Towards extensibility features of knowledge-based systems development platform

Aleksandr I. Pavlov, Aleksandr B. Stolbov and Anna A. Lempert

Matrosov Institute for System Dynamics and Control Theory, Siberian Branch of Russian Academy of Sciences (ISDCT SB RAS), Lermontov St. 134, Irkutsk, Russia

Abstract

The paper continues the series of publications devoted to the knowledge-based systems development platform (KBSDP). The focal point is a discussion about extensibility features of the platform: both presently available and potentially possible. The deployment issues related to contemporary technologies are also discussed. It is proposed to implement KBSDP components in accordance with the Platform as a Service paradigm. With the help of serverless (cloud) functions, it is also suggested to integrate external functionality into the KBSDP. An implementation model is proposed and related software is modified. In the context of illustrative examples (component-based modeling and integrated environmental modeling), the problems of technical, conceptual, and methodological integrations are analyzed. A set of ontologies for component-based modeling is adopted for utilization in KBSDP. The ideas for implementing conceptual and methodological integrations in the KBSDP workflow component are considered. Examples of sematic ports and controllers are presented, the principal algorithm of smart coupling is defined.

Keywords

Knowledge-Based Systems, Platform as a Service, Workflow, Component-based Modeling.

1. Introduction

Knowledge-based systems (KBSs) are one of the oldest applications of the artificial intelligence paradigm but are still in demand. They are known for their ability to solve poorly structured and formalized problems under conditions of uncertainty, where currently popular AI tools depending on big data availability can fail. KBSs are strong in processing expert knowledge and storing human experience that is hard to represent in a quantitative manner. The theory and practice of KBS development process can be considered from the AI methodology point of view or from a software engineering position. In the current paper, the last option is primary. So, let's make a brief look at common KBS architecture.

Despite the fact that KBSs are different either by formalisms for knowledge representation and reasoning and for domain focus, most of KBSs include at least two core elements: the knowledge base (KB) itself and the inference engine. Expert systems as the most notable KBS exemplar have an explanation subsystem (independently or as a part of inference engine). Another element that can explicitly exist in KBS is the computing subsystem complementing standard knowledge representation in symbolic, quality form. Text or graphical user interface subsystem must be also presented if KBS is intended for human interaction. Taking into account contemporary trends about big data processing and aggressive distribution of service-oriented architecture modern KBS is inevitably growing into huge and complex software. Moreover, as frequent changes apparently become constant in software engineering practice KBS architecture must be initially oriented on flexibility and extensibility.

1ITAMS 2021 – Information Technologies: Algorithms, Models, Systems, September 14, 2021, Irkutsk, Russia
EMAIL: asd@icc.ru (A. 1); stolboff@icc.ru (A. 2); lempert@icc.ru (A. 3)
ORCID: 0000−0002−7753−7514 (A. 1); 0000−0001−6513−7030 (A. 2); 0000-0001-9562-7903 (A. 3)


To address mentioned challenges the knowledge-based systems development platform (KBSDP) has been designed [1]. At the moment the KBSDP includes over 10 components for data control; ontology, rule, operation (workflow) editing, general-purpose and specialized web-oriented UI manipulation; one and two-way communication; as well as a set of applied components. Some notable KBSDP capabilities have been already proposed in the following publications: the architecture principles and program interface [1]; the workflow component [2] as the main assembling tool; user interface design and customization issues [3]. But some ideas have not been considered in a proper manner yet.

In the paper currently supported extensibility features of KBSDP along with promising ideas for the future are discussed. As illustrative example integrated environmental modeling (IEM) domain is chosen. In the context of extensibility, an important problem about platform elements deployment is also discussed.

2. A platform functionality extensibility

There are two main ways to extend the KBSDP functionality. First, the developer can create a new component based on the design principles and architecture of the platform. This way is considered in our early work [1]. In the current paper, we present the second way, when the developer integrates existing external functionality as a basic operation.

2.1. External functionality in workflow component

The functionality of the basic set of platform components, even organized into algorithms for solving applied problems based on combining their methods, is not sufficient to provide the applied needs of some arbitrarily taken subject or/and problem area. For example, Figure 1 shows a fragment with the algorithm for generating the knowledge base code. Herewith, only the basic components are used: data control component and rule-based reasoning component.

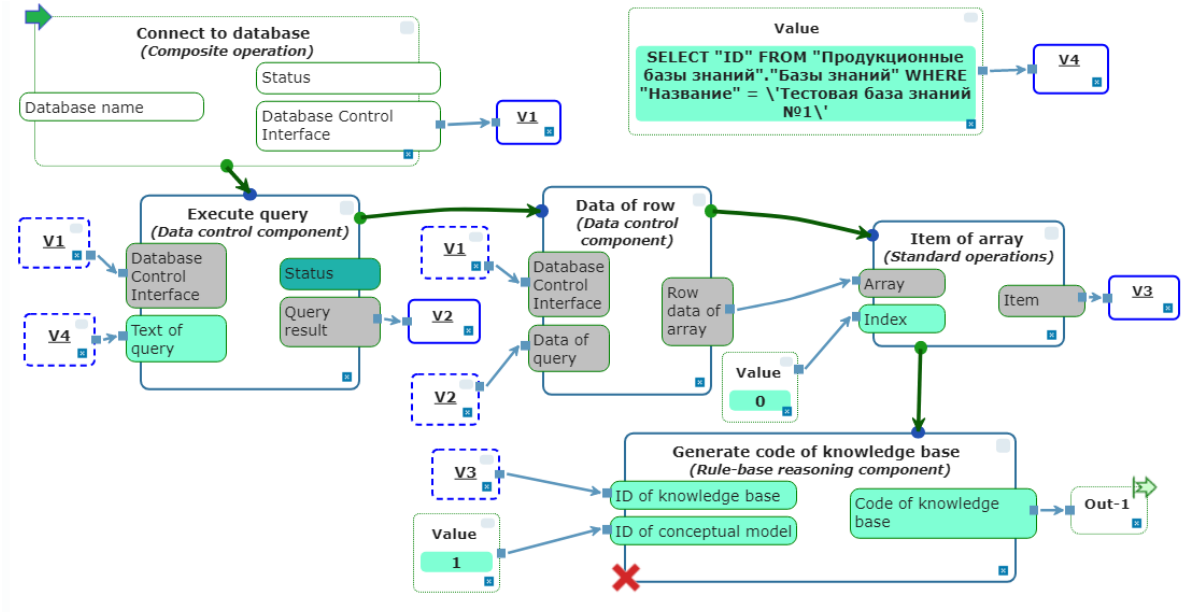

Figure 1: The algorithm for generating the knowledge base code (fragment)

At the same time, the specificity of problem domains, namely the possibility of using the same methods for a wide range of subject domains, determines the existing of a large number of ready-to-use software. So, the trend is about the organization of effective reuse of the existing methods and tools rather than developing new algorithms from scratch. In the context of the KBSDP it is proposed to provide code reuse based on the application of the visual design (programming) paradigm within

the workflow component. A visual scheme for solving an applied problem (algorithm) is based on the proposed model of a composite operation (Op^C) [2], the substantive elements of which are basic operations (Op^B), representing problem-oriented external functionality. So, Op^B is a description of basic operation that can be directly executed by the platform, Op^C is a description of composite operation interpreted in the workflow engine. A more detailed explanation about Op^B, Op^C, and other related issues of the operation model is discussed in [2]. To create Op^B from some external functionality it is proposed the Op^B implementation model (Op^BIM):

$$Component = < ID >, < Name >< Description >< Component\ function > \quad (1)$$

$$Component\ function = < ID >, < Name >, < Description >, \{< Component\ function\ parameter >\}, \quad (2)$$

$$Component\ function\ parameter = < ID >, < Name >, < Description >, < Type >, < Refrence\ to\ concept >, < Position\ in\ signature >, < Is\ multiple\ values >, < Is\ return\ value > \quad (3)$$

$$Type = Integer \mid RealNumber \mid Boolean \mid Text \mid Concept \mid Instance \mid Relation \quad (4)$$

The meaning of 1-4 formulas is clear from the names. So, let's explain only some elements: the last three in (3) are boolean flags showing that parameter can have lots of values, is (or not) returning as the output of the function and is (or not) the result of the function (since it is assumed that there can be only one result but some outputs for). In (4) a set of possible types is extended by knowledge-based ones.

2.2. KBSDP deployment as PaaS

In the course of the existing trends towards supporting of distributed work of research groups and teams, there is a task of organizing a technical infrastructure that can meet all the requirements for speed, scalability, fault tolerance, etc., required for modern web applications. Currently, the most effective way to solve this problem is to place the proposed the KBSDP platform as a PaaS solution in one of the existing cloud providers. To date, this area of information technology is represented by a large number of solutions, among which there are both foreign: Amazon Web Services, Microsoft Azure, Google Cloud; and domestic Yandex. Cloud and Mail.ru Cloud Solutions. In the context of current research, Yandex.Cloud [4] was chosen as a cloud provider, because its capabilities, firstly, correspond to modern global trends (many services are API compatible with Amazon Web Services), and, secondly, it complies with the legal norms of the Russian Federation in the field of personal data protection.

The KBSDP deployment problem can be divided into a number of stages, depending on the aspect under consideration: data storage; organization of data processing, including integration of external functionality; visualization of processing results; information security; maintaining operability and administration. Thus, to provide data storage for historical reasons, the PostgreSQL DBMS was used, both in the form of a cluster and in the form of a separate server. The organization of the data processing infrastructure was tested on the basis of a Kubernetes cluster, an instance group, a separate server, as well as in a serverless version in the form of cloud functions. To visualize the results of data processing, the Yandex DataLens service can be used, which includes all the main tools for business analytics. Data exchange based on the secure HTTPS and WSS protocols can be implemented on the basis of Yandex Certificate Manager. Service activities related to the processing of increased load, including DoS attacks, monitoring the state of the infrastructure, backup, and collection of system logs are carried out by standard Yandex.Cloud tools.

The experience gained with Yandex. Cloud confirms the reasonability and feasibility of cloud development and hosting, including for small teams of developers. Automation of routines and secondary tasks (visualization of processing results) provides an increase in the efficiency of the development process of the main functions of the KBSDP.

3. An illustrative example

In the context of considered issues among the many potential applications of the KBSDP it is desirable to select such domain that may benefit both from component architecture and explicit knowledge processing. So, the domain should provide a rich set of ready-to-use components; diverse subject and problem areas with a lot of terms, templates, and patterns; be presently relevant. With this in mind, an Integrated Environmental Modeling (IEM) domain [5, 6] and component-based modeling approach were chosen to test the capabilities of KBSDP.

3.1. Component-based modeling approach overview

IEM is a relatively new research area bringing together simulation and mathematical modeling, software engineering, and decision support to address actual challenges related to ecological and economic problems requiring joint efforts and collaborations of scientists, public organizations and authorities. Typical IEM tool includes simulation, data base, analytic and visualization modules.

A notable characteristic of IEM approach is focus on component-based modeling i.e., decentralizing of complex model functionality into independent parts. This technique provides the following benefits over traditional model development approaches: the research team can attract more specialists and resources to develop the model with the help of the community; the experts can spend more efforts on the implementation of elements related to their competence, whereas adjacent domains are still available via specialized coupling means; end-users can understand the structure of holistic systems built from standardized pieces. Nevertheless, there are a number of areas where the component-oriented approach, due to its initial generality, is inferior to models and systems obtained on the basis of the traditional centralized approach: computational efficiency (architecture efficiency, computational accuracy, hardware implementation features); data storage (unified standards for data processing and presentation, distributed storage, interconnection of storages and components); visual presentation, complexity of the study (documentation, manual, training, user interface); coupling tools. The KBSDP can contribute to addressing the last 3 mentioned problems, especially the last one.

Generally, the coupling problem can be divided into 3 classes: technical, conceptual and methodological integration. Despite the fact that the boundaries between these types are blurred, further we will consider the key features of each of them.

Technical integration makes it possible to carry out computations with a complex model, i.e. implement model chain execution. This type of integration can be done without core capabilities of the KBS. The model-component can be coupled by connecting outputs and inputs without formal computer-aided constraint checks or just standard type checking: string, number, boolean, dimension of collection, etc. All responsibility for problem and subject domain specific of coupling process is delegated to the users. Typical technical integration problems include: coordination of messaging based on unifying data exchange protocols; regular repeatability and reproducibility of model chain runs; optimization of model execution on different computing devices and architectures.

Conceptual integration forms a common language and provides understanding both between the models and within the design team. It is advisable to use the KBS capabilities. Typical conceptual integration problems include harmonization of concepts between disciplines used within and across models; formation of a models chain, providing meaningful links between models and related constraint check; creating simple convertors to deal with input/output inconsistency.

Methodological integration considers the methods used and the corresponding changes in the models. It is advisable to use knowledge bases. Typical methodological integration problems include spatial, temporal, or organizational scaling of models; creating sophisticated converters to harmonize issues both on the input/output and component-model levels.

Up to date most achievements have been made in technical integration. A lot of standards for components interactions are proposed (OpenMI, OMS, BMI. etc.) and applications in different domains are developed (SOBEK, Delft3D, MASCARET, ISIS, InfoWorks, SWAT, URUNME, etc.). These standards contain a set of functions for query execution and management, facilitating integration with other software elements. The standards can be implemented directly in the model code or through the facade software-design pattern.

3.2. An ontology for component-based modeling

Let's consider an example of technical integration in the KBSDP. As an illustrative example a well-known complex balance model for calculating dynamics of the hydrological and ecological characteristics of water objects [7] is used. From a mathematical point of view, this model is a system of linear ordinary differential equations, which allows us to represent each sum and as a model-component and the equation itself as the final model-component. All these model-components in terms of the KBSDP are examples of Op^B – basic operations added to the platform as external functionality. For testing purposes, a set of model-components to generate data is also proposed. The fragment of workflow related to considered model is shown on the Figure 2.

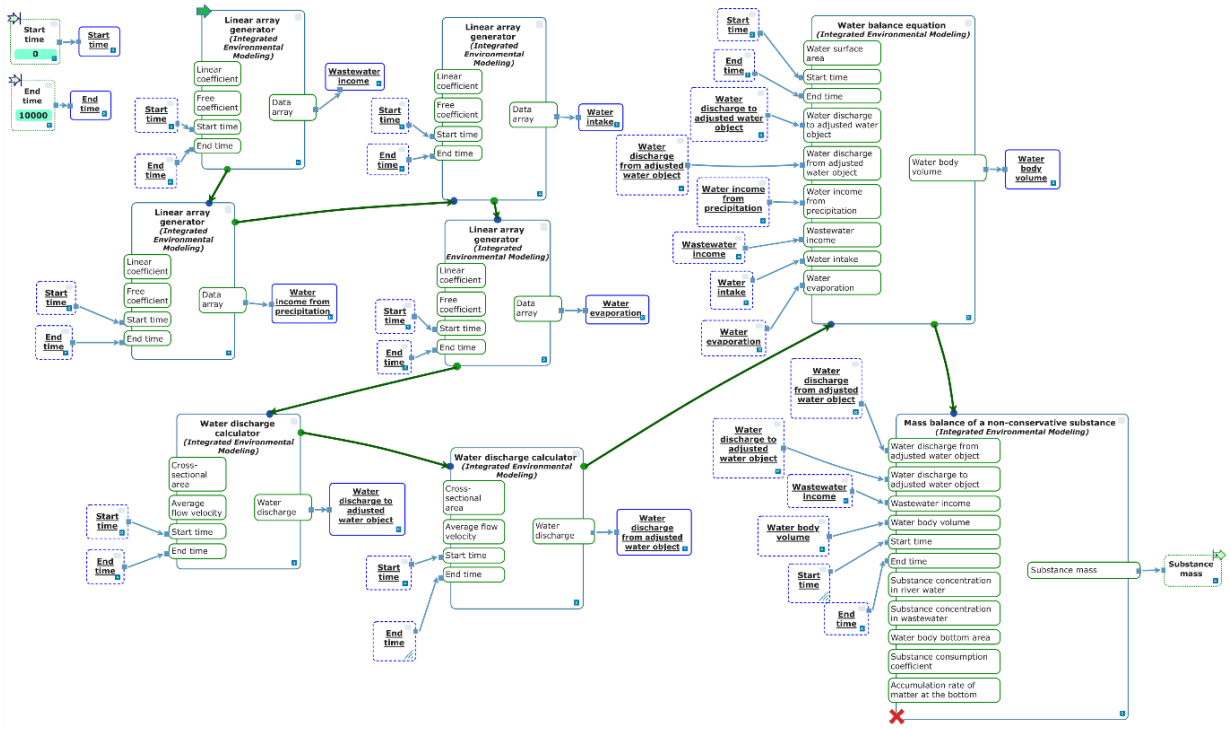

Figure 2: The workflow for complex balance model (fragment)

Note that in addition to standard data types the KBSDP allows one to explicitly use knowledge-based specific types (concept, relation, instance). But despite the meaningful titling and general constraints on types of parameters of basic operations, there is still a lack of domain-specific information to use for the component linking process.

One of the options to address this problem is the application of the ontological approach, which is traditionally used in cases where it is necessary to formally represent and reconcile various kinds of semantic and syntactic inconsistencies. Currently, the first results in creating ontologies of the component-based modeling process and its applications are obtained [8, 9]. The paper authors also are making contributions to the mentioned research area. Thus, the adoption of existing ontologies [8-15] into the KBSDP has been done (nowadays as an early prototype). Mainly based on the ideas from [8] the KBSDP operates the following levels of component-based modeling ontology (Ont^{Op}):

- The resource level contains information about the research domain and the project as a whole.
- The technical level contains information about the characteristics of the computing environment and software (including operating systems and programming languages) necessary for the component to work properly.
- The coupling level contains information about the characteristics of the environment within which the components are connected, each of which, in turn, has an associated programming interface and architecture.
- Math level contains information about research tools (for example, the structure of equations and methods for solving them)
- The domain level. Currently, IEM is being applied.

- The level of quantity types and units of measure.
- Scenario level contains information about methods for carrying out meaningful-interpreted multivariate calculations and analysis.

3.3. Towards semantic coupling

Having Ont^{Op} in mind it is possible to start solving the problems of conceptual and methodological integration. The first option is to allow user to make ontology queries for selecting proper models for suitable situation. Note that such a method begins to be implemented for conceptual integration in some existing IEM applications. The disadvantage of this option is the low level of automation as queries themselves are loosely formally connected and it is the user who has to organize query chains and understand which query is currently appropriate, even if somebody provides a set of query templates.

The second option is to develop special knowledge base (or groups of knowledge bases) containing the rules for checking possibilities of operation linking and for giving advice about selecting proper elements. This option is well corresponded with the following KBSDP components: ontology and rule editors, rule-based reasoning, data control. The related topic about using knowledge bases and the KBSDP for creating scenario analysis support tools have been partially discussed in [16]. So under this article, this option of integration is not considered in detail.

The third option for conceptual and methodological integration can be done in workflow component. As stated in [17] and further reformulated in the context of the KBSDP [2], the workflow can be considered as a set of perspectives or views. The two most valuable are the control view (CV) and the data view (DV). The set of elements of these views are depending on the specific implementation. For example, the KBSDP workflow component supports CV and DV in the form of the model of operations [2]. In addition to CV and DV, there are other views such as resource or implementation views. So continuing this idea it is proposed the new semantic view (SV) of the workflow to support conceptual and methodological integration.

The SV has to contain additional information (for example in the form of restrictions) necessary for the correct assembly of operations that take into account the features that are not explicitly expressed in DV and CV. The formally semantic view can be defined as follows:

$$SV = \{< Port, Controller >\}, \quad (5)$$

where *Port* – any connection point that can be attached to a visual node representing an operation; *Controller* – any procedure for evaluating the port connectivity.

So, during the design time of the complex model development, when a connection is attempted, the communication between the ports is validated. The elements of SV can be created manually based on Ont^{Op} or in semi-automatic mode based on the predefined rules. The last option is a promising stand-alone problem that needs to be addressed separately. Nowadays a set of possible ports in the context of IEM and component-based modeling is proposed:

- type of input and output parameters of Op^B and Op^C;
- constraints for input and output parameters of Op^B and Op^C;
- type of simulation: synchronous or asynchronous;
- model or equation type;
- data quality characteristics: complete, incomplete, minimum size/length;
- temporal and spatial scale;
- type of computational algorithm.

The examples of SV controllers mainly related to input and output parameters constraints:

- type constraints;
- unit constraints;
- dimension constraints (scalar, vector, matrix, tensor);
- quantitative type constraints (none, unitless, acceleration, angle, area, concentration, etc.);
- format constraints (csv, excel, etc).

In some cases, related to type, unit, or format inconsistency a set of convertors can be proposed. And summarizing the proposed ideas about semantic view let's consider the idea of an algorithm for smart coupling:

1. The ports of SV for Op^B or Op^C are created.
2. A set of controllers are associated with each port.
3. During the design process, when attempting to connect, the link between ports is validated.
4. If there is inconsistency in the units of measurement, types, or formats the proper converters can be applied.
5. If a port is associated with a large number of controllers, then the scope for the port values is reduced, and it is possible to show available connection routes or a list of candidate elements to select.

4. Conclusions

The paper deals with the knowledge-based system development platform (KBSDP) in the aspect of its extensibility. This issue is one the most significant in the current context of software engineering as regular changeability is immanent property of any modern complex software. The problem of extensibility is discussed from different points. First, we referred to previous work where the architecture principles providing extensibility for basic platform components are proposed. Second, we consider the problem of external functionality integration via basic operation. As a result, currently the KBSDP workflow component can organize together internal and external functionality that, in particular, allows one to implement the intelligent user support via adding into the solving algorithm of rule-based reasoning blocks to control the process of selecting the next operation and assessing intermediate results. Moreover, for each problem, several variants of the solving algorithm can be created.

Another aspect of extensibility lies in the area of ease of access and deployment of the KBSDP. The modern trend for attracting new users and exploitation external resources through service orientation stipules the idea of the KBSDP deployment towards Platform as a Service software. The issues about data storage and organization of data processing in the Yandex.Cloud ecosystem are considered and related changes in the KBSDP are made.

Finally, starting from the illustrative examples concerning component-based modeling and integrated environmental modeling the problems of technical, conceptual, and methodological integrations along with related ontologies are discussed. The promising idea of smart coupling during the design time in the workflow component is considered.

5. Acknowledgements

The present study was supported by the Ministry of Education and Science of the Russian Federation Project no. 121030500071-2 "Methods and technologies of a cloud-based service-oriented platform for collecting, storing and processing large volumes of multi-format interdisciplinary data and knowledge based upon the use of artificial intelligence, model-driven approach and machine learning", no. 121041300065-9 "Theoretical foundations, methods and high-performance algorithms for continuous and discrete optimization to support interdisciplinary research" and the Russian Foundation for Basic Research and the Government of the Irkutsk Region as part of a scientific project No. 20-47-380001. Results are achieved using the Centre of collective usage «Integrated information network of Irkutsk scientific educational complex».

6. References

[1] O. A. Nikolaychuk, A. I. Pavlov, 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, IEEE, 2018, pp. 1234–1239. doi:10.23919/MIPRO.2018.8400194.

[2] A. I. Pavlov, A. B. Stolbov, A. S. Dorofeev, The workflow component of the knowledge-based systems development platform, in: Proceedings for 2nd Scientific-practical Workshop Information Technologies: Algorithms, Models, Systems, ITAMS, volume 2463 of CEUR Workshop Proceedings, 2019, pp. 47–58.
[3] A. I. Pavlov, A. B. Stolbov, Domain-oriented specialization tools for knowledge-based systems development platform, in: Proceedings of the 3rd Scientific-practical Workshop Information Technologies: Algorithms, Models, Systems, ITAMS, volume 2677 of CEUR Workshop Proceedings, 2020, pp. 59–69.
[4] The secure, fault-tolerant cloud platform from Yandex, 2021 URL: https://cloud.yandex.com/en/.
[5] R. Argent, An overview of model integration for environmental applications - Components, frameworks and semantics, Environmental Modelling & Software 19 (2004) 219–234. doi:10.1016/S1364-8152(03)00150-6.
[6] R. Moore, A. Hughes, Integrated environmental modelling: Achieving the vision, Geological Society London Special Publications 408 (2016) 17–34. doi:10.1144/SP408.12.
[7] V. N. Mihajlov, Hydrology of river mouths, Izd-vo MGU, Moscow,1998 (in Russian).
[8] M. Elag, J. L. Goodall, An ontology for component-based models of water resource systems, Water Resour. Res. 49 (2013). doi:5077–5091.10.1002/wrcr.20401.
[9] R. Dunlap, L. Mark, S. Rugaber, V. Balaji, J. Chastang, L. Cinquini, C. DeLuca, D. Middleton, S. Murphy, Earth system curator: Metadata infrastructure for climate modeling, Earth Science Informatics 1 (2008) 131–149. doi:10.1007/s12145-008-0016-1.
[10] J. Horsburgh, D. Tarboton, D. Maidment, I. Zaslavsky, A relational model for environmental and water resources data, Water Resources Research 44 (2008) 1–12. doi:10.1029/2007WR006392.
[11] O. A. Nevzorova, N. Zhiltsov, A. Kirillovich, E. Lipachev, OntoMathPRO Ontology: A Linked Data Hub for Mathematics, Communications in Computer and Information Science 468 (2014) 105–119. doi:10.1007/978-3-319-11716-4_9.
[12] R. Raskin, M. Pan, Knowledge representation in the Semantic Web for Earth and Environmental Terminology (SWEET), Computers & Geosciences 31 (2005) 1119–1125. doi:10.1016/j.cageo.2004.12.004.
[13] J. Syvitski, E. Hutton, S. Peckham, R. Slingerland, CSDMS - A modeling system to aid sedimentary research, Sedimentary Geology 9 (2011) 4–9. doi:10.2110/sedred.2011.1.4.
[14] The Mathematical Modelling Ontology, 2021 URL: https://sourceforge.net/p/mamo-ontology/wiki/Home.
[15] Units of Measurement Ontology, 2021 URL: https://github.com/bio-ontology-research-group/unit-ontology.
[16] A. I. Pavlov, A. B. Stolbov, The Application of the Knowledge-Based Systems Development Platform for Creating Scenario Analysis Support Tools, Advances in Intelligent Systems Research 169 (2019) 43–48. doi:10.2991/itids-19.2019.36.
[17] N. Russell, A. H. M. ter Hofstede, D. Edmond, W. M. P.van der Aalst, Workflow Data Patterns: Identification, Representation and Tool Support, volume 3716 of Lecture Notes in Computer Science, Springer-Verlag, Berlin, 2005, pp. 368–394. doi:10.1007/11568322_23.