The workflow component of the knowledge-based systems development platform*

A.I. Pavlov[1][0000−0002−7753−7514], A.B. Stolbov[1][0000−0001−6513−7030], and A.S. Dorofeev[2][0000−0002−8498−3301]

[1] Matrosov Institute for System Dynamics and Control Theory of Siberian Branch of the Russian Academy of Sciences, 134 Lermotov st., Irkutsk, Russia
{asd,stolboff}@icc.ru
http://http://idstu.irk.ru
[2] Irkutsk National Research Technical University, 83 Lermotov st., Irkutsk, Russia
dorbaik@istu.edu
https://www.istu.edu

Abstract. The paper is concerned with the workflow component of the knowledge-based systems development platform. The proposed workflow component supports the creation process of the platform composite operations. These operations can be used, for instance, to define assembling models of functional components of the platform or to describe procedural behavior of the action part of the particular rules of the knowledge base. The issues of the data flow view of the workflow components are considered. The introduced data flow view is designed to explicitly reflect the features related to the specificity of information processed in knowledge base systems like concepts or instances. The proposed data flow view let the users extend the workflow with additional semantic. This feature can facilitate the knowledge base system design process. As an illustrative application, the development process of the knowledge-based system for decision support in the infrastructure logistics domain (KBS4IL) is considered. The exemplary workflow related to the computational component of KBS4IL is presented.

Keywords: knowledge-based systems · workflow · infrastructure logistics.

1 Introduction

Knowledge-based technologies can play a significant role in the modern informational era. The big data availability, variety of digital platforms for automated monitoring, communication and performing actions steadily open opportunities and create niches for the knowledge-based systems (KBSs) in different fields, including popular e-titled domains: eHealth, eLearning, eCommerce, eGovernment, eSupply Chain, etc. As a rule, the KBSs are more suited for poorly formalized problems and as stated in preface [1]: they are especially valuable in situations

* Supported by RFBR 18-07-01164, 18-07-00604

in which the amount of available information is prohibitive for the intuition of an unaided human decision maker and in which precision and optimality are of importance. The particular development of applied KBS can be done in an appropriate general-purpose programming language or with the help of specialized tools (Protege, Drools, G2, etc.). The current paper contributes to the last option and considers the issues related to the creation of knowledge-based systems development platform [2].

The platform is being designed as a modern web-oriented software tool facilitating programmers and knowledge engineers in the course of applied KBS creating process. Explicit knowledge representation and related automated reasoning are the two key features of any KBS. Nowadays different forms of knowledge representation (conceptual models, ontologies, concept maps, etc.) and some types of reasoning techniques (logical-based reasoning; case-based reasoning; classifiers; rule-based reasoning and et.al.) are available for KBS creation. In addition, a lot of problem-oriented tools can be integrated inside applied KBS. This variety of available models and methods is a significant issue that should be in focus and defines one of the current work relevancies. As stated in [2] the proposed KBSs development platform should originally include methods and tools aimed at reconfiguring its architecture without significant efforts on rewriting the source code of the applied KBS.

The claimed platform functionality is provided by the unified interface-based approach: the component of the platform must implement the IComponent interface [2] that provides the ability to control the state and behavior of the component. So the platform is being designed as a component-based system with open architecture and, in theory, any desired problem-oriented task of KBS system can be "wrapped" in some software component. For example, to date some main components of the platform are developed and described: the data control component (S^{DB}), the data representation and editing component (S^{Dlg}), the subject domain model design component (S^{Ont}), rule-based reasoning component (S^{RB}). The current paper continued the presentation of another brick of the platform the workflow component, that can be considered as essential for integrated and assembling stages of the KBS development process. Together these components can facilitate the development of some applied rule-based KBS with modern web-oriented GUI.

The rest of the paper is organized as follows. First, we give an overview of the workflow approach regarding the issues of software development. Next section describes the data flow view of the proposed workflow component. The final part of the paper is devoted to an illustrative example concerning the application of the KBS platform and its workflow component for creating a decision support system in the infrastructure logistics domain (KBS4IL).

2 The workflow approach state-of-the-art

The concept of workflows can be discussed from different points of view: organizational, informational, manufacturing. But in the context of current research, we shall focus mainly upon the workflow approach regarding to the issues of software development.

2.1 The workflow management systems overview

Despite some fruitful attempts to describe and implement workflow processes methods and tools as a standalone artifact of the software development (1960s: Petri nets; 1970s: Xerox PARC "Office Automation Systems", 1980s Pi-calculus) only in the 1990s the workflow management systems (WfMS) were recognized as a standard component for information systems. One of the reasons for this is the history of software development evolution.

The emergence of the WfMS is the result of the stepwise evolution of application architecture. At first, the programs handled with their functionality (data storage, user interfaces, business logic, etc.) entirely by their own. Since 1970s to 1990s database management, user-interface management, workflow management systems were consistently extracted from program architecture and now exist as separate solutions. Some additional deterrent conditions for WfMS arising are indicated at [3]: workflow was never considered as a really new piece of functionality, the rigid and inflexible character of the early products scared away many potential users, and users had not been linked to a computer network till 1990s. The next phase of WfMS evolution relates to the popularity of the web- and cloud-based means and subsequent rapid growth of the applications with service-oriented architecture.

As the main application domain for the WfMS is the business management, a lot of WfMSs change their names to BPMS (business process management system). In particular, this led to confusion in terminologies. Here we give two opinions that may help in removing the uncertainty. On the one hand, from a methodological point of view business process management (BPM) is a process-oriented management discipline aided by IT whereas workflow management (WfM) is a technology, that can be found in business process management suites as well as in other product categories [4]. On the other hand, from a system architecture point of view WfM is a subset of BPM [5] according to stages of the BPM life cycle [6] (process design, system conguration, process enactment, diagnosis) with the diagnosis stage as the main difference. Currently the myriad of WfMS and BPMS are developed and actively used as industry-specific software systems that allow for better process control. Nowadays the workflow management component is a highly expected element for any modern software development tool.

The standard WfMS architecture according to the Workflow Management Coalition's (WMC) reference model include: workflow enactment service (containing workflow engines), process definition tools, workflow client applications (the employees access point), invoked applications (that WfMS can call), administration and monitoring tools, and a set of 5 corresponding interfaces between listed elements.

In these terms for the proposed workflow component workflow enactment service and process definition tools are designed from scratch whereas working with invoked applications is implemented on the top of existing platform functionality. Also note that we were inspired and guided by Workflow Patterns Initiative [7] and jBPM workflow engine [8] as reference tools for our work. On this basis lets consider some ideas concerning workflow approach.

2.2 The workflow approach overview

In general, a workflow approach can be thought as a way to organize "works" (units, tasks, activities) related to some business logic. This organization can be considered from different viewpoints or as given in [9] perspectives:

- The control view (CV) considers sequence, split, join, iteration, choice, parallelism, and synchronization of tasks.
- The data view (DV) concerns with data issues, like different types of variables passing between tasks, pre- and post-conditions of transitions.
- The resource view (RV) focuses on the types and properties of the resources needed to execute workflow tasks. Availability, qualification, performance, and capabilities of the software, devices, equipment, staff and etc. are described in the RV.
- The implementation view (IV) is about the ways on how workflow specification is bonded to real activities and applications via programming or graphical user interfaces, job description and so on. IV sometimes considers as ancillary because workflow management systems, as a rule, are generic software and do not oblige to actually perform any of the tasks in a handled process.

Depending on the levels of details, goals, expressiveness of used description languages the workflow model can include information from either one to four of given views. For more specific language free information about the variety of possible techniques of workflow model development, we can refer to Workflow Patterns Initiative [7], where a lot of patterns are designed for the mentioned views.

On the most abstract level, a workflow model consists of a number of tasks composed in the form of a directed graph. Several notations and languages have been proposed to define workflow models. Some of them concern problems of convenient primitives for a graphical description of workflows in diagramming way (BPMN, EPC, UML AD). The researches [10, 11] demonstrate that as BPMN (Business Process Model and Notation) and UML AD (Activity Diagram) could represent most of the workow patterns [12], but could not be translated into executable code due to the absence of adequately semantic and computational formalisms. Other languages are designed to fill this gap and focus on the deployment and automation aspects (BPML, BPEL, WS-CDL, XLANG). There are also languages in an intermediate position (BPDM, XPDL, YAWL).

For the purposes of theoretical analysis workflow models are often mapped to different types of Petri nets (Transition nets, Coloured Petri, Predicate nets) as the control flow view can be represented in terms of places, transitions, and arcs. For example, the Standard Workflow Models [13] based on Petri nets were introduced to have the tool for comparison of different workflow approaches. Note also, that the often-used owcharts do not have clear interpretation in the sense of mentioned views and cannot be considered as theoretically reasonable and practically applicable for WfMS as an real modeling language.

Finally, a proper workflow model is executed by a WfMS, and an executing instance of a workflow model is called a process instance (or a case). A specific workflow model can be executed via multiple instances that can run simultaneously but, as a rule, these instances are independent and have no references to each other. Currently there are two main approaches for workflow engine implementation: token-based and instance-based. The former originates from Petri nets and is familiar for workflow experts, whereas the last relates to modern OOP style and can be used for more convenient workflows integration into complex software environment.

To carefully take into consideration the peculiarity of KBS development the data flow view and implementation issues are in focus of further description of the proposed workflow component. The basic control flow patterns were utilized for testing the component. Also note, the resource view is not covered in the current paper and can be considered as future work.

3 The workflow component of the platform

The proposed workflow component (see Fig. 1) is created as a tool that can be used in different ways related to specific KBS. For example to organize compositions of functional platform components to assemble some method of applied KBS or to represent some imperative behavior of particular rule in rule-based KBS. So the workflow component is originally designed to be actively reused thought the applied KBS development process meanwhile providing specific KBS features support.

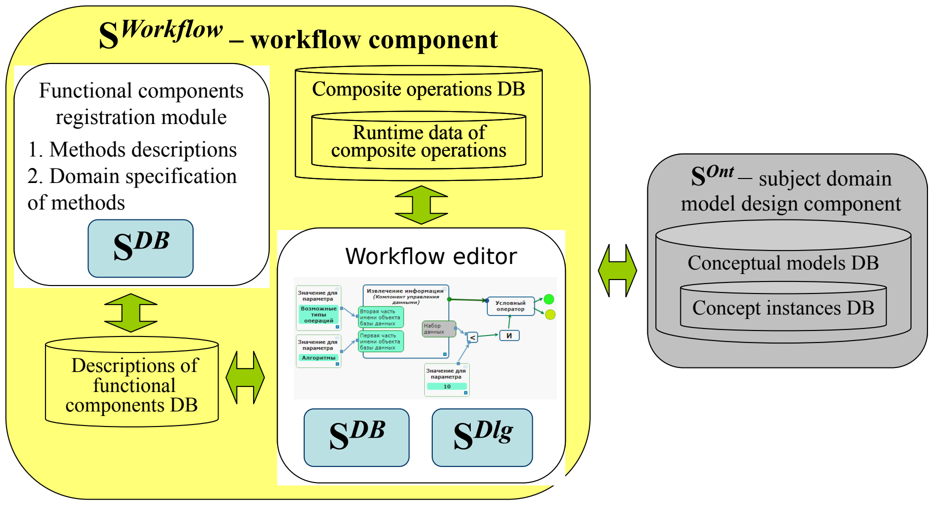
One of the main KBS features related to utilized approach [2] is the central role of the conceptual model in the development process. For example, the elements of the conceptual model are the basis for the fact templates of a rule-based expert system or for parts of agent-based simulation models (agent, event, environment, etc.). In the platform, the model is represented in the well-known concept-attribute-relation whereas based on the declared $Concept$ an $Instance$ can be defined. So any $Value$ of the data elements of the workflow used in the platform is in the following set:

$$Value \in \{Literal \cup Concept \cup Instance\},$$

$$Literal \in \{Text|Number\},$$

$$Concept =< Name, \{Attribute\} >,$$

$$Attribute =< Name, \{Value\} >,$$



S^Workflow – workflow component
Functional components registration module
1. Methods descriptions
2. Domain specification of methods
S^DB
Composite operations DB
Runtime data of composite operations
Workflow editor
S^DB
S^Dlg
Descriptions of functional components DB
S^Ont – subject domain model design component
Conceptual models DB
Concept instances DB

Fig. 1. A workflow component architecture

Hence any possible data element inside the workflow can be a $Constant$, which value defined in the design time; the parameter of some proper registered functional platform component method namely basic operation Op^B (an atomic task); the parameter of early designed composite operation Op^C (a block task). Also note, that taking into account KBS nature of workflow data elements we can explicitly specify their value range – $Value^{Range}$: any $Literal$, any $Text$, any $Number$, any $Concept$, any $Concept$ descendant of the specified concept, any $Instance$, any $Instance$ of the specified concept, any $Instance$ of the specified concept or its descendants. Introducing $Value^{Range}$ may help users to extend the workflow with additional semantic that always a valuable option for KBS. Summarizing the above assumptions now we can define $Workflow^{KBS}$ – the design time control and data flow view of the applied KBS method.

$$DataElement \in \{Constant \cup Parameter^{Op^B} \cup Parameter^{Op^C}\},$$

$$Constant \in \{Constant^{Literal} \cup Constant^{Concept} \cup Constant^{Instance}\},$$

$$Constant^{Literal} \in Literal,$$

$$Constant^{Concept} \in Concept,$$

$$Constant^{Instance} \in Instance,$$

$$Parameter^{Op^B} =< Name, Input^{Op^B} | Output^{Op^B} >,$$

$$Parameter^{Op^C} =< Name, Input^{Op^C} | Output^{Op^C} >,$$

$$Input^{Op^B} = Value^{Range},$$

$$Output^{Op^B} = < Value^{Range}, Value >,$$

$$Input^{Op^C} = < Value^{Range}, Value >,$$

$$Output^{Op^C} = Value^{Range},$$

$$Op^B = < Location, Name, \{Parameter^{Op^B}\} >,$$

$$Operator = < O_{DT} | O_{CT} | O_{If} | O_L | O_B >,$$

$$Workflow^{KBS} = \{DataElement \cup Op^B \cup Op^C \cup Operator\},$$

$$Op^C = < Name, \{Parameter^{Op^C}\}, Workflow^{KBS} > .$$

Here Op^B is a description of basic operation that can be directly executed by the platform; *Location* is an address of its implementation; Op^C is a description of composite operation interpreted in the workflow engine; O_{CT} is a control flow transition from one task to another; O_{DT} is a data flow transition from one task to another; O_{If} is an IF operator; O_L is a loop operator; O_B is a grouping operator.

Hence, based on the above the following algorithm for creating and using workflow can be formulated.

– Setup the workflow component for specific KBS K.
  1. Formation of a basic set of available actions (atomic tasks) by registration the methods of functional components along with input and output parameters $\{Op^B\}$.
  2. Defining the domain of K by referencing K to one or more conceptual models.
  3. Creation K domain-related $Value^{Range}$ for registered methods $\{Op^B\}$ of functional components if necessary.
– Creation specifications of the K functions. At this stage, a composite operation Op^C (block tasks) is designed for each element from a nested set of K functions using as building blocks the obtained on previous steps $\{Op^B\}$ set, as well as *Operator* set.
  1. Entering the general information of the composite operation (name, description and etc.), along with description their inputs and outputs.
  2. Forming the control flow of a composite operation by specifying the sequence of calls to the elements of the operation.
  3. Description of the set of required constants, as well as the sequence of passing values from the inputs and constants to the elements of the operation and then to the outputs of the composite operation.
  4. Definition the way of the composite operation invocation (composite operation, the element of the composite operation; a call from the action part of rule).

4 An illustrative example

The illustrative example describes the implementation of the proposed KBS development platform for creating the decision support system in the infrastructure logistics domain (KBS4IL). The originally designed methodology for creating KBS4IL is partly presented in [14]. The KBS4IL is aimed at supporting the research process of the regional infrastructure logistics and related multi-scale facilities on the base of a set of analytical methods and software [15, 16].

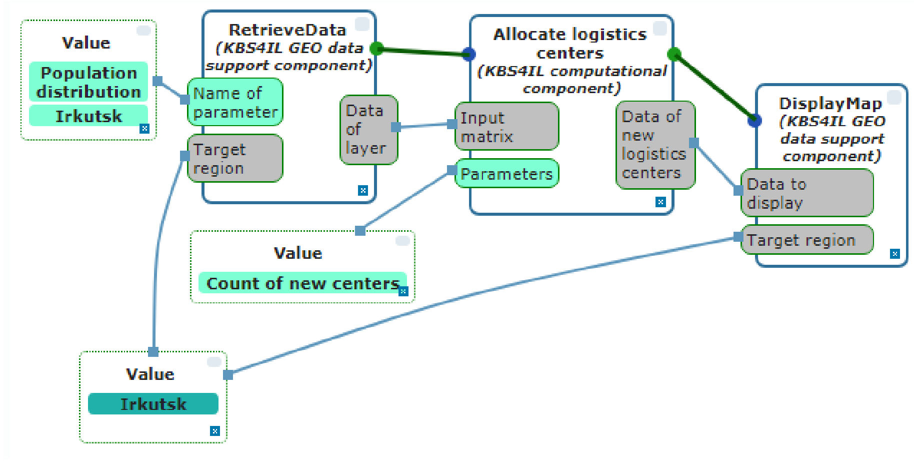
The infrastructure logistics considered on 3 levels: macro (a whole region industry level), meso (an allocation of facilities) and micro (a functioning of the particular type of a facility). As such research process is complex, different types of knowledge bases (KBs) for each level of consideration can be proposed: research session management, formation of the initial data set based on the analysis of the passport of the region in accordance with the specifics of the current level of consideration, identification of the infrastructure problems in the region, proposals of the measures related to identified problems, assessment of measures based on analytical methods and tools.

Currently, the main focus of KBS4IL is on the meso level of infrastructure logistics research process, and, in particular, on the development of research sessions and knowledge bases for scenarios of the heterogeneous infrastructure facilities distribution in accordance with transport and logistics system requirements. The scenarios may take into account a variety of factors such as financial constraints; insufficient number of required types of facilities and replacing them with analogs; the presence of given network topology.

The KBS4IL exploits a platform functionality for creating Conceptual Model of Infrastructure Logistics (CMIL) and instances of CMIL. The detailed description of the structure and underlying ontologism is presented in [17].Currently, CMIL contains information about infrastructure logistics objects, measures for improving the logistics situation, specifications of computational methods and tools. Eventually CMIL must include all the possible information to support researching process, therefore, it has special hierarchical structure for facilitating improvements and refinements.

During the process of the data and knowledge acquiring for CMIL, the students of Irkutsk National Research Technical University were engaged. They utilize the functionality of S^{Ont} component for developing specific client applications to insert information into the instance database. The GUI of client applications contains a different set of control elements depending on the features of the related subdomain.

The related KBS4IL computational component (S^{IL}) is based on the original algorithms for solving the cover and packaging problem of circles for special non-Euclidean metrics, to which the optimization problems can be formulated. According to the platform requirements the S^{IL} is a wrapper over exciting domain specific application (see Fig. 2). The input data for S^{IL} includes: the parameters of numerical methods, the parameters of infrastructure objects under consideration (values of possible radii of objects, number of objects, etc.), and



Value
Population distribution
Irkutsk
RetrieveData (KBS4IL GEO data support component)
Name of parameter
Target region
Data of layer
Allocate logistics centers (KBS4IL computational component)
Input matrix
Parameters
Data of new logistics centers
DisplayMap (KBS4IL GEO data support component)
Data to display
Target region
Value
Count of new centers
Value
Irkutsk

Fig. 2. A KBS4IL workflow example

the scalar value of point in the 2d or 3d (characteristic of the environment) in the form of a matrix.

The general formulation of S^{IL} parameters makes it possible to redefine them in a particular workflow using appropriate $Value^{Range}$ and depending on the domain-specific issues. For example, for the block task "the deployment of additional logistics centers" related custom S^{IL} would have the following domain parameters: matrix with population distribution; a set of exciting logistics centers of particular type (for example, any instance of "store" or "convenience store" Concept) with coordinates and radii; a number of new centers; a properties of new centers; a number of consumers of the center; delivery time from all centers to their customers. There are also special methods of S^{IL} to map and matrix data conversions.

5 Conclusions

The proposed workflow component can be reused on different levels of applied KBS development process for creation assembling scheme of functional components of the platform or representing some imperative behavior used in rules. The specific of KBSs is explicitly highlighted in the data flow view of the workflow component.

Thus, the capabilities of the platform along with the presented in the paper workflow component on the base of SIL component and CMIL help to create the meso level functionality of the KBS4IL that support the research process on the following stages: the initial conditions for scenario of distribution of infrastructure objects, determining the type and characteristics of infrastructure objects, determining the number of objects by type, facilitates the data conversion and required calculations.

The current research is partially supported by RFBR (18-07-01164, 18-07-00604).

References

1. Alor-Hernndez, G., Valencia-Garca, R. (eds.): Current Trends on Knowledge-Based Systems. Intelligent Systems Reference Library series. Springer, Cham Switzerland (2017)
2. Nikolaychuk O.A., Pavlov A.I., Stolbov A.B.: 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), pp. 1234–1239. IEEE (2018). https://doi.org/10.23919/MIPRO.2018.8400194
3. van der Aalst W.M.P., van Hee, K.: Workflow Management – Models, Methods and Systems. MIT Press, Cambridge MA London England (2002). https://doi.org/10.1016/S0933-3657(03)00011-3
4. Hill, J.B., Pezzini, M. and Natis, Y.V.: Findings: confusion remains regarding BPM terminologies. Gartner Research, Stamford CT (2008)
5. Georgakopoulos, D., Hornick, M. and Sheth, A.: An overview of workow management: from process modeling to workow automation infrastructure. Distributed and Parallel Databases **3**(2), 119–153 (1995)
6. van der Aalst, W.M.P., ter Hofstede, A.H.M., Weske, M.: Business process management: a survey. In: van der Aalst W.M.P., Weske M. (eds.) Proceedings of the International Conference on Business Process Management 2003, LNCS, vol. 2678, pp. 1–12. Springer, Heidelberg (2003).
7. Workflow Patterns Initiative Homepage, http://www.workflowpatterns.com. Last accessed 01 Sep 2019
8. jBPM Homepage, https://www.jbpm.org/. Last accessed 01 Sep 2019
9. Russell N., ter Hofstede A.H.M., Edmond D., van der Aalst, W.M.P.: Workflow Data Patterns: Identification, Representation and Tool Support. In: L. Delcambre et al., eds Proceedings of the 24th International Conference on Conceptual Modeling (ER 2005), LNCS, vol. 3716, pp. 353–368. Springer, Verlag Berlin (2005)
10. Russell, N., van der Aalst, W.M.P., ter Hofstede, A.H.M. and Wohed, P.: On the suitability of UML 2.0 activity diagrams for business process modeling. In: Stumptner, M., Hartmann, S., Kiyoki, Y. (eds.) Proceedings of the 3rd Asia-Pacific Conference on Conceptual Modelling, vol. 53, pp. 95–104. Australian Computer Society, Darlinghurst Australia (2006)
11. Koskela, M., Haajanen, J.: Business process modeling and execution: tools and technologies report for the SOAMeS projec. VTT Research Notes 2407, VTT Technical Research Centre of Finland, Espoo (2007)
12. Russell, N., ter Hofstede, A.H.M., van der Aalst, W.M.P., Mulyar, N.: Workflow Control-Flow Patterns: A Revised View. BPM Center Report BPM-06-22, BPMcenter.org (2006), http://www.workflowpatterns.com/documentation/documents/BPM-06-22.pdf. Last accessed 01 Sep 2019
13. Kiepuszewski, B., ter Hofstede, A.H.M., van der Aalst, W.M.P.: Fundamentals of Control Flow in Workflows. Acta Informatica **39**(3), 143–209 (2003)
14. Bychkov, I.V., Kazakov, A.L., Lempert A.A., Bukharov D.S., Stolbov A.B.: An intelligent management system for the development of a regional transport logistics infrastructure. Autom Remote Control **77**(3), 332–343 (2016)

15. Kazakov, A.L., Lempert, A.A.: An approach to optimization in transport logistics. Automation and Remote Control **72**(7), 1398–1404 (2011)
16. Kazakov, A.L., Lempert, A.A., Bukharov, D.S.: On segmenting logistical zones for servicing continuously developed consumers. Automation and Remote Control **74**(6), 968–977 (2013)
17. Lempert, A.A., Stolbov, A.B.: An approach to knowledge bases development for support of complex research in infrastructure logistics. Information and mathematical technologies in science and management **11**(3), 45–54 (2018).https://doi.org/10.25729/2413-0133-2018-3-05