PROCEEDINGS

of the Union of Scientists - Ruse

Book 5

Mathematics, Informatics and Physics

Volume 8, 2011

RUSE

ISSN 1311-9974

The Ruse Branch of the Union of Scientists in Bulgaria was founded in 1956. Its first Chairman was Prof. Stoyan Petrov. He was followed by Prof. Trifon Georgiev, Prof. Kolyo Vasilev, Prof. Georgi Popov, Prof. Mityo Kanev, Assoc. Prof. Boris Borisov, Prof. Emil Marinov. The individual members number nearly 300 recognized scientists from Ruse, organized in 13 scientific sections. There are several collective members too – organizations and companies from Ruse, known for their success in the field of science and higher education, or their applied research activities. The activities of the Union of Scientists – Ruse are numerous: scientific, educational and other humanitarian events directly related to hot issues in the development of Ruse region, including its infrastructure, environment, history and future development; commitment to the development of the scientific organizations in Ruse, the professional development and growth of the scientists and the protection of their individual rights.

The Union of Scientists – Ruse (US – Ruse) organizes publishing of scientific and popular informative literature, and since 1998 – the "Proceedings of the Union of Scientists- Ruse".

BOOK 5

"MATHEMATICS, INFORMATICS AND PHYSICS"

VOLUME 8

CONTENTS

Mathematics

Meline Aprahamian 7
Mean Value Theorems in Discrete Calculus

Antoaneta Mihova 13
Polynomial Identities of the 2x2 Matrices over the Finite Dimensional Grassmann Algebra

Veselina Evtimova 19
Analysis of the Impact of the Incoming Calls Flow Intensity on Some Basic Characteristics of an Emergency Aid Centre

Veselina Evtimova 25
A Study on the Influence of Incoming Calls Flow Intensity on the Waiting Time Characteristics of an Emergency Medical Aid Centre

Ivanka Angelova 31
Numerical Solution of the Two-Phase Stefan Problem for Sphere

Ivanka Angelova 38
Mathematical Models of Interface Problems for Steady-Unsteady Heat Conduction

Informatics

Valentin Velikov 44
Some Possibilities For Automatic Programs Generation

Margarita Teodosieva 50
Information System for Medicines

Mihail Iliev 55
Extending the Lifetime of Wireless Sensor Networks by Using a Modified Method for Hierarchical Organization of the System in Clusters with Unequal Number of Devices

Georgi Krastev, Tsvetozar Georgiev 63
One Approach for Continuous Signals Representation

Viktoria Rashkova 70
Design and Implementation of Knowledge Control Test System

Physics

Galina Krumova 77
Calculations of Light, Medium and Heavy Neutron-Rich Nuclei Characteristics

Vladimir Voinov, Roza Voinova 86
Calculation of the Characteristic Impedance of a Microstrip, Reversed Microstrip and Embedded Microstrip Lines

Galina Krumova 93
Some Problems of Atomic and Nuclear Physics Teaching

Tsanko Karadzhov, Nikolay Angelov 101
Determining the Lateral Oscilations Natural Frequency of a Beam Fixed at One End

BOOK 5

"MATHEMATICS, INFORMATICS AND PHYSICS"

VOLUME 8

Education

Plamenka Hristova, Neli Maneva 106
An Innovative Approach to Informatics Training for Children

Margarita Teodosieva 114
Using Web Based Technologies on Training in XHTML

Desislava Atanasova, Plamenka Hristova 120
Human Computer Iteraction in Computer Science Education

Valentina Voinohovska 125
Computer – based conceptual mapping for facilitation of creative and meaningful learning in the course of "Multimedia Systems and technologies"

Galina Atanasova, Katalina Grigorova 132
An Educational Tool for Novice Programmers

Valentina Voinohovska 139
A Course for Promoting Student's Visual Literacy

Magdalena Metodieva Petkova 145
Teaching and Learning Mathematics Based on Geogebra Usage

Participation in International Projects

Nadezhda Nancheva 153
Mosem 2 Project - Learning Electromagnetic Phenomena and Superconductivity by Integration of Data Acquisition, Data Video, Modelling, Simulation and Animation

SOME POSSIBILITIES FOR AUTOMATIC PROGRAMS GENERATION

Valentin Velikov

Angel Kanchev University of Ruse

***Abstract**: Automatic program generation is a direction in the software industry, connected with saving time and human resources, as well as and receiving syntactic clear and logically correct units. It is possible to separate the application to interface and business logic, too. That means to create a new or use different existing toolsets for interactive generation of the necessary system.*

***Keywords**: automatic programs generation, DBMS, User interface*

INTRODUCTION

In many places people are working in the automatic software generation area. That is a problem, connected not only with human resources and time for development saving. Program generation allows to create syntactic clear and logically correct units – that is connected with time and price for software development, too. Another reason – some time it's difficult to find the suitable specialist to create the necessary correction in a software unit.

It's possible to examine the Automatic software/application generation in a different direction. One of classifications can be on a functional sign – we can separate the application in two parts: user interface and a functional part. So we can differentiate two directions:

1. Interfaces automatic generation;
2. Business-logic automatic generation (i.e. – application logic – some information system).

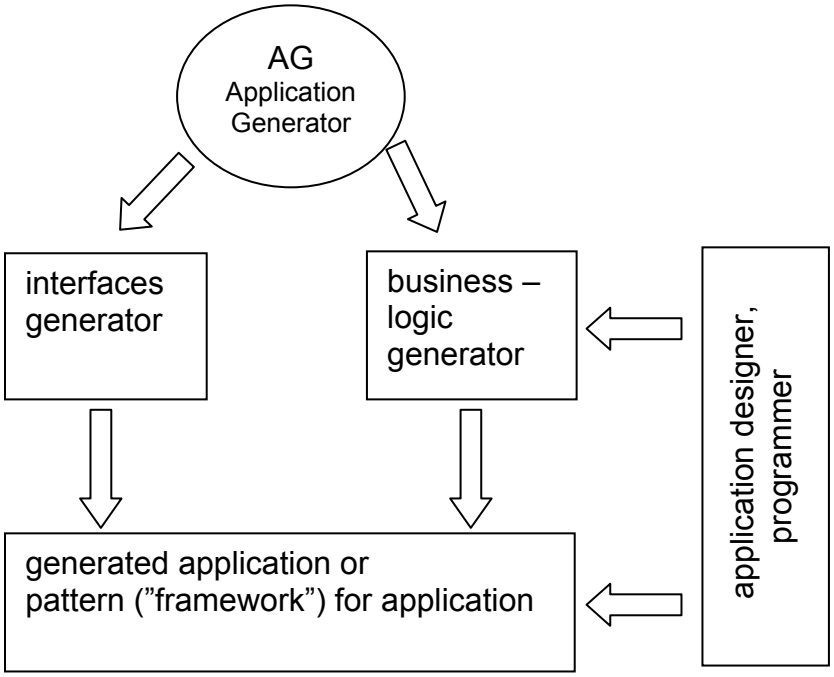

fig. 1 – automatic system generation

DETAIL DESCRIPTION

Interfaces automatic generation – there are many toolsets for visual interface creating (windows and objects in them) – MS Visual C, MS Visual Basic, Delphi, [3] and so on. Summary – they describe the objects and their attributes (size, position, colours; actions for responding – mouse, keyboard, another objects; the interaction with another objects and so on).

Business-logic generator: create software units, which can be parts of code, competed finished application or description of documents streams only (using them, the designer team can create a working application). The purpose is time saving (i.e. – resources) to software development. There are a few systems in that area, one of the leaders is from IBM - Rational System Architect. But to use this very expensive, very good and very effective system are necessary good knowledge of UML [2].

We can examine The Menu generator in few aspects:

1. Natural languages analysis. In this case, using a natural language description [1] (possible – limited language), it's generating frames (or filling of beforehand creating such, describing stage or scenarios);
   - stage – describe a static situation;
   - scenarios – describe a dynamic situation (actions);
2. Using preliminarily created frames (describing stages or scenarios), filling from response designer, it's creating a software generation for application with the necessary possibilities.

As a condition of primary importance for effective automation the highly efficient interaction between the user and the system is also pointed out. As the requirements to the functioning of the interface system are completely independent from the problems been solved in the course of interaction, the development of the interaction itself can be differentiated in a separate research area. In many cases it's more reasonable to divide the construction of a software system into an interactive module and a problem-oriented module. From another point of view these modules can be considered as managing and executive modules. In the executive module there are included subroutines performing particular actions on the functioning of the system. To the managing module the so called "cover" belongs, i.e. the hierarchical set of submenus, providing a unique choice and activation of the correspondent subroutine from the executive module. A great deal of the time for the implementation of a software system is spent for the designing and testing of this "cover". Hence the idea of computer aided design of this module comes into being.

It's convenient to develop a specialized graphics editor, which will allow describing iteratively the menus in the system being designed. By means of this graphics editor the designer of the system (fig. 2) determines the place of each menu in the hierarchy of the specified menus, defines its attributes (type and displacement of the frame on the screen, text, colours, etc.). Thus a description is entered iteratively of all the submenus and the procedures connected to the options of the main menu, accordingly of the submenus stemming from the main menu. For instance (fig. 3): an option in the created menu can activate either a subroutine (an executable file – in this case the path of the file must be specified, as well as its name), or can activate the creation of a new subroutine which is to be described in the same way (i.e. a recursive subroutine can be applied for the description of the menus). It's appropriate all the values of the menu attributes to be taken from previously defined sets. This allows a full formal and logical control from the system.

Two basic approaches are possible when creating this system:

a.) Retrieval of a previously fixed template of menus, marking those of them, which will be used, thus making them active. The inactive fields in the menu are no longer displayed. At the bottom of the tree (the template) – as the elements of the "leaves" the

user specifies the path and name of the program which is to be activated. To start working with the system the template thus prepared is activated. An advantage of this approach is that the menu interface is easier to implement. A disadvantage is the larger program (managing "cover") – it contains the codes of possible, but not used alternatives.

b.) A menu description is made according to a previously specified algorithm, which takes its hierarchical place in the tree of menus. For each position of this menu definite information is filled in, which indicates whether another menu will be activated though it, or a subroutine will be started. The action is a recursive. This tree is being built, containing information on the hierarchical sequence of menus, connected with the function of the software system being designed. The next stage is the retrieval of the tree, thus a text file is generated, which contains a set of commands in a given programming language, providing the work of the defined "cover" (i.e. – the source of the program). The last stage is compilation in the corresponding programming language, as previous additional corrections and tuning of the obtained text file are possible. An advantage of this approach is the possibility of replacing just the generator module to get programs in different programming languages. The remaining part of the software system – the creating and retrieval of the internal machine representation of the data structure (multy-moving tree) remains standard and does not require additional changes and tuning.

According to the internal machine representation of the menus being described it's convenient to keep information on them in a multy-moving tree (fig. 4). A node of the tree corresponds to each submenu. The hierarchical place of the node determines the hierarchical place of the submenu. Each node keeps part of the specific information on the menu. This menu attributes are stored in an array. The elements of the tree are records/structures (fig. 4), providing information on:

1. option name in the menu;
2. path to an executable subroutine in case this option is chosen;
3. a message to the option;
4. row on which the message is displayed;
5. number of an element in the attribute array;
6. a pointer to a submenu;
7. a pointer to the next option of the same menu.

The array elements are also records/structures containing information on:

1. row and column coordinates;
2. coordinate of the frame upper left corner;
3. number of the characters in a row;
4. number of the rows in a menu;
5. frame type;
6. colour of the text displayed in the menu;
7. background colour of the text;
8. text colour of the chosen option;
9. background colour of the chosen option;
10. frame colour;
11. background frame colour.

After the description of the interface of the software system is finished the multy-moving tree (fig. 4) can be saved in a file (in this case – a text file, including additional control characters). Editing is possible – deleting/adding of a submenu, changing of a attributes and so on. It's an advantage that the user of the system is not interested in a tree – on the screen he gets the menus as they were created and he can edit the existing image.

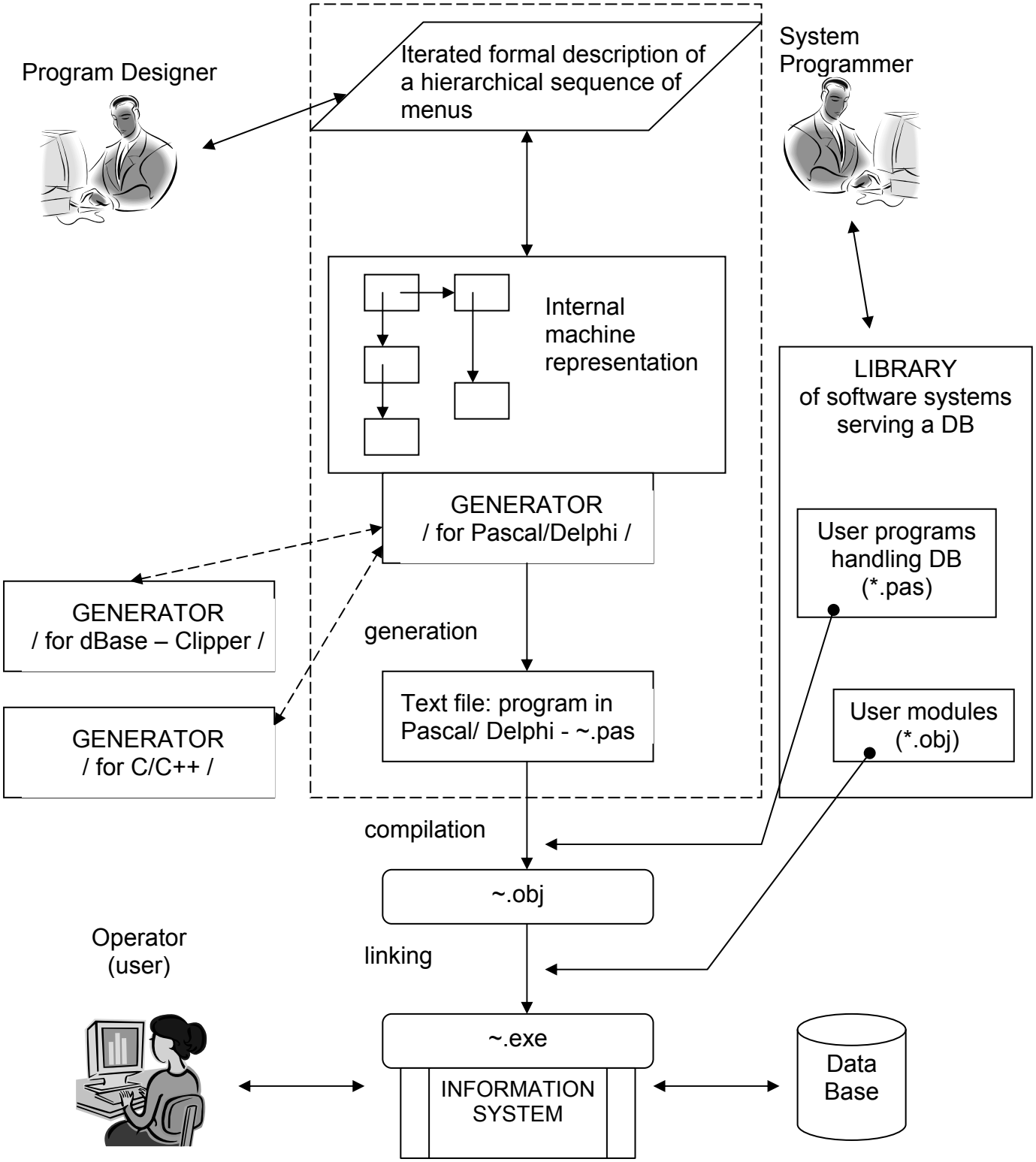

fig. 2 – an application system with automatic generated user interface

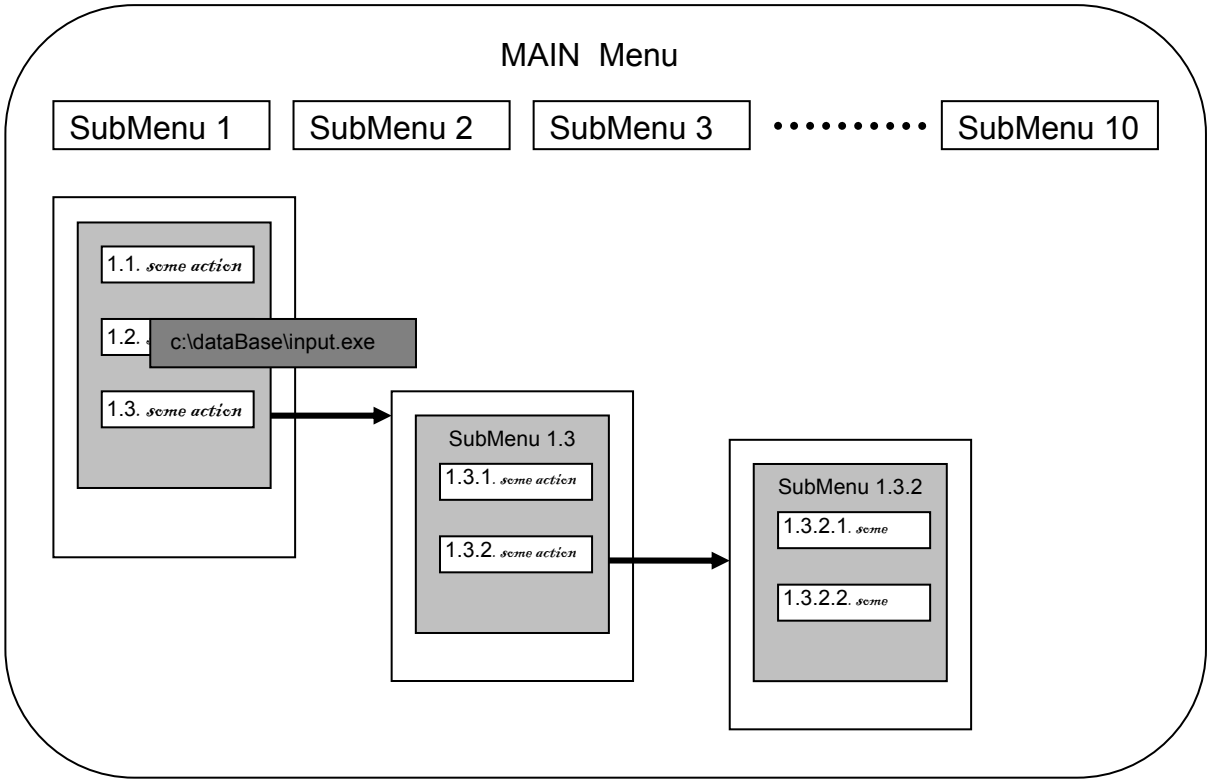

fig. 3 – Automatic generated menus

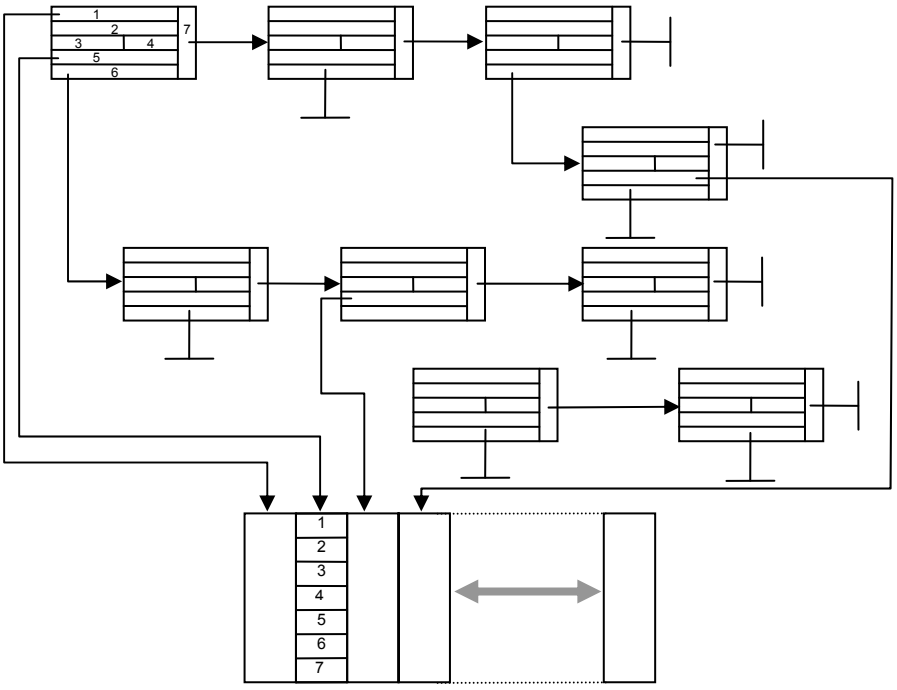

fig. 4 The multy-moving tree

CONCLUSION

Several limitations which are not of grate importance, are inherent to this approach for design of the menu generator – it's assumed that the main menu will not have more than 10 alternatives; the number of the nested levels of submenus is 3.

The suggested architecture is widely applicable by its means any managing module ("cover") of a software system can be designed (fig. 3). No direct connection is necessary to exist between the designers of an information system and the designers of programs (completed modules) for managing the corresponding database(fig. 2). By replacing just the generating module, work with any DBMS is possible.

Creating and using applications of similar systems is closely connected with the automation of programmer's work.

REFERENCES

[1] Stanev I. Generation of Computer Programs for Robot Control Specified through Limited Natural Language Texts. In Elektrotechnica & Electronica. Vol. 5/6 1999г. Pp. 18 – 23.

[2] Hristova, P – Sudying UML in The "Master degree" in Informatics education level, Proceedings, vol.47, book 5.1, Mathematics, Informatics and Phisics, Ruse, 2008

[3] Krastev, G., M. Teodosieva, S. Smrikarova – CASE study of software environments for automatic generation of WEB sites, Proceeding of the International Conference "E-Learning and knowledge society", September 6-8, Ghent, Belgium, 2004

CONTACT ADDRESS:

Valentin Petrov Velikov, senior lecturer,
Angel Kanchev University of Ruse
Dept. of Informatics and IT
cell phone: 0886 011 544,
e-mail: val@ami.uni-ruse.bg
web: http://www.valveliko.com

НЯКОИ ВЪЗМОЖНОСТИ ЗА АВТОМАТИЗИРАНО ГЕНЕРИРАНЕ НА ПРОГРАМИ

Валентин Великов

Русенски университет "Ангел Кънчев"

***Резюме**: Автоматизирано генериране на програми – това е направление в софтуерната индустрия, свързано не само с печелене на време и човешки труд, но и с получаването на синтактично чисти и логически коректни модули. Възможно е едно приложение да се раздели на интерфейс и бизнес-логика, като се създадат или използуват различни съществуващи инструменти за итеративно генериране на желаната система.*

***Ключови думи**: автоматизирано генериране на програми, БДИС, Потребителски интерфейси*

ISSN 1311-9974