AUTOMATIZATION OF COMPUTER BUSINESS GAME AUTOMATON MODEL CONSTRUCTION

Olga Vikentyeva, Alexander Deryabin, Dmitrij Kozhevnikov, Lidiia Shestakova

***Abstract**: This paper suggests an approach, which enables automatization of computer business game automaton model construction. The processes of business game design and conduction occur within the Competence-based Business Game Studio source environment. This fact provides the universality of the domain. Separation of information system into operating and automaton models causes control logic to be concentrated within the automaton model. Due to this fact the control object (operating model) is to have unsophisticated behavior: accepting commands from the automaton and then executing stated commands. This paper verifies the need for automaton model, provides analysis of requirements and describes the design of corresponding program module (interactive visual model editor) of the Competence-based Business Game Studio source environment. This module is used to construct and edit business process models interactively during the stage of business game design.*

***Keywords**: competencies, active learning methods, business-game, business-process, control automat, operating automat, business process modeling.*

***ACM Classification Keywords**: K.3 COMPUTERS AND EDUCATION: K.3.2 Computer and Information Science Education – Information systems education. I.2 ARTIFICIAL INTELLIGENCE: I.2.1 Applications and Expert Systems – Games.*

***Conference topic**: Technology-Facilitated Learning in Complex Domains.*

Introduction

Educational technologies in professional sphere tend to leverage increasingly high number of active training methods. One of the most wide-spread methods is represented by business games (BG). A computer business game (CBG) differs from the conventional one by a considerably lower time required for training, as well as a significantly higher number of business simulations. Currently, CBGs are vastly employed in corporate training, higher and secondary education due to their ability to simulate real business cases and help building approaches to address professional problems. The relevance of this research is evidenced by numerous publications of both Russian and foreign authors [Biggs, 1990], [Draganidis, 2006], [Girev, 2010], [Vikentjeva, 2013], [Bazhenov, 2014].

At the moment the market of computer business games is represented by a large variety of software titles, SimulTrain being amongst the most known and sophisticated, for example.

Commonly, a computer business game is limited to a certain domain (project management, marketing, general management, financing, etc.), thus, making it highly specific. This paper suggest an approach, that enables designing and conducting business games based on the business processes of an organization, which makes the business game universal in regard to the domain. A formalized description of business game domain is acquired with the help of consecutive transformation of business processes models, which makes the automatization of business game design possible.

A project called “Competence-based Business Game Studio” (CBGS) [Vikentjeva, 2013] has been initiated to implement the model of production and management activity. This project is composed of several intertwined subsystems (including design subsystem) and serves the purpose of creating and assessing competences through business games, built in real business processes.

The design subsystem consists of many modules, including the one that automates the transformation of business process models into business game scenarios. This paper considers design and development of such program module.

Leveraging automaton model during business game design

Thus far many of the mathematical models related to automaton programming have found successful implementations across different domains. These models may vary in details but still have much in common.

Automaton models owe their diversity to the broad variety of implementation domains. The latter include mathematical linguistics, logic control, human behavior simulation, communication protocols, formal language, computability and computational complexity theories.

It is a common practice to distinguish control devices and control subjects. Following this concept, the system may be divided into following parts:

– control part (control system) is responsible for behavior logic: transition to the new state of the system, choice of actions to be executed (which depends on current state and incoming signals);
– controlled part (control subject) is responsible for execution of actions, determined by control system, and, possibly, for the creation of certain incoming signals for control part (feedback).

Therefore, the logic of system behavior is concentrated in the control automaton. Control subject is characterized by unsophisticated behavior, i.e. it does not process input signals from the outer environment, rather it merely receives commands from the control automaton. Each command stands for one and only action of the subject [Polikarpova, 2008].

Automaton model is described with the use of algorithm logic scheme language (ALS). A sequence of operators written down in the language of ALS implements and algorithm of business game control.

ALS expression may be represented as following: $L = \{ H, A, P, \omega, \uparrow, \downarrow, K \}$, where H – algorithm start operator, K – algorithm finish operator, P – conditional transition, A – controlling action, ω – unconditional transition, $\uparrow$ – transition start, $\downarrow$ – transition end.

Each operator of the ALS expression implies a command, interpretable by the automaton module (for instance, transition from one state of the business game to another, or conveying control signals from the operating model).

Automatization scheme of the automaton model acquisition may be seen on Fig. 1.

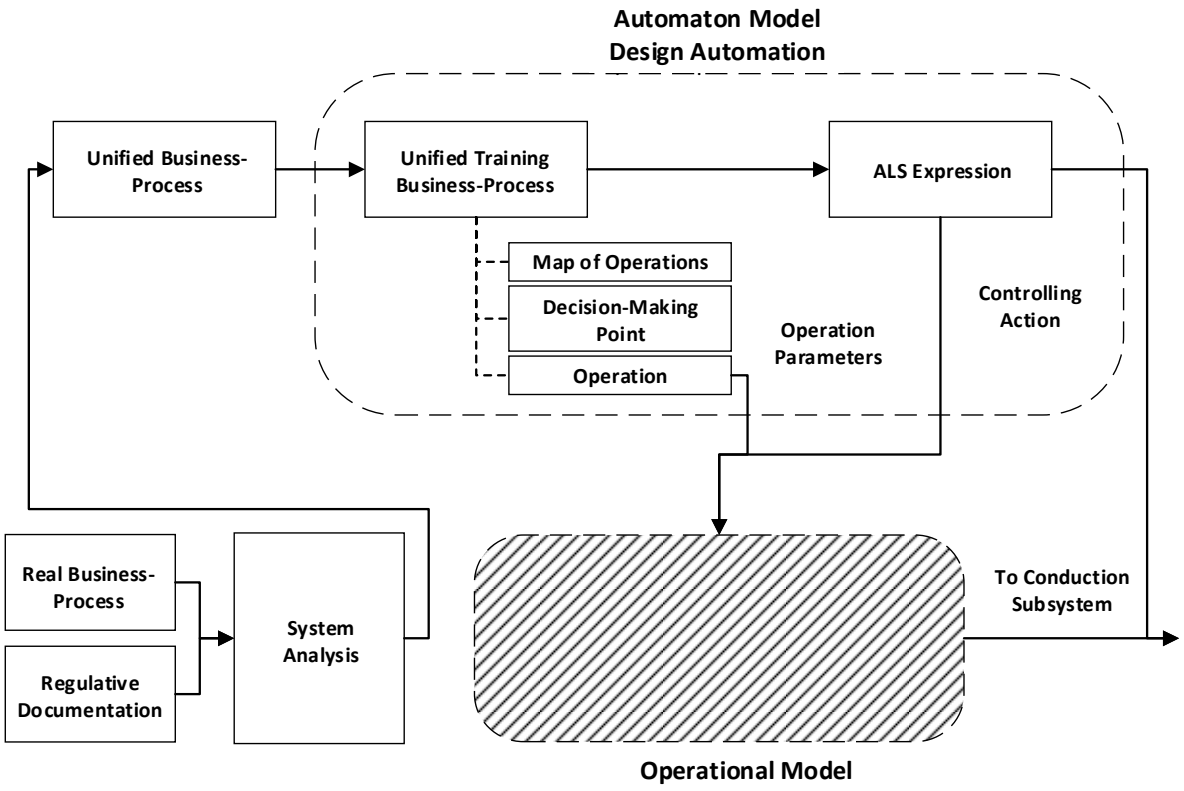

Figure 1. Automatization scheme of the automaton model acquisition

Input data for the model is represented by poorly-formalized data sources (graphical business process models, text descriptions, regulative documentation, etc.). During system analysis of the domain real business process model is transformed into unified business process model (UBP). The latter is comparatively more generalized, due to the fact, that process of UBP construction disregards business operations, specific only to certain companies. The correctness of the models is ensured by compliance with regulative documentation and industry standard business process models.

Unified business process may happen to be considerably sophisticated and include conditional and repetitive business operations besides sequential operations. To convert such model into sequential representation all the cyclic and conditional structures are suggested to be replaced with generalized blocks. Each of these blocks receives a separate Map of Operations of training UBP.

Unified business process requires to be transformed into unified training business process. This is needed to implement additional changes to the original UBP model, related to the process of competence establishment on the trainee's side.

Unified training business process description consists of three metamodels:

– Map of Operations (MO) represents a tree-like structure with all the possible sequences of business operations;
– Operation contains information about operation's parameters (inputs, outputs, controlling information, regulations, mechanisms, time limits, costs, etc.);
– Decision-Making Point (DMP) implements interaction with the player by enabling the latter to choose which operation should be performed next.

When the user makes a decision at this point, the system performs an operation and proceeds to the state, when the user is to choose next operation once again. The amount of possible BG states is finite, due to the fact that repetitive states are not allowed. I.e. once an operation is performed, it becomes unavailable for the user to repeat it.

The task of DMP number minimization is solved by business process decomposition. This enables breakdown of Map of Operations into complementing parts, which contain relatively small amount of Decision-Making Points. Therefore, the automaton model of the business game is going to be represented as a set of strings that store the algorithms of business game control in the form of ALS expressions.

The need for BG automaton model construction automatization is caused by following factors:

– Business game design leverages multimodel representation of business processes [Vikentjeva, 2015].

– Unified business process may happen to be fairly sophisticated, perplexing the derived models and impeding manual business game design.
– All the models are well-formalized, making the transition to automated process possible.

Thus, a contemporary CASE tool needs to be developed, allowing user to interactively build and edit UTBP models in real-time during business game design (interactive visual model editor).

Interactive Visual Model Editor Requirements Analysis

The process of business game design is considered in [Vikentjeva, 2015]. Analysis of TO BE model has enabled formulating a list of functional specifications. According to the latter the editor's purpose is to provide transformation of UBP description into automaton model represented in the form of ALS.

Interactive visual model editor is to provide BG designer with following functionality:

– UBP model creation, utilizing a set of elements, defined in previously developed notation.
– UBP model creation occurs via "drag & drop" technique, or via selecting an element from the context menu, called by right clicking on the work area.
– Each element of the notation has a set of editable attributes.
– Resources of an operation can be set and modified on a separate work screen for operating model editing.
– Generated model complies with syntactic rules of the notation.
– Models can be modified, saved and loaded.
– Syntactically correct UBP model enables generation of UTBP model and grants access to Map of Operations metamodel.
– Generated UTBP models can be modified, saved and loaded.
– Decision-Making Point element can be edited.
– UTBP model is complying with the syntactic rules.
– Syntactically correct UTBP model enables generation of ALS expressions.
– ALS expressions can be modified, saved and loaded.
– Generated ALS expressions are syntactically correct.

Functional specification formulated above have enabled Use Case Diagram (Fig. 2) to be built as well as description of use cases and Activity Diagrams (Fig. 3) to be provided.

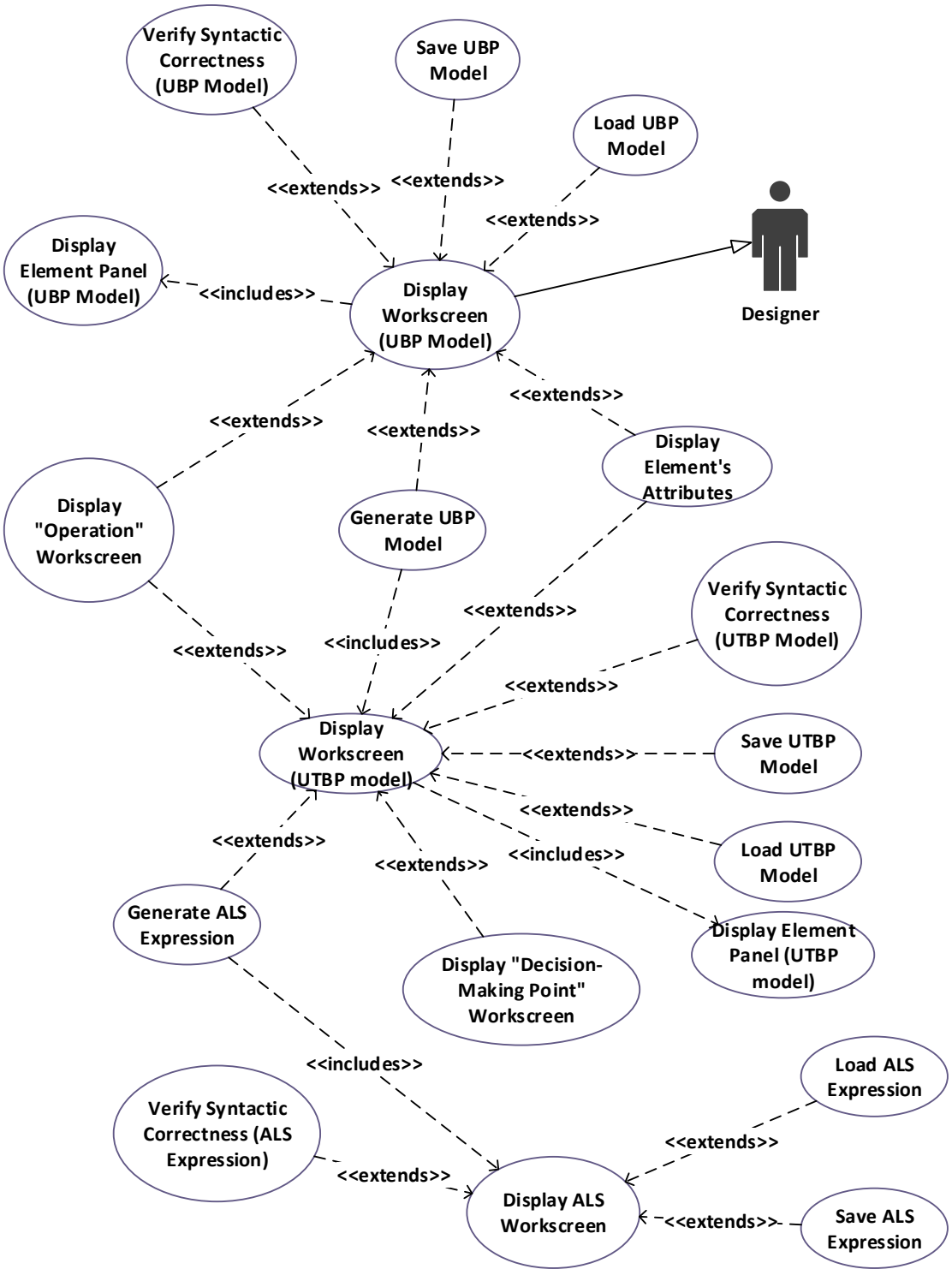

Figure 2. Use Case Diagram

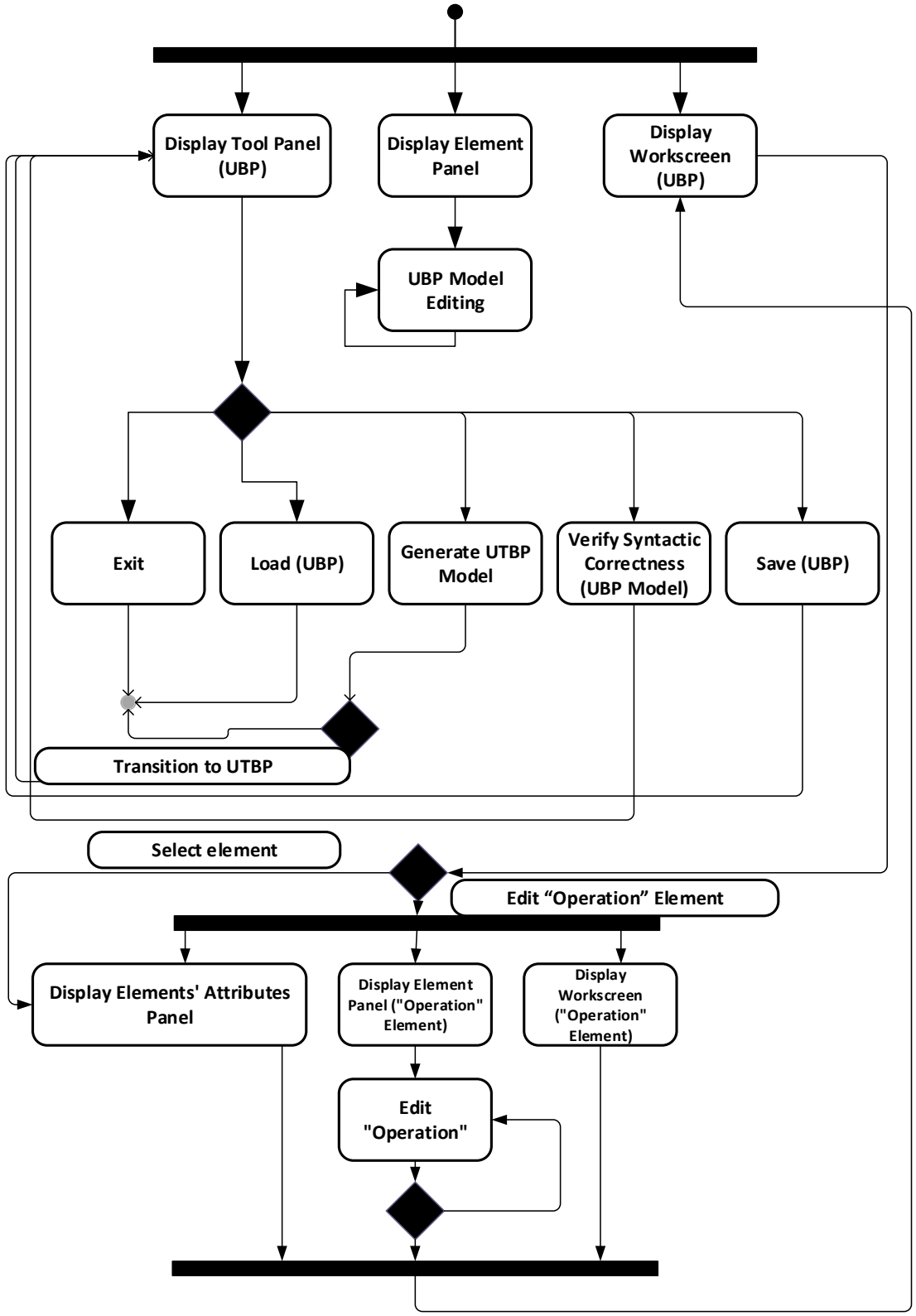

Figure 3. UBP Model Interaction Activity Diagram

Interaction with the UBP models begins with displaying a screen for model editing. Four major areas of this screen may be distinguished:

– *Main Work Area*. Serves the purpose of UBP model construction and display.
– *Toolbar Area*. Contains a set of main tools for model editing.
– *Element Panel Area*. Hosts a set of UBP metamodel elements, which are used for model construction.
– *Element Attributes Panel Area*. Displays and enables editing of selected element's attributes.

After the screen has been displayed, user may proceed constructing or modifying the model by adding notation's elements from the Element Panel.

Modification of the Operation element happens on a separate work screen. This is stipulated by the fact that Operation element possesses a considerable amount of attributes that require to be displayed on the screen. Thus, the stated operation of modification was decided to be carried out on a separate screen.

Modification of the remaining elements does not require additional screens, rather the Attribute Panel is leveraged. The latter is displayed only when an element is in user's focus, once the focus is lost the panel is hidden away.

The user may verify the syntactic correctness of built model at any moment of time. If the verification is successful, the user is given an option to generate the UTBP model from the current UBP model. Algorithm of UTBP model generation may be found in [Vikentjeva, 2014].

The user is also able to save his current model or load a previously built one.

The process of UTBP model construction is very much identical to that one of the UBP model. A special screen for UTBP models is displayed, which contains the same areas: Work Area, Toolbar, Element Panel, and Attribute Panel.

The user modifies the UTBP model in the Work Area using the same Toolbar and respective Panels.

When an element is selected, his attributes are displayed in the Attribute Panel. Operation element modification happens similarly to UBP model. Modification of DMP element is also carried out on a separate screen for the purpose of maintaining readability and convenience of the model.

The user may verify the syntactic correctness of built model at any moment of time. If the verification is successful, the user is given an option to generate the ALS expression from the current UTBP model.

The user is also able to save his current model or load a previously built one.

Interaction with ALS also occurs on an individual screen, which consists of Work Area and Toolbar Panel. The user can modify ALS, then save, or load another expression. The syntactic correctness of an ALS can also be verified.

Interactive Visual Model Editor Design

The analysis of requirements has shown that specific functions of model constructions (UTBP generation, DMP modification, ALS generation, resource modification of an operation, etc.) should be implemented as well as some of the generic functions of visual editors: element rendering and focusing, element alignment, layering, deletion, dragging etc. Due to this fact, it was decided to prepare a prototype using a premade editor with open source code and complete it with necessary changes and additions. Below are listed the prototype requirements:

– Open source code (the code must be enhanced to meet the CBGS requirements);
– Editor should focus in diagram creation (the editor is to be used for business process model creation);
– Editor should be written in C# (this is the main programming language of the development);
– Generated models (diagrams) should be XML-exportable.

"WPF Diagram Designer" has been selected as an editor prototype. This editor provides the necessary generic functionality and uses WPF-technology. Designer's license permits source code modification and use in own applications. Moreover, according to the license, if the proportion of own code outweighs the proportion of source code, the application is allowed to be used for commercial purposes.

It is generally viable to use architectural patterns (design patterns) during the process of application development. Object-oriented patterns commonly provide a concept of classes and their interactions with each other [Gamma, 2001].

MVVM pattern was considered to be feasible for editor development, since the specificity of the latter implies a direct linkage between data model and data representation.

MVVM (Model – View – View model) consists of three elements:

– Model – reflect application's business logic;
– View – visualizes data model;
– View model – is a view abstraction, alters the view upon suffering any changes.

Windows Presentation Foundation (WPF) is believed to be the most popular technology that supports MVVM [Gossman, 2005]. It is also used as a basis for the considered editor.

WPF can be successfully implemented when constructing both autonomous desktop applications and web-based browser applications.

The basic editor is quite limited in terms of required functionality, supporting only generic operations. Thus alterations to basic classes as well as completely new classes will be required for application development.

Below are given the limitations of basic “WPF Diagram Designer”:

– Naturally, the required layer of business logic is completely missing (elements and attributes of developed notation are not implemented);
– Database interface is not present (mechanisms of database uploading and downloading for both data model and visual representations) must be implemented;
– Element’s attributes cannot be edited;
– Metamodel transitions are not implemented (the user should be able to edit, generate and switch between relevant models).

Thus the graphical user interface of the editor should undergo following changes (Fig. 4):

– Add database interaction dialogues;
– Develop sets of elements for each of the models;
– Add buttons for UTBP and ALS generation;
– Implement Attribute Panel.

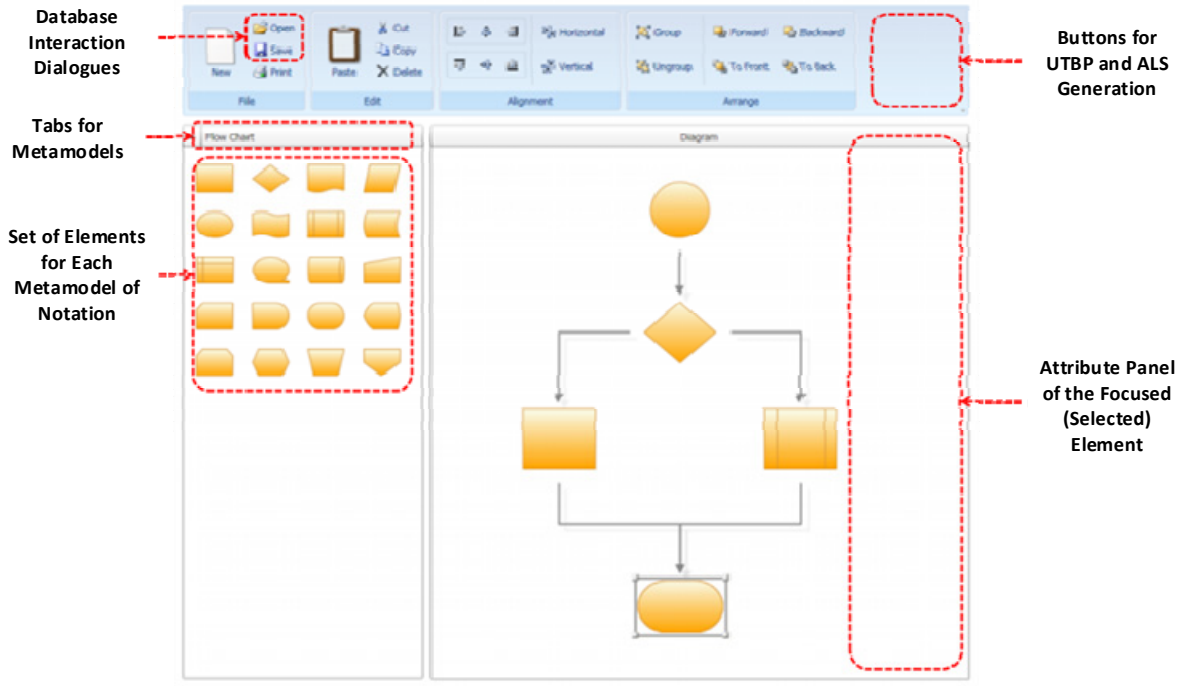

Figure 4. Changes to the Basic GUI

This screen stencil has been used during the development of own visual model editor. Taking stated requirements into account, the GUI of the editor has received the following form (Fig. 5).

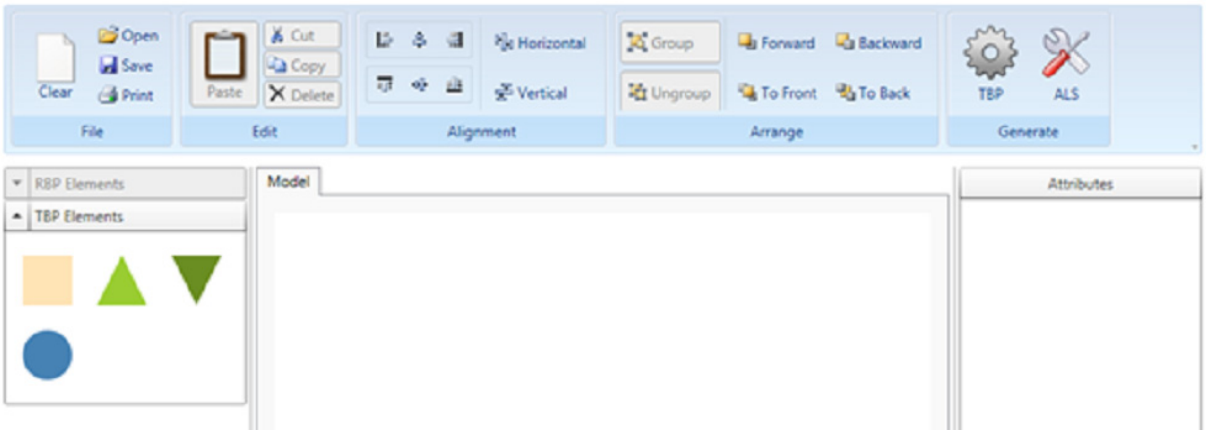

Figure 5. Visual Model Editor GUI

Fig. 6 shows Class Diagram of the editor. The diagram contains unaltered basic classes, modified and completely new classes.

Each class of the model corresponds to an element of the notation. All data model classes are inherited from the parent class BaseLogic. This has been done in order to implement common methods of attribute displaying.

Class Window1 contains XAML markup of the editor, accompanied by the resource and style dictionaries. This class is required for data model visualization and, basically, implements GUI of the application. It also hosts linkages to view model and data model. XAML markup implements the previously described screen stencil. Window1 is one of the basic classes, which belongs to the original assembly of the editor. The XAML markup was changed during the development, own style and resource dictionaries were implemented as well.

Class App is inherited from the standard class Application, it implements enumeration of all used in Window1 XAML markup dictionary resources.

Class ProgressBarPopup is called to visualize processing of slow procedures. The main purpose of this class is displaying a progress bar during procedure execution.

Table 1 contains all the classes of view model with description and relevant commentaries.

Considered classes have been used in the latest version of interactive visual model editor to implement attribute panel, element panel, graphical elements, UTBP and automaton model generation.

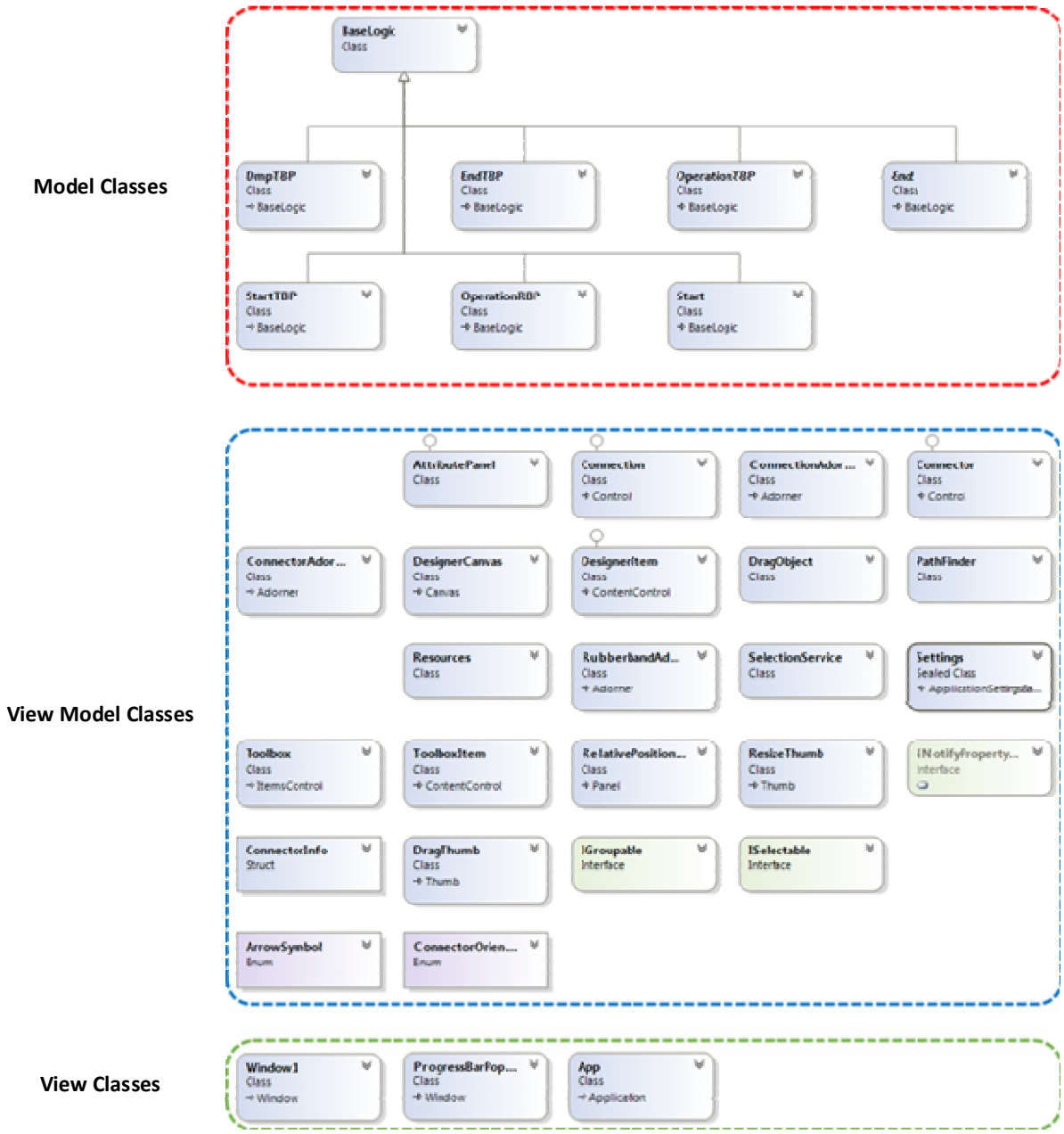

Figure 6. Class Diagram

Table 1. List of New and Modified Classes

Class Name	Description	Changes
AttributePanel	Displays and implements the logic of Attribute Panel.	This class does not belong to the basic assembly of the designer and has been composed from scratch. The class displays attributes of a logic element, which is bound to the selected graphical element on work screen. It also implements messaging mechanism and, therefore, conveys changes of interface to the data model.
ToolboxItem	Implements interaction with Element Panel items.	This basic class had to be modified so that Element Panel items contain logic object, besides graphical object.
DesignerItem	One of the main classes of the application. Displays and implements the logic of editor's graphical elements.	Changes to this class include following: – Linkage between graphical objects and logic objects. – Display of logic object's name on top of graphical object. – Notification mechanism that warns view and data model upon. Modification of class intance's attributes.
DesingerCanvas	Implements logic of interaction with editor's graphical elements.	Modifications to this class mainly relate to invocation of class methods, responsible for element focusing.
SelectionService	Implements the mechanism of element focusing.	Changes of this class include developments in invocation methods of Attribute Panel class.
DesignerCanvas.Commands	Implements interface commands of class DesignerCanvas, which are called via Toolbox Panel.	Modifications of this class are related to the Save and Load procedures as well as UTBP and ALS generation. All commands had to be set up to include cooperation with logic objects, commands of UTBP and ALS generation also had to be developed from scratch.

Conclusion

The paper suggests an approach, which enables automatic acquisition of UTBP model and automaton model of computer business game based upon multimodel representation of business processes.

The need for BG automaton model construction automatization is stipulated by following factors:

– mulimodel representation of business processes;

– complexity of process models impede manual design of business game;

– well-formalized models, which are used for business game design.

"WPF Diagram Designer" has been chosen to be used as a prototype of model editor. The Designer provides generic functionality of working with model elements and utilizes WPF technology. The license also permits source code to be leveraged in own developments. The GUI of the basic editor has received following modifications:

– database interaction dialogues have been added;

– for each metamodel a set of elements has been developed;

– buttons for UTBP and ALS generation have been added, corresponding algorithms have also been implemented;

– Attribute Panel has been added.

Thus, a contemporary CASE-tool has been developed, which enables construction and modification of UTBP model in real-time during the process of BG design (interactive visual model editor).

Bibliography

[Biggs, 1990] Biggs William D. Introduction to Computerized Business Management Simulations // Guide to Business Gaming and Experiential Learning – London : Nichols/GP Publishsing, 1990.

[Draganidis, 2006] F. Draganidis. Chamopoulou P and Mentzas G An Ontology Based Tool for Competency Management and Learning Paths 6th International Conference on Knowledge Management I-KNOW 06, Special track on integrating Working and Learning, 6th September 2006, Graz, (2006).

[Gossman, 2005] Gossman John. Introduction to Model/View/ViewModel pattern for building WPF apps // MSDN Blogs. [URL: http://blogs.msdn.com/b/johngossman/archive/2005/10/08/478683.aspx] [26.04.2015].

[Vikentyeva, 2014] Vikentyeva O.L., Deryabin A.I., Shestakova L.V. Algorithms of Automate Model Constriction for Business Game Execution Subsystem // International Journal "Information Models and Analysis". 2014. Vol. 3. No. 3. P. 271-279.

[Bazhenov, 2014] Баженов Р.И. Об организации деловых игр в курсе «Управление проектами информационных систем, Научный аспект, 2014. – т.1, №1, С. 101-102.

[Vikentyeva, 2013] Викентьева, О.Л. Концепция студии компетентностных деловых игр [Электронный ресур] / О.Л. Викентьева, А.И. Дерябин, Л.В. Шестакова // Современные проблемы науки и

образования. – 2013. – № 2; [URL: http://www.science-education.ru/108-8746] (дата обращения: 03.04.2013).

[Vikentyeva, 2015] Викентьева О.Л., Дерябин А.И., Кожевников Д.Д., Красилич Н.В., Шестакова Л.В. Подсистема проектирования информационной системы для проведения деловых игр // В кн.: Технологии разработки информационных систем: сборник статей международной научно-практической конференции. Таганрог : Издательство ЮФУ, 2015. С. 27-32.

[Vikentyeva, 2015] Викентьева О.Л., Дерябин А.И., Шестакова Л.В., Лебедев В.В. Многомодельный подход к формализации предметной области // Информатизация и связь. 2015. № 3. С. 51-56.

[Gamma, 2001] Гамма Э. Приемы объектно-ориентированного проектирования. Паттерны проектирования / Э. Гамма, Р. Хелм, Р. Джонсонс, Дж. Влиссидес. СПб: Питер, 2001.

[Girev, 2010] Гирев, П.Е. Инновационные подходы к использованию интерактивных моделей в обучении / П.Е. Гирев, О.И. Мухин, О.А. Полякова // Дистанционное и виртуальное обучение, 2010. – С.84.

[Polikarpova, 2008] Поликарпова Н. И., Шалыто А. А. Автоматное программирование. СПб, НИУ ИТМО, 2008.

Authors' Information

Alexander Deryabin – National Research University Higher School of Economics, City of Perm, Perm, Russia, e-mail: paid2@yandex.ru.

Major Fields of Scientific Research: General theoretical information research, Multi-dimensional information systems

Olga Vikentyeva – National Research University Higher School of Economics, City of Perm, Perm, Russia, e-mail: oleovic@rambler.ru.

Major Fields of Scientific Research: General theoretical information research, Multi-dimensional information systems

Lidiia Shestakova – National Research University Higher School of Economics, City of Perm, Perm, Russia, e-mail: L.V.Shestakova@gmail.com.

Major Fields of Scientific Research: General theoretical information research, Multi-dimensional information systems

Dmitrij Kozhevnikov – National Research University Higher School of Economics, City of Perm, Perm, Russia, e-mail: mefaze@yandex.ru.

Major Fields of Scientific Research: General theoretical information research, Multi-dimensional information systems