22	16	18	-
Resources	28	8	-	8	-
Exception Handling	16	32	18	-	-

Often new patterns are being added for either further simplifying the process modeling or fixing problematic zones that arises from the use of already existing prior patterns. One issue which can be observed is that there isn't a single standard that covers or can promise to cover all of the possible necessary patterns for optimal and seamless process description. Below we present a possible solution, by proposing a system in which new building elements and patterns can be defined in the same manner as processes.

2. SYSTEM ILNET

The ILNET system is a light framework for creating business process modeling solutions. It is the final product of a long and extended development of series of tools for linguistic processing. One of its main features is that, although a fixed base model with which processes can be defined and executed is given, there is a built-in feature to enhance and adapt both the graphical editor and the underlying engine, to define and execute processes using models with which the end users may be more familiar with. Currently Petri Net modeling is in testing and most of the BPML elements are drafted.

The first version of the system was designed to be used primarily in the field of computational linguistics [1]. The system's model is directed graph in which the nodes can contain custom code and control the head movement of the built-in text reader. The logic of the parser and the code of all the nodes are compiled and executed in a custom virtual machine. The system is used successfully in the text-to-speech solution SLOG [15]. The use of memory and simplified programming language syntax for the nodes allows the user to model various types of linguistic grammars in one tool including non-deterministic finite state automata and context-sensitive grammars.

The new and current version of the system, called ILNET [7] (now based on .NET), is used during the creation of a prototype e-learning architecture for modeling e-learning processes related to the course workflow. It is an extended version of the previous system model minus the built-in text reading head. Instead of a using virtual machine the model is translated into C#.NET.

The system is modular and consists of five main components, which communicate between them through inside messages or SOAP-based web services:

- Workflow compiler, which translates process descriptions to programming code;
- Server for execution and management of the described processes;
- Graphical editor for visual process modeling;
- Event manager which handles internal communication between processes and provides a gateway for the outside environment to exchange messages with running process instances using SOAP;
- Building blocks library which contains descriptions of utility sub-processes which can be used for more rapid process modeling and also provides the ability to use familiar structures from other modeling tools.

Figure 1 presents the architecture of the system in common.

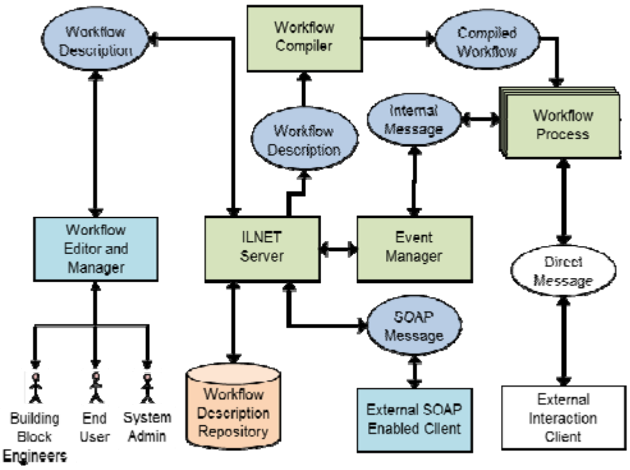


Fig. 1 Arhitecture of ILNET

Another feature of the proposed system is its memory model. The language for process description in ILNET supports working with variables on three different access levels:

- Local variables: they are always copied and are accessed by only one instance. They can represent the memory of workflow thread, or Petri Net token. They are used for calculations which do not depend on the result of other simultaneously running process threads, eliminating the risk of accessing one variable being accessed from more than one place at the same time;
- Intermediate variables: they are copied when the process is initializing. They are accessed by each control workflow, derived from the initial (when the process is started). They are used for synchronization of the active roads in the frame of one calling of the process and for implementation of most of the templates used for modeling of business processes which deal with synchronization of workflows;
- Global variables: they are never copied and are available to all instances. These variables provide a way of creating process instance data pools where different instances of the same process collect and share data. For short-term data storage of non-critical data these variables are faster and easier to work with than using an external database (which is preferred for critical or long-term data).

Using all three types of variable access levels it is possible to easily model complex data workflows.

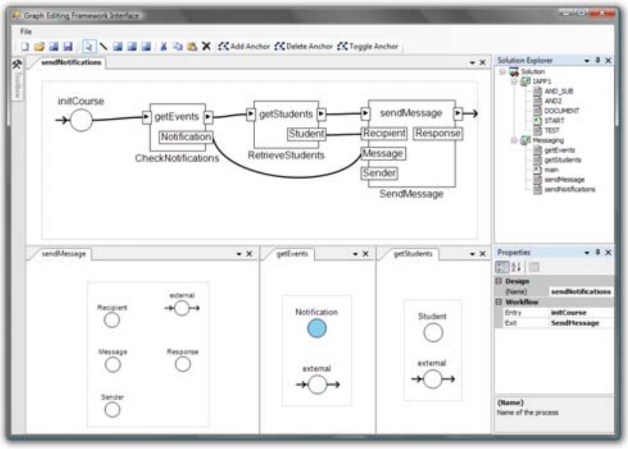
Loading and execution of the modeled processes and transferring of inner- and outer- process messages is realized by ILNET server. The server operates with events for work with messages and serves for connection between the surrounding environment and the running process instances giving opportunity for:

- loading process models;
- starting processes;
- receiving and sending messages;
- termination of processes instances.

The graphical editor for visual process modeling provides means for seamless process modeling in the ILNET framework. Figure 2 shows the working screen of the graphical interface of the module for visual process editing GEFI. The environment enables several processes to work simultaneously, they are synchronized in real time with the server, in which they are loaded and gives basic means for tracking and monitoring of executed instances of modeled processes.

The environment supports the use of different modeling standards using custom building blocks. The resulting visual information generated for a particular model is attached to the base elements as metadata.

The current version of the system is implemented in C#.NET and the same language is used in the models for variable declarations and function definitions at the lowest level (building blocks). The workflow compiler, event manager and ILNET server are hosted in one thick windows service. The SOAP message communication is provided from a separate thin ASP. NET web service which acts as a proxy to an ILNET service.



Design
(Name) sendNotifications
Workflow
Entry initCourse
Exit SendMessage

(Name)
Name of the process

Fig. 2 Grafical environment for process edition

Internal message communication occurs when a process invokes the internal messaging system using one of the built-in functions for creating message listeners, sending message responses. The following functions are available:

- raiseEvent (Message) – generates an event with a message body and waits until an answer is available in the EM;
- queryEvents (Query) – queries the EM and returns a list of messages ids that are waiting for processing in the EM queue;
- fetchEvent (EventId) – retrieves the body of the message with the given event id;
- answerEvent (EventId, Answer) – sends an answer to a waiting message event in the EM. Multiple answers can be sent while the event is open, each resulting in the creation of a separate execution thread;
- closeEvent (EventId) – closes an event that has been processed;
- listenEvent (Filter) – hooks the action to the EM. Each time an event is generated with a message that matches the filter the hooked action will be triggered;
- closeListener (ListenerId) – unhooks the listener from the EM.

This type of message communication is not new, however it provides great flexibility when embedded in a process modeling system. For example, one process can identify itself as a sort of service provider by creating a listener for a specific type of messages. By doing so, other running processes are able to send requests and receive responses from this process service. The usage of built-in messaging system provides the faster intra-process communication (there's no need to wrap the messages in a particular standard like SOAP or JSON) and will also enable the two processes to be on different machines when the ILNET server is running in a distributed environment.

On the other hand, direct message communication occurs when a particular process model connects via TCP to external clients (for example to connect to a database, to send or download emails, to download a web page, or to connect to cloud services like Amazon's S3 and SES). This type of connectivity is essential for the creation of useful utility building blocks.

During workflow execution, after successfully completing an action, each of the next available actions is activated using a user-defined execution model. Every workflow can have its own execution model, use the default, or inherit the model of parent workflow. The default model is to simply follow all available transitions asynchronously but custom execution helps significantly when creating modelers for different modeling standards (for example for Petri Nets a custom model which decides which transitions to follow can be used to achieve better and more efficient simulation execution speed as opposed to using calls to building blocks for every transition).

The workflow compiler takes a workflow description of a given process and outputs executable library that can be loaded and used to create running instances of that workflow. The implementation is generic, and is not bound to a particular programming language. Instead, a *shell* program is used as a template during the code generation process. The shell contains the code for a single workflow with a single function and one variable but each section is tagged (commented) in a specific way, which allows easy replication and alteration of the code. All other parts of the workflow compiler reuse the code from the shell, thus enabling the easy switch to a different function language if needed.

One novel approach used in the tool is the use of a hybrid client-server rendering engine. While most systems provide either full-client or full-server rendering (for web clients) GEFI uses client-side rendering for the model whilst allowing server-side rendering for specific elements (building blocks) of the model. In essence, this means that the tool can be used to edit how the blocks are rendered inside the tool. In addition, since the rendering of the block is handled by the block description on the side of the server, it is possible to model complex interactive multi-user elements.

3. APPLICATION SYSTEM

The paper presents an application of the business process modeling system ILNET and its basic model for describing and executing workflow models, introduced in the previous section. The application system has a purpose to follow and maintain the whole administrative process of Bachelor and Master thesis preparation of the students for the graduation of the respective university degree.

The process covers the following stages: publication of the thesis themes, thesis choice, administrative acceptance, documents fulfillment, thesis writing with support with help materials (as thesis templates and information about university thesis rules and time schedule about each stage of preparation of the graduation thesis), thesis submission and publication, additional materials (as thesis defense presentation, code of the developed software, software documentation, published scientific papers and other prepared materials) publication, thesis review, thesis evaluation and thesis view.

Administrative acceptance is done on two stages of the process, according to the university rules – first, tutor has to accept the concrete student for the chosen theme and second, the head of the department has to accept the student to work on thesis with the respective tutor (*Application Form 1*). During the whole process two official paper documents (application forms) have to be fulfilled. The second official document – *Application Form 2* is a request for permission of defending the thesis in front of the evaluation commission. The user types that can use the opportunity of the system are student, teacher-tutor, teacher-evaluator and teacher-head.

The tutor publicizes the thesis themes that he/she gives, views and updates own themes, views received requests from students to work on a particular theme, answers to the students' requests (accepts or rejects) and reviews the thesis. The tutor can follow all steps of the student during the whole process of preparation of the thesis.

The head, which is the head of the department, approves the *Application form 1* and determines the reviewer of the thesis.

The evaluator, which is a member of the thesis evaluation commission, has the opportunity to see the list of themes for defense and publishes the final thesis mark. The student chooses the thesis theme from the list of free themes, fills in two application forms (which is almost automatic), views informational and help materials, and submits thesis and additional thesis materials.

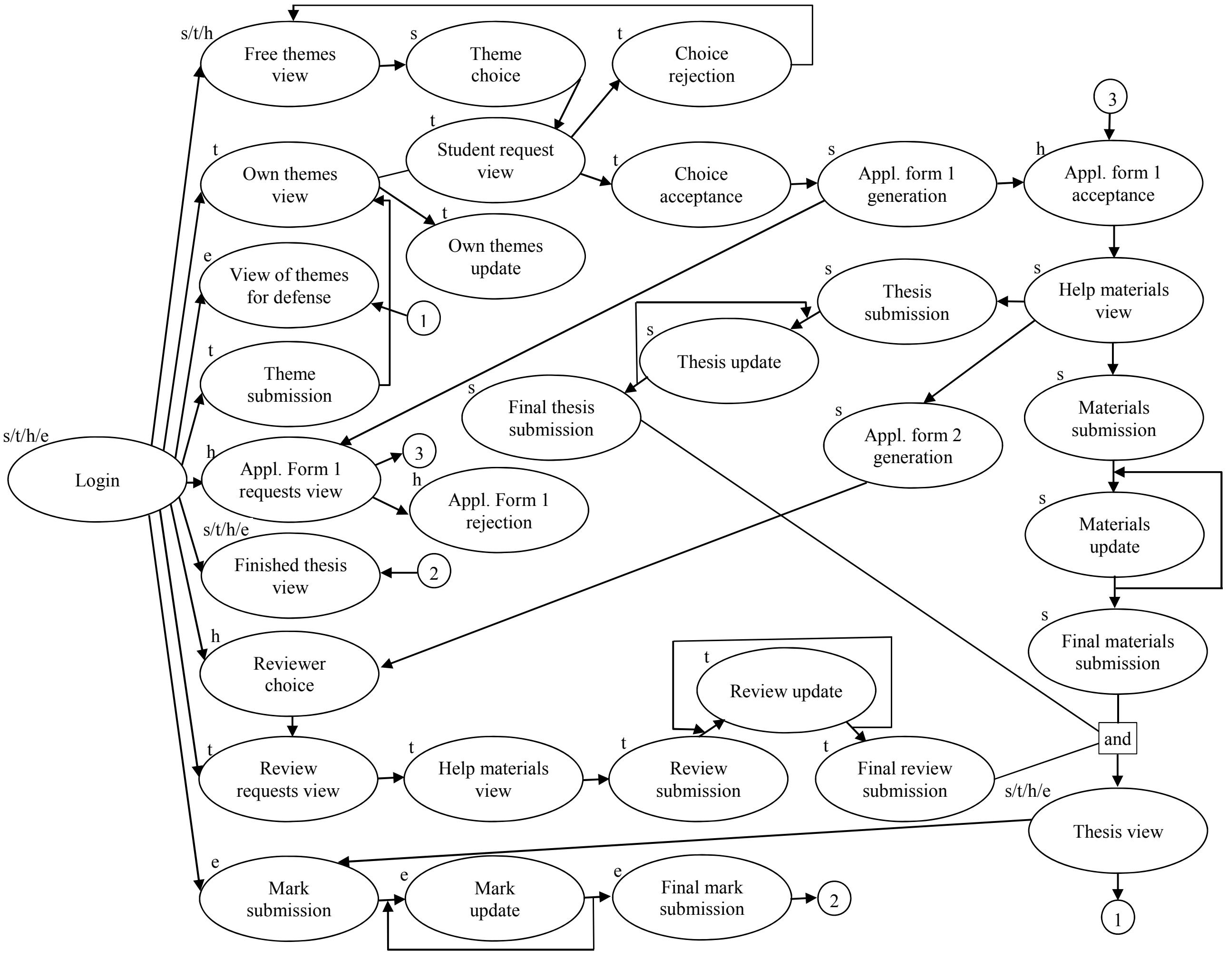


Fig. 3 Workflow model of the application system

Different views are prepared for each type of user – free themes, own themes, themes for defense, *Application Form 1* requests, current thesis with status, review requests and finished thesis.

The chosen approach about passing over processes for all users is to use one workflow for all users instead of different workflows for each of them. The used approach gives easy management of processes and control of the relationship between processes executed by different users.

The scheme of the thesis administrative process is given in the Figure 3. In the figure the possible actors of the actions are pointed with s, t, e and h respectively for student, tutor, evaluator and head.

Passing over the workflow is consecutively with some time interruptions when particular user has to wait other user to execute some action. For example, while tutor do not accept student request for specific theme, student cannot continue the process and start to prepare the thesis.

In the process there are some stages that depend by the time limitation – fulfillment of both application forms and all final submission (thesis, materials, review and evaluation mark). Passing through these stages can be done by user choice or reaching the definite time.

In the workflow there is used one special type of ILNET blocks, which presents and-logic in execution. For example, only after when the thesis, the additional materials, the review and the *Application Form 2* (according to the university rules) are submitted, the thesis work is prepared and ready for evaluation.

The application system is developed on the base of the following software technologies: MS Visual Studio 2010, MS Silverlight 4.0 and LINQ and database MS SQL Server.

The work is funded by projects DO 02-308 to the National Science Fund and BG051 PO 001-3.3.04/13 to European Social Fund of Operational Program “Development of the human resources” 2007-2013.

4. CONCLUSION

The work presents example how to realize an application system on the base of business process modeling system using workflow models. The system follows the whole administrative process of preparation of Bachelor and Master graduation thesis. We intend to expand the system with connection to the university student system for automatic decision if the student has the right to prepare thesis according to the evaluation marks.

REFERENCES

[1] BLAGOEV, D. – TOTKOV, G.: Visual Parser Builder, RANLP’05, Borovetz, pp. 112–116.

[2] ELLIS, C. A. – NUTT, G. J.: Office Information Systems and Computer Science, ACM Comput. Surv. 12, 1 (March 1980), pp. 27–60. DOI: 10.1145/356802.356805, http://doi.acm.org/10.1145/356802.356805

[3] DESEL, J. – OBERWEIS, A. – REISIG, W. – ROZENBERG, G.: Petri Nets and Business Process Management, http://www.dagstuhl.de/Reports/98/98271.pdf, 1998.

[4] DIJKMAN, R. – DUMAS, M. – OUYANG, Ch.: Semantics and analysis of business process models in BPMN, Information and Software Technology, Volume 50, Issue 12, 2008, pp. 1281–1294.

[5] DUMAS, M. – van der AALST, W. – ter HOFSTEDE, A. – ENGELS, G. – FÖRSTER, A. – HECKEL, R. – THÖNE, S.: Process-Aware Information Systems: Bridging People and Software through Process Technology, Chapter 5. Process Modeling using UML, published by John Wiley & Sons, Inc., 2005.

[6] GIAGLIS, G.: A Taxonomy of Business Process Modeling and Information Systems Modeling Techniques, International Journal of Flexible Manufacturing Systems, Volume 13, Number 2, 2001, pp. 209–228.

[7] INDZHOV, Hr. – BLAGOEV, D. – TOTKOV, G.: Executable Petri Nets: Towards Modelling and Management of e-Learning Processes, ACM International Conference Proceeding Series; Vol. 375, Proc. of the 10th International Conference on Computer Systems and Technologies and Workshop for PhD Students in Computing 2009, Rousse, Bulgaria, June 18-19, 2009, IIIA.12.1 IIIA.12.6.

[8] JENSEN, K.: Coloured petri nets: A high level language for system design and analysis, Lecture Notes in Computer Science, Volume 483/1991, 1991, pp. 342–416.

[9] KOREČKO, S. – SOBOTA, B. – SZABÓ, C.: Performance Analysis of Processes by Automated Simulation of Coloured Petri Nets, 10th International Conference on Intelligent Systems Design and Applications (ISDA), 2010, Cairo, Nov. 29, 2010 –Dec. 1, 2010, pp. 176–181.

[10] LIU, D. – WANG, J. – CHAN, St. – SUN, J. – ZHANG, L.: Modeling workflow processes with Colored Petri Nets, Computers in Industry, Volume 49, Issue 3, 2002, pp. 267–281.

[11] MENDLING, J. – NEUMANN, G. – NÜTTGENS, M.: Towards Workflow Pattern Support of Event-Driven Process Chains (EPC), Second GI-Workshop XML4BPM XML for Business Process Management within the scope of the 11th Conference Business, Technologie, and Web, Karlsruhe, Germany, 2005.

[12] NÜTTGENS, M. – FELD, Th. – ZIMMERMANN, V.: Business Process Modeling with EPC and UML: Transformation or Integration?, http://wiso.uni-hamburg.de, 1998.

[13] RECKER, J. – MENDLING, J.: On the Translation between BPMN and BPEL: Conceptual Mismatch between Process Modeling Languages. In Latour, Thibaud & Petit, Michael (Eds.) The 18th International Conference on Advanced Information Systems Engineering. Proceedings of Workshops

and Doctoral Consortium, Namur University Press, 2006, pp. 521–532.

[14] SARSHAR, K. – LOOS, P.: Comparing the control-flow of EPC and Petri Net from the end-user perspective, in: W. Aalst, B. Benatallah, F. Casati, F. Curbera (Eds.), Business Process Management, Proceedings of 3rd International Conference BPM'2005, Nancy, France, 2005, pp. 434–439.

[15] TOTKOV, G. – BLAGOEV, D. – ANGELOVA, V.: Slog 1.0: Bulgarian text to speech transformation system, International Joint Conf. on Computer, Information, Systems Sciences and Engineering (CIS2E'05), Bridgeport (CT), Dec. 10 – 20, 2005.

[16] van der AALST, W. – STAHL, C.: Modeling Business Processes: A Petri Net Oriented Approach. MIT press, Cambridge, MA, 2011.

[17] van der AALST, W.: Patterns and XPDL: A Critical Evaluation of the XML Process Definition Language, QUT Technical report, Queensland University of Technology, Brisbane, 2003.

[18] van der AALST, W.: The Application of Petri Nets to Workflow Management, The Journal of Circuits, Systems and Computers 8 (1), 1998, p. 21-66.

[19] van GORP, P. – DIJKMAN, R.: BPMN 2.0 Execution Semantics Formalized as Graph Rewrite Rules: extended version, Beta Working Paper, series 353, Eindhoven University of Technology, 2011.

[20] WHITE, S.: Process Modeling Notations and Workflow Patterns, BPTrends, http://www.omg.org/bp-corner/bp-files/Process_Modeling_Notations .pdf, 2004.

[21] WHITE, S. – BEYOND, S.: XPDL and BPMN, Workflow Handbook 2003, published by Future Strategies Inc., 2003.

[22] Workflow Patterns Initiative, http://www.workflowpatterns.com/.

[23] ZISMAN, M.: Representation, Specification and Automation of Office Procedures, PhD Dissertation, Department of Business Administration, Wharton School, University of Pennsylvania, Philadelphia, PA, 1977.

Received January 7, 2012, accepted April 26, 2012

BIOGRAPHIES

Dimitar Blagoev was born on 05.12.1983. In 2006 he graduated (MSc) with distinction at the Faculty of Mathematics and Informatics at Plovdiv University in Plovdiv. Since 2009 he is working as a tutor with the Department of Computer Science. His scientific research is focusing on computational linguistics, process modeling, test automation and e-Learning. In addition, he also investigates questions related with image processing and software design patterns.

George Totkov was born on 23.04.1950. In 1973 he graduated (MSc) at Sofia University in Bulgaria. He defended his PhD thesis in 1979 and DSs thesis in 2005. Totkov is Professor at Computer Science Department at Plovdiv University since 2009. Prof. DSc G. Totkov is vice rector of Plovdiv University and head of the Regional Distance Education Center at Plovdiv University. His main scientific interests are in the sphere of computer science (e-learning, computer linguistics, information modeling, etc.) and computational mathematics (approximation, mathematical statistics, etc.).

Elena Somova was born on 14.11.1972. In 1995 she graduated (MSc) at Plovdiv University in Bulgaria. She defended her PhD thesis "Platform for Design, Development and Maintenance of e-Learning Environmets" in 2003. She is Associate Professor at Computer Science Department at Plovdiv University since 2007. Her major fields of scientific research are e-learning, innovative learning technologies, metadata standards in different domains (incl. learning), e-learning environments, modeling of learning processes, etc.