Advances in Intelligent Systems Research, volume 169
VIth International Workshop 'Critical Infrastructures: Contingency Management, Intelligent, Agent-Based, Cloud Computing and Cyber Security' (IWCI 2019)

The Application of the Knowledge-Based Systems Development Platform for Creating Scenario Analysis Support Tools

Alexander Pavlov
Matrosov Institute for Systems Dynamics and Control Theory of Siberian Branch of the Russian Academy of Sciences
Irkutsk, Russia
asd@icc.ru

Alexander Stolbov
Matrosov Institute for Systems Dynamics and Control Theory of Siberian Branch of the Russian Academy of Sciences
Irkutsk, Russia
stolboff@icc.ru

***Abstract*— The article discusses the problems of the knowledge-based systems development platform application for creating scenario analysis tools and their approbation for the decision support in the infrastructure logistics domain studies and for the assessment of environmental impacts based on the ecological-economic mathematical regional model. The decision support process is defined by a set of problem-oriented knowledge bases that are consistently interpreted. The platform provides the basic functionality for the knowledge bases design process: creation of subject domain model and rules, reasoning, data storing. However, to meet the targets problem-oriented and several auxiliary components are additionally developed. The proposed components extend the basic platform capabilities by providing the following: new data type for storing N-dimension data; the information representation in map view based on OpenLayers library, special scenario editor component for defining scenario analysis properties and workflow.**

Keywords—knowledge-based system, scenario analysis, component-based software system development

I. INTRODUCTION

Knowledge-based system (KBS) is the special kind of software tool utilizing explicit knowledge representation and reasoning. As a rule, KBS is focused on working in poorly formalized domains, and as a consequence, KBS specificity is the high degree of variability throughout the life cycle. So, KBS development and implementation cost can be significant due to a lot of changes of various kinds such as error correction, data model evolution and functionality expansion. Thus, there is the actual and challenging problem related to reducing the cost of creating, modifying and upgrading of applied KBS. To address the represented problem the authors are creating a specialized platform for KBS development, that originally includes methods and tools aimed at reconfiguring its architecture without significant efforts on rewriting the source code of applied KBS. The platform architecture is proposed in the article [1]. A well-known component-based approach [2, 3] is utilized for the platform development.

The basic platform functionality was tested for the development of KBS for decision support in the infrastructure logistics domain (KBS4IL) [1] and for agent-based modeling driven by declarative specification of implementation process [4]. Experience in application and maintaining of applied KBS has revealed the new directions for platform capabilities development. It was found that scenario analysis related problems are some of the most important and frequently repeated tasks that are required special attention.

So the purpose of the current article is to discuss the platform capabilities for creating scenario analysis support tools. First, we present the overview of the platform and introduce new components for extending the supported data types (N-dimension data and geographical coordinates) that according to our experience could further benefit the development process of the applied KBS in general and facilitate the work with scenario editor component in particular. Then based on the brief overview of exciting scenario analysis approaches we propose architecture of scenario editor component. Scenario analysis features were tested for KBS4IL refinement and for new (for the platform application) subject domain: scenario analysis based on the “Region” model set [5].

II. THE OVERVIEW OF SOFTWARE PLATFORM FOR KNOWLEDGE-BASED SYSTEMS DEVELOPMENT

A. The platform architecture

According to the platform architecture [1], particular KBS is a collection of elements of only two types: management sub-system and the set of components. To solve the problem of components integration and intercomponent data exchange each component, in addition to the problem-oriented functions, must ensure the implementation of the *IComponent* interface for the metadata processing and the object/component handling. The management sub-system is an element of the platform architecture that exists as a single instance (Fig.1).

The management subsystem provides the creation, modification and interpretation of the applied KBS specification. The main part of the KBS specification is the definition of its operations, each of which is a sequence of calls to the problem-oriented functions of the platform components and service elements, such as conditional operator, loop operator and block operator. Thus, the control flow and data flow of the KBS

operation are defined only by a set of the calls to problem-oriented functions of the platform components, the values of parameters and the service elements. The management subsystem provides the specialized workflow editor for the visual construction of the definitions of KBS operations on top of the proposed set of elements (Fig.2.). The only requirement of its target user is a programming skill of basic level. The proposed process of constructing the definition of applied KBS operation assumes that all the problem-oriented or domain-specific rules and algorithms of the decision-making process are not included into the operation definition, but are implemented either in the form of the corresponding problem-oriented component functions or developed knowledge bases (KBs).

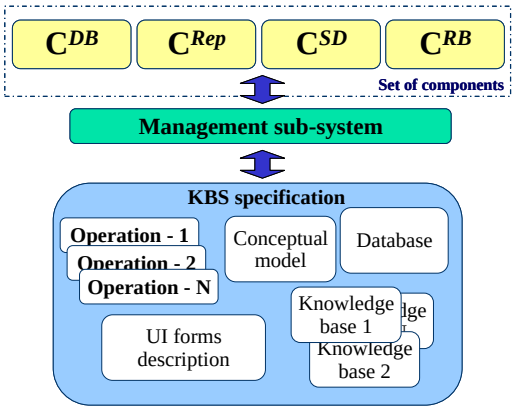

Fig. 1. The platform architecture

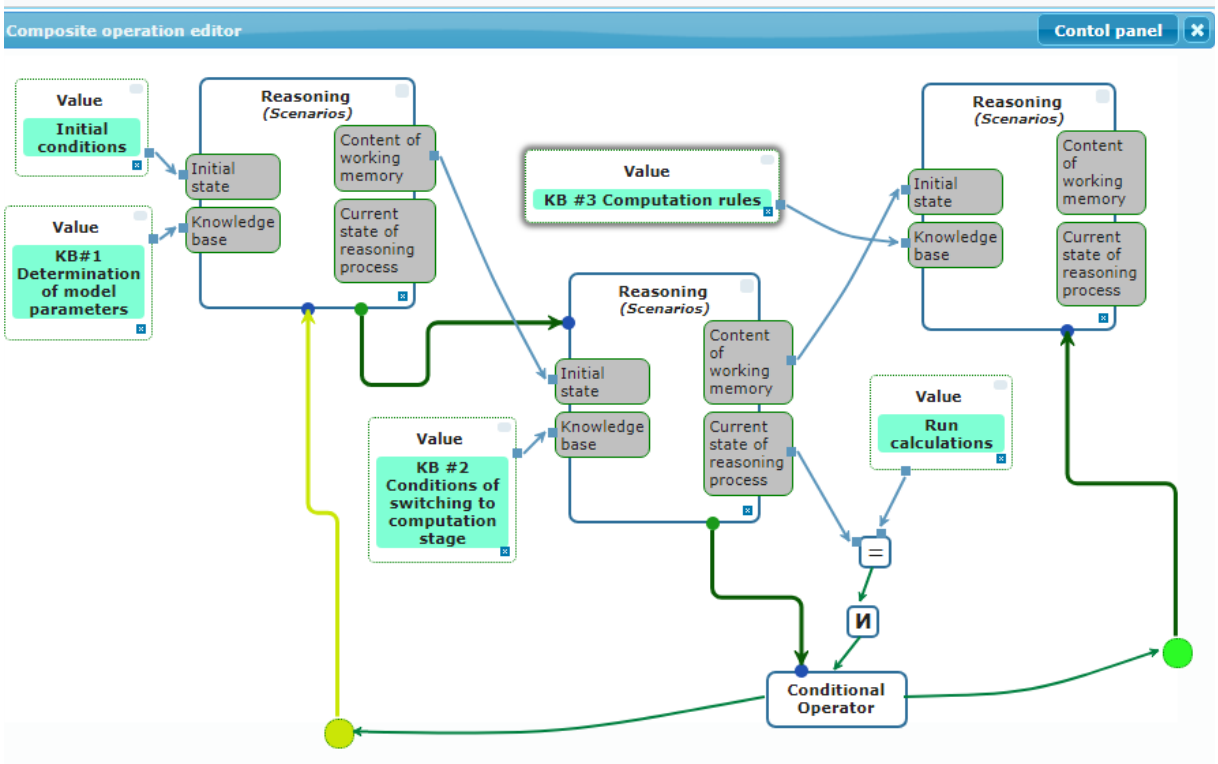

Fig. 2. The workflow editor

There are two ways to create platform components: direct implementation with some programming language and creating a wrapper around an existing software system, for example, using the "facade" design pattern. Currently, the management subsystem supports the following set of protocols for invoking the functions of the components: HTTP, SOAP and WebSocket.

To implement the basic functionality of the platform uses the following set of components: C^{DB} – the data control component, C^{Rep} – the data representation and editing component; C^{SD} – the subject domain model design component; C^{RB} – rule-based reasoning component. The main task of the C^{DB} is to encapsulate all operations required for data manipulations (for example, CRUD and storing). For requested data the C^{DB} provides their (meta) description in terms of unified interface. C^{Rep} provides the implementation of standard platform user interface forms and controls. The component implements a data model that stores information about the developed forms of the user interface (UI), containing name, description, list of used UI controls. The C^{SD} has the following functionality: the formation of a hierarchical structure of concepts from abstract to more concrete ones; the definition of the properties and relations; the creation of instances of concepts. The description of relations is carried out by constructing a semantic network, where the vertices are concepts and the relationships are edges. C^{RB} has the following functionality: creating knowledge bases on top of the selected conceptual model; the formation fact templates list based on concepts of the conceptual model; visual construction of the rules in the form of

"IF conditions THEN actions"; creating initial conditions; generation of knowledge base code in Drools, Jess and Clips format and its inference; creation the concept instances from inference results. It should be noted that C^{RB} provides the ability to invoke an imperative code during the reasoning process by the special type of action: "call an external method".

The platform was implemented in the form of a web application using the "thin" client technology. So the architecture of typical platform component consists of client and server parts. The client part provides user interface whereas server part implements data processing methods related to the component functionality. In cases when a component uses existing software tools, libraries or services for implementing its functionality the server part can be further divided into software modules available to users and modules with limited access. The GUI of platform and components is built on the top of HTML, CSS and JavaScript languages and popular JavaScript libraries such as jQuery and jQueryUI. The data storage functionality is implemented by well-known PostgreSQL DBMS, while communications is based on HTTP/HTTPS, SOAP and Websocket protocols.

As stated above the basic functionality of the platform is provided by 4 main components (C^{DB}, C^{Rep}, C^{SD}, C^{RB}) that currently are designed and tested. However, depending on the particular purposes of KBS creation it may be required to facilitate some tasks by developing new auxiliary components or reusing methods of existing software tools. According to the platform architecture access to parameters and methods of some problem-oriented system (POS) is done with the use of facade pattern for representing POS capabilities in terms of unified interface (*IComponent*). As a rule, new functionality is related to extending the supported data types, utilizing or developing specific problem-solving methods.

In next section new data types processing components are introduces: the N-dimension data handler (C^{N}) and web-map representation (C^{Map}). Further in the paper the scenario analysis component (C^{Sc}) as an example of problem-solving component is described.

B. Examples of Auxiliary Components

As an example, let us consider the **N-dimension data handler component** (C^{N}). The C^{N} provides functionality for creation, retrieving, modification and storing information of the multidimensional datasets, including UI for data input/output operations. The data model of C^{N} supplies possibility to define the dimension structure of the each dataset, herewith support of rectangular and jagged dataset types is provided. The distinguishing feature of C^{N} is a possibility to setup domain-specific "measures" for dataset dimension based on the conceptual model defined with C^{SD}. The implementation of C^{N} data model within PostgreSQL DBMS is based on the well-known two level recursive structure, which provides opportunity to define N-dimension dataset. The dataset consists of set of data series it owns, and each of data series also can own set of data series, etc.

In the context of applied KBS development, the actual problem is the task of improving the quality of decision-making based by the application of maps and satellite images as UI elements for displaying the dynamics of domain-specific processes and location of considered objects. To solve this problem, it was proposed to develop the **web-map representation component** (C^{Map}) with the following functionality: creating and editing point-based and poligon-based representations of domain entities; determining the distance between points, searching for objects in a certain area. The popular open-source JavaScript library OpenLayers [6] was chosen as the basis for C^{Map}. The component was created by developing a specialized software wrapper that automates the process of using a corresponding subset of the OpenLayers library methods. The process of displaying domain-specific entities on the map is based on the processing of properties containing geodata (point, path, polygons and etc.), herewith these properties can be automatically determined from corresponding data types of the PostgreSQL DBMS and specified by the user during the creation of the conceptual model.

III. The Scenario Analysis Problem

A. The Scenario Type Description

Scenario analysis helps us formalize one or more possible answerers to questions about the future. As a rule, scenario analysis involves the definition of a specific state of the world, and related impact on variables being analyzed, to capture and quantify outcomes.

As stated in literature related to scenario analysis and development a systematic use of scenarios for clarifying thinking about the future started after the World War II as a method for military planning in 1950s at RAND Corporation [7]. After that scenario methodology was used in many subject domains and a lot of valuable techniques and tools were created. The extensive growth of scenario related approaches (scenario development, scenario analysis, scenario planning) has led to the emergence of some confusions about used terms and methods. To address this problem a lot of studies were devoted to summarize and harmonize the obtained results in scenario domain [7-12].

Recognizing the efforts undertaken by many research groups our understanding of scenario development techniques is based mainly on approaches described in [11]. There more than 20 scenario types are considered: 8 main categories along with their variations. Authors claimed that proposed category set covers the most of existing scenario development approaches. The category set includes the following: "judgment, "baseline, "elaboration of fixed scenarios", "event sequences", "backcasting", "dimensions of uncertainty", "cross-impact analysis" and "modeling".

All these scenario categories and types [11] can be classified depending of their basis as judgment or quantification. So in that context using the proposed platform for creating scenario analysis tools seems to be an adequate solution. As to meet the targets of scenario analysis the developer has to utilize various methods based on domain modeling, knowledge formalization and specific problem-oriented solvers processing both quantitative and qualitative information.

The main scenario related problem solving methods for selected subject domains (KBS4IL and "Region" model set) is based on varying the inputs and/or the structure of the models that generate the alternatives. So according to [11] the closest scenario type is "modeling". More precisely following the [11] definitions there are 3 groups of possible changes:

- variations of the exogenous variables values;
- variations of the parameters that define the effect of variables on one another;
- variations of the variables in the model itself, including both values and the actual structure of the model.

To facilitate the scenario analysis the platform allows one to explicitly describe these 3 variation types in general form in scenario analysis conceptual model (CM^{Sc}) using the C^{SD} component. Next depending on particular subject domain experts can further define specific of domain changes based on CM^{Sc} by inheriting mechanism.

B. The Scenario Editor Component Architecture

Despite the fact that each subject domain makes its own specifics to the "modeling" scenario analyses process it is possible to reveal some common features that would be exploited by platform:

- the scenario analyses based on some methodology that defines the instructions to perform for solving domain related tasks;
- the scenario analyses process is implemented on the top of the platform facilities and uses its basic and auxiliary components;
- during the scenario analyses process a variety of 3rd party analytical tools are intensively utilized.

So from the implementation point of view, the scenario analyses process can be considered on the three levels: macro, meso, and micro. Micro-level corresponds to particular rule and/or utilization of 3rd party analytical tool method that is wrapped by the rule and is considered as elementary operation. Meso-level is associated with some knowledge base containing micro-level rules. At last on the macro-level, the overall scenario analysis process called SA-session are defined. Here algorithm describing appropriate data and control flows is presented and is created as a combination of executing functions of C^{RB} and C^{Sc} (described below), choices (if-then-else statements) as well as collection-controlled, condition-controlled and count-controlled loops.

Under these considerations, the scenario editor component (C^{Sc}) is implemented on the top of C^{SD}, C^{RB}, C^{Rep}, C^{N} components. It uses C^{SD} for defining scenario properties in the concept-attribute-relation form, C^{N} for data manipulations, C^{RB} for creating scenario related knowledge base (micro- and meso-level), C^{Rep} for designing GUI as well as management sub-system facilities for describing scenario algorithm (Fig.2.) in the form of workflow [13]. Currently on the prototyping phase of C^{Sc} component development "alternator generator" (C^{Sc}_ag), "make step of scenario analysis" (C^{Sc}_step) and "knowledge base restriction" (C^{Sc}_limit) problem-oriented functions are designed.

The C^{Sc}_ag function allow one to create patterns for creating collections of alternatives. For example, the user can define the range and the step for generating numerical values or explicitly form the collections of some values including object ones. As the main efforts were devoted to the development of the knowledge representation and reasoning functionality of the platform currently only a simple set of patterns is presented but it will be surely expanded to support data generators like in any simulation software. The C^{Sc}_ag uses functions of C^{SD} and C^{N} components.

With the use of the CSc_limit function, users can set a limitation on how the meso-level scenario knowledge base during SA-session can be processed. Virtually this function allow one to describe scenario analysis patterns with the typed slots for particular kind of knowledge bases characterized by a set of user-defined properties.

The C^{Sc}_step function is implemented on the top of C^{RB} component facilities. The C^{Sc}_step allows users to define some subset of knowledge base facts that can store the SA-session sate or other useful information to be retrieved for decision-making on the macro level of scenario analysis process depending on the values

IV. ILLUSTRATIVE EXAMPLES

A. Scenario Analysis in the Infrastructure Logistics

The first illustrative example describes refinement of knowledge-based system for decision support in the infrastructure logistics domain (KBS4IL) partly presented in [1]. The main objective of KBS4IL is to support research of the functioning and allocation of interrelated, multi-scale infrastructure facilities utilizing a variety of analytical tools. The research process is carried out in the form of a multivariate scenario analysis and is defined by the original methodology [14]. The KBS4IL exploits a conceptual model containing information about infrastructure logistics domain (vehicle, terminal, stakeholders, route, etc.) and description of applied methods and tools (problem statement descriptions, solvers). The initial information and the results of decision making are stored as instances of concepts. KBS4IL utilizes the platform ability for rule-based reasoning. Also externally defined problem-oriented methods can be

included to the reasoning process via the platform integration facilities. For example, special tools for the pathfinding [15] , the problem of location [16], the problem of organizing communications [17]

With the use of C^{Sc} component the user can define specialized knowledge base type for infrastructure logistics – KBIL, containing additional information about level of details on which infrastructure objects are represented (macro, meso, micro) and the kind of the task under consideration (creation of a transport and logistics assessment of the current region state, identification of problem situations in the region, solution searching for the identified problems, et.al.).

The specific SA-session can be considered as workflow with KBIL as the only type of knowledge base used in C^{Sc}_step function. Currently, depending on the particular characteristics of KBIL the two main SA-session for studying the infrastructure logistics problems are created: reference-information mode and research mode. Reference-information mode is designed for interviewing experts and creating narrative scenarios in terms of subject domain. Research mode involves a comprehensive assessment with the use of externally-defined analytical tools for the solution of infrastructure logistics problems.

B. Scenario Analysis Based on "Region" Model Set

Another example is concerned with scenario analysis with more numerical values than KBS4IL is scenario analysis based on the "Region" model set [5] that includes various variants of mathematical ecological-economic, medical-ecological and socio-ecological-economic models, as well as ecosystems of Lake Baikal. These models have been developed for 40 years at the ISDCT SB RAS (see the author's affiliation) together with other scientific groups from Irkutsk, Angarsk and Pereslavl-Zalessky (Russia), Ulaanbaatar (Mongolia). These years of work has led to the emergence of original "Region" methodology that is intended for organizing interaction between specialists of different profiles (mathematics, domain experts, programmers) during the interdisciplinary study of complex unique objects and solving problems of forecasting and assessing the influence of anthropogenic factors on the environment and social sphere, where mathematical model is a key tool for analysis and synthesis of the research group experience and knowledge.

The C^{SD} component is utilized for describing mathematical model and scenario terms (parameter, variable, equation, etc.); domain terms (factor, measure, indicator, causal relation); research session state (scenario, survey stage et.al.). The C^{RB} component is utilized for designing rules for the model structure identification (nomenclature of variable); parameters value identification based on a set of specialized methods; scenario design based on predefined templates. At last externally defined problem-oriented tools implementing numerical methods for root-finding, integration as well as specialized parameters identification methods can be included.

V. CONCLUSIONS

The scenario analysis related problems were chosen as one of the promising directions for the further engineering of knowledge-based systems development platform. The scenario analysis paradigm is implemented on the top of conceptual modeling and rule-based reasoning functionality provided by the platform along with new auxiliary components firstly introduced in the current paper: the N-dimension data handler component (C^{N}); the web-map representation component (C^{Map}); the scenario editor component (C^{Sc}). Herewith the C^{Map} is an example of a facade pattern utilization way for new component creation, whereas the C^{N} and C^{Sc} have been developed from scratch.

In current version of the platform the particular scenario analysis session is defined in the form of macro level algorithm describing appropriate data and control flows with the use of the following primitives: executing methods of C^{Sc} (including rule-based reasoning), choices and loops.

The proposed results can be considered as the starting point in our research concerning the platform application for creating scenario analysis support tools. The future plans are to progressively extend the supported scenario types and as a consequence to introduce the facility of multi-typed scenario creation. As for particular applications presented in the paper it is interesting and challenging to converge two considered domains by merging their conceptual models and used analytics tools.

ACKNOWLEDGMENT

The reported study was partially supported by RFBR projects 18-07-01164, 18-07-00560.

REFERENCES

[1] O. A. Nikolaychuk, A. I. Pavlov, and A. B. Stolbov, "The software platform architecture for the component-oriented development of knowledge-based systems, " in proceedings of the 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO), K. Skala, Ed., 2018. pp. 1234–1239.

[2] G. T. Heineman, and W. T. Councill, Component-Based Software Engineering: Putting the Pieces Together, Boston, MA: Addison-Wesley, 2001.

[3] O. A. Nikolaychuk, and A. I. Pavlov, "Application of component-based software development approach for computer-aided investigations," Herald of computer and information technologies, No. 4, pp. 23–32, 2010 ["Primenenie komponentnogo podhoda dlja sozdanija sistemy avtomatizacii issledovanij," Vestnik komp'iuternykh i informatsionnykh tekhnologii, No. 4, pp. 23–32, 2010] (In Russian).

[4] O. A. Nikolaychuk, A. I. Pavlov, and A. B. Stolbov, "Web-Oriented Software System for Agent-Based Modeling Driven by Declarative Specification of Implementation Process," in Proceedings of the 3rd Russian-Pacific Conference on Computer Technology and Applications (RPC), 2018, pp. 1–5.

[5] V. Gurman, and V. Baturin, "Ecological-economic model of the region: information technology, forecasting and optimal control," Mathematical Modelling of Natural Phenomena, vol. 4, No. 5, pp. 144–157, 2009.

[6] T. Gratier, P. Spencer, and E.Hazzard, OpenLayers 3 Beginner's Guide, Birmingham: Packt Publishing Ltd., 2015.

[7] M. Amer, T. U. Daim, and A. Jetter, "A review of scenario planning," Futures, vol. 46, pp. 23–40, 2013.

[8] P. Schwartz, The art of the long view: planning for the future in an uncertain world, NY: Currency Doubleday, 1991.

[9] M. Godet, and F. Roubelat, "Creating the future: The use and misuse of scenarios," Long Range Planning, vol. 29, No. 2, pp. 164–171, 1996.

[10] Ph. Van Notten, Scenario development: a typology of approaches, OECD: Think Scenario, Rethink Education, 2006, pp. 69–84.

[11] P. Bishop, A. Hines, and T. Collins, "The current state of scenario development: an overview of techniques," Foresight, vol. 9, No. 1, pp. 5–25, 2007.

[12] M. J. Spaniol, and N. J. Rowland, "Defining scenario," Futures & Foresight Science, vol.1, No. 1, pp. 1–13, 2018.

[13] N. Russell, A. H. M. Hofstede, D. Edmond, and W. M. P. Aalst, Workflow Data Patterns, Brisbane: Queensland University of Technology, 2004.

[14] I. V. Bychkov, A. L. Kazakov, A. A. Lempert, D. S. Bukharov, and A. B. Stolbov, "An intelligent management system for the development of a regional transport logistics infrastructure," Automation and Remote Control, vol. 77, No. 2, pp. 332–343, 2016.

[15] A. L. Kazakov, and A. A. Lempert "An approach to optimization in transport logistics," Automation and Remote Control, vol. 72. No. 7, pp. 1398–1404, 2011.

[16] A. L. Kazakov, A. A. Lempert, and D.S. Bukharov, "On segmenting logistical zones for servicing continuously developed consumers," Automation and Remote Control, vol. 74, No. 6, pp. 968–977, 2013.

[17] A. L. Kazakov, and A. A. Lempert, "On mathematical models for optimization problem of logistics infrastructure," International Journal of Artificial Intelligence, vol. 13, No. 1, pp. 200–210, 2015.