Ontological CFD-repository

V.A. Zenkin <vl.zenkin@gmail.com>
Bauman Moscow State Technical University
5/1, 2-nd Baumanskaya st, Moscow, 105005, Russia

Abstract. Based on the RDF-storage, a software tool was developed for creating a knowledge base containing information about the CFD-calculations performed. The software is a set of scripts written in bash and python, which are published under the GNU GPL3 license. The tool is designed to support the user when making research studies that do not have a strict, pre-defined design of experiment or problem solving algorithm. To formalize the description of the calculation stored in the knowledge base, an ontology is created that serves as the information model for the calculation. As an auxiliary mechanism for carrying out an automated comparison of calculations with each other (a mechanism of "comparators" and "features") was developed and also described in the article. In addition to the systematized data storage, the complex provides the possibility of their automated and semi-automated analysis, including the presentation of a set of calculations in the form of an undirected graph, the construction of flat and spatial dependencies, the search for similar calculations, etc. The article gives examples of data processing results for the project on design of channel in the cylinder head of a piston engine.

Keywords: CAE methodology; ontology; knowledge management; computational gas dynamics; semantic technologies

DOI: 10.15514/ISPRAS-2018-30(5)-15

For citation: Zenkin V.A. Ontological CFD-repository. Trudy ISP RAN/Proc. ISP RAS, vol. 30, issue 5, 2018. pp. 249-264. DOI: 10.15514/ISPRAS-2016-30(5)-15

1. Introduction

At the initial stage of numerical research during Computer-Aided Engineering (CAE), the typical situation is intuitive search in a multidimensional factor parameter space. The purpose of this search is to obtain primary knowledge about a system. At this stage, it is almost impossible to produce a rigorous plan of experiment, thus, it is often unsystematic and extremely difficult to process and analyze its data. Usually, all results are presented in a tabular form (an example of the table of CFD calculations from the experience of the author of this article is shown in fig. 1), which is often very poorly systematized (because the system is just the result of this work). The analysis of such tables becomes very difficult when the number of calculations exceeds several dozen. It results in an increase in the complexity of this research stage, frequent excess and repeated calculations, which can be avoided with proper systematization of obtained results.

	A	B	C
1	---------- Profiling the exhaust channel ----------	G, kg/s	
2	base — base channel for analysis, in 2 sections, with a lift of 7 mm. For him, the analysis of grid converge	0,155	0,0%
3	base_without_bob — it's without a lug	0,155	0,0%
4	base_without_styk - it's without a groove at the junction of the channel and pipe	-	
5	base_without_ustup - it's without a step between the saddle and the channel	0,156	0,6%
6	----------		
7	opt7 - the geometry formed for optimization on lift of 7 mm. Compared with the base cleaned ledge, lug, jo	0,155	0,0%
8	opt7_b - it's with an increased output section to R19	0,156	0,6%
9	opt7_b_wps - it's with an increased output section to R19 but without an intermediate section	0,157	1,3%
10	opt7_g - it's with a narrowed intermediate section	0,147	-5,2%
11	opt7_gb - it's with a narrowed intermediate and increased output	0,148	-4,5%
12	opt7_rb - it's also with extended intermediate and extended output	0,157	1,3%
13	opt7_eb - it's with an ellipse of 33 to 38 in the intermediate section (a little more than the standard) and an	0,156	0,6%
14	opt7_e36 - ellipse as above. Output section R18 (between standard and b)	0,156	0,6%
15	opt7_b_wps_E - it's with an increased output section to R19 in the shape of an ellipse and without an inter	0,1575	1,6%
16	---------- Checking other pressure drop		
17	------- At a drop of 1,1 to 1		
18	opt7	0,0287	0,0%
19	opt7_rb	0,0297	3,5%
20	------- At a drop of 5 to 1		
21	opt4	0,132	0,0%
22	opt4_rb	0,1325	0,4%

Fig. 1. Example of the table with the results of the initial search study of the engine exhaust channels (the initial stage of channel profiling)

The problem of scientific and methodological support for an engineer and researcher in the field of knowledge management (accumulation, structuring, reuse, automatic and semi-automatic analysis) has recently been the subject of a large number of works. This growth rate in publications and papers is associated with the development of semantic technologies (methods and tools that provide and use information coding, in which the value is stored separately from data) and with the emergence of protocol standards and file standards, and the development of supporting software.

If one considers publications only about Computer-Aided Engineering (CAE), the most important works are on user support when carrying out numerical modeling (due to the relatively high complexity of this procedure); in addition, these technologies can be used for configuration problem solving [1,2].

In the work [3], the authors solve the problem of experiment planning. For this purpose, they created an ontology – an explicit formal description of terms in a domain and the relationships among them – and a data warehouse of the performed optimization procedures together with the results of their work. An engineer uses this set as a knowledge base when choosing parameters to plan a numerical experiment. In the work [4], an ontology and semantic knowledge base are used for the logical processing of constraints imposed on structural components of construction, which should reduce the number of errors in solving a design problem. In the work [5], the use of standardization technologies for strength calculations promises a 75% reduction in the complexity of similar tasks. The author of [6, 7] works on the application of these technologies in the aviation and space industry. In addition, there

are many examples of the use of ontologies and other semantic technologies in other areas included in the product lifecycle.

Currently, there are many software tools designed to systematize and accumulate knowledge. They can be both general (PDM systems) and narrowly focused on the creation of CAE knowledge bases [8, 9], but these products have a predominantly corporate purpose and are intended, first of all, to reuse the results of calculations (including by other calculation specialists) and to organize work in a large team. This is expressed in the fact that final calculations are saved in such systems, and intermediate (erroneous or incomplete) calculations are not recorded; however, sometimes they contain no less important information for a specialist. The author of this article was unable to find a single software product designed to support the systematization of calculations for the individual work of a calculation specialist that solves a task without any known algorithms in advance. (An example of a similar product could be the popular git version control system, which provides access to a program code at all stages of its development at once).

Thus, the purpose of this article is to create a tool that solves the following problems:

- collecting and storing information about calculations;
- presenting this information in a structured (corresponding to a relational model) or semi-structured (containing labels for separating semantic elements and for ensuring a hierarchical structure of records) way, i.e. not as raw data, but in the form of ordered knowledge;
- processing and presenting the collected knowledge to a user in a way that simplifies analysis, or that helps to perform this analysis automatically.

The field of application of the developed tool is performing calculations in Computational Fluid Dynamics (CFD); however, the proposed method can be easily adapted to other areas of CAE.

2. Storage of Repository of Calculations in a Knowledge Base

This article proposes to use a semantic knowledge base in the RDF format [10], which can be accessed via a SPARQL. This base is for storing information about the performed calculations. The RDF format stores data in a triple format, i.e. statements of the following type “subject-predicate-object” (for example, “calculation1 obtained_with_program simpleFoam”. In addition, each entity appearing in the knowledge base may be mentioned an unlimited number of times and in any of these positions. As a result, the entire knowledge base is a directed graph of arbitrary structure. Existing SPARQL implementations make a graph search more efficient using a query language similar to SQL for usual databases.

To consider a semantic database as a knowledge base, in addition to direct information about specific calculations (ABox), it must also contain an information model defining the structure of this information and its semantics (TBox).

An information model is a formal model of a limited set of facts, concepts or instructions designed to meet a specific requirement [11]. In other words, such model sets a formal data structure by defining the relationship among data and, thereby, by

transforming this data into knowledge. As a rule, the construction of this model is based on mathematical logic. To process and present data (especially numerous and complexly structured), the formation of an information model is as necessary as the formation of a mathematical model for performing calculations. Moreover, as a mathematical model, an information model is verified by its practical application and compliance with the requirements imposed on it. An unsuccessful information model is hard for the perception of information by a man and makes it difficult or even impossible for machine processing.

Currently, the most common way to represent information models is ontologies. They help to select classes of objects and define their interdependence, i.e. at the same time they determine the syntax of a knowledge base by fixing the key names of concepts and relationships and making their logical connections among themselves, thus, they make the work of an inference machine possible.

In this article, the information model of CFD calculations in the form of ontology was proposed. The description of this ontology is presented in the next section.

3. Ontology of CFD Calculations

The structure diagram of the key elements of the developed ontology is shown in fig. 2. The ontology file is a part of the repository and is loaded into its database when a server is started, but it can also be used by itself. Currently, all names of classes and relationships are recorded in Russian for the convenience of a Russian-speaking user, but the author is working on an English version with the translation of all relationships between languages.

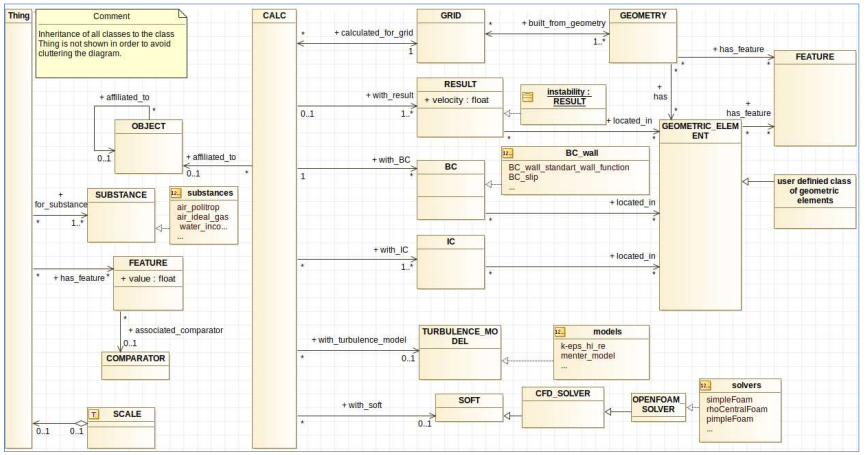

Fig. 2. Ontology of CFD calculations

The central part of this ontology is the mutual relationship between the “Calculation”, “Grid” and “Geometry” classes, which correspond to the standard methodology of CFD calculations. Geometry can include the hierarchy of components – “Geometric Elements”. Each object can contain a number of parameters as defined by the

ontology (turbulence model, solver, etc.) or by a user. In this case, no adjustments of the ontology for the repository are not required. For example, to set the “diameter” parameter, use the following line:

```
:geometry_a :diameter "0.1" .
```

where `geometry_a` – a corresponding instance of the `geometry` class. Despite the undoubted importance of the flexibility of data entry, which provides the ability to create the arbitrary hierarchy of components of calculation elements, in addition to this flexibility, the efficiency of recording is of great practical importance, because it directly affects the time, which a user spends when working with the repository. From this point of view, the proposed ontology may seem to be over complicated and the alternative option should be considered, where instead of the “hierarchical” data specification, meaningful prefixes are used. Table 1 shows an example of one object set in these two ways.

Table 1. Comparison of hierarchical and prefix setting of object parameters

Representation	Example of setting a group of objects in a turtle format
Hierarchical representation	:channel_a :contains: input_section_of_channel_a. :input_section_of_channel_a a :Input_section ; :shape :circle ; :area “0.1” .
Hierarchical representation with an anonymous node	:channel_a: contains [a :Input_section; :shape :circle ; :area “0.1”] .
Prefix representation	channel_a :input_section_shape :circle ; :input_section_area “0.1” .

Where :Input_section – user’s geometric class.

The first two representations are formally more stringent and better fit into the concept of a semantic database; in addition, they help to perform accurate addressing to the geometry of other important calculation parameters, for example, boundary conditions. Therefore, the developed ontology is based on the first representation. However, one should point out that they require a longer recording, more difficult in the formulation and perception, thus, in some tasks they may be redundant for a user. Therefore, the structure of concepts was constructed in the way that a user can use the last prefix representation if there is a need. The use of both representations within the same repository is undesirable, but possible.

The other ontology elements generally duplicate the standard structure of a CFD calculation and do not require special comments with the exception of a comparator mechanism, which will be discussed in the following sections. The idea of a function object was presented in order to enable the classification of calculations according to their application area from the point of view of a specific research object or an ongoing project if a user is going to store several projects in one repository.

4. Ontology Repository Architecture

To demonstrate the capabilities of semantic storage of data on numerical simulation results, the author has created a software package consisting of a set of scripts providing input and processing of data stored in the repository.

In the use-case diagram in fig. 3, typical tasks of a calculation engineer are shown. Elements of the work of an engineer are green (I), which, when using the proposed methodology, are becoming simpler due to a structured knowledge base. Red color (II) – additional tasks that occur when working with the system. Yellow (III) – tasks that are modified when working with the repository. The system, despite the fact that it requires some effort for its customization and development, provides substantial support in the most complex issues of analysis of the calculation bank.

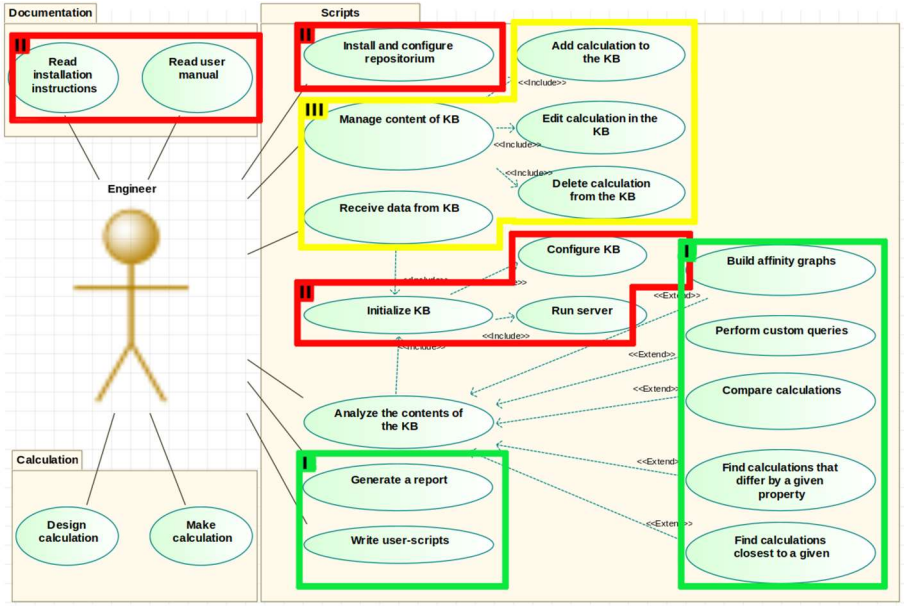

Fig. 3. Use-case diagram for CFD repository

The maintenance of the database of the performed calculations accompanies the work in any case, but the proposed repository provides an alternative tool for this. Unlike many trivial ones (text and table files), it provides the capabilities of automatic and semi-automatic data analysis. In this case, the procedure for tracking data is complicated. For this reason, the use of the repository is expedient in case of a relatively small number of calculations (dozens) combines with their hard-formalized relationships with each other. For example, this includes identification problems, various search studies without a known algorithm, etc. When solving typical problems or performing a large number of automated calculations that implement factor experiments, the expediency of using the repository is questionable.

Currently, the repository has several tools that solve typical problems:

- comparison of two calculations with each other and selection of similar and different properties;
- construction of two-dimensional and three-dimensional graphs;
- search for two most similar calculations that are different in a given parameter;
- search for calculations that are as close as possible to each other in their parameters;
- construction of an undirected graph containing a complete bank of calculations (or its fragment), whose branches connect calculations that are similar to each other.

The semantic representation of information helps to expand this list indefinitely based on the tasks of a particular user by adding scripts that access the knowledge base. The use of the SPARQL query language, which allows receiving answers to very complex questions from the knowledge base, opens up the widest capabilities for a user. The amount of time required to understand this language (at a basic level) does not exceed several days.

In future, the availability of the knowledge base can be used to solve more complex problems, for example, providing functions to expert systems and decision support systems using an inference machine to the existing ontology and information. However, the mechanisms necessary for this still require their development.

5. Features and Comparators

As can be seen in fig. 3 and the described list of implemented scripts, the key issue in repository work is the problem of comparing two calculations with each other.

To compare, the ontology applies the concept of features and comparators. A feature is a special entity automatically generated in a knowledge base. Each object has a set of such features, and it is assumed that the more differences these lists contain, the greater the number of parameters that objects differ in from each other, and, consequently, the more they are separated from each other. Features are inherited by Geometry -> Grid -> Calculation; in other words, any feature of geometry is a feature of all calculations performed according to this geometry.

A user does not set these features directly, because they are not included in the repository, but they are the result of its processing. Instead, a user assigns corresponding comparators to the parameters of his interest. This approach provides the ability to enter into the knowledge base all available data, because it will be possible not to specify comparators for parameters that are not relevant for a given task, and they will be excluded from automatic analysis procedures.

A comparator is a special label object, which means that an inference machine must automatically generate a feature for all related triplets by a given rule. Currently, five comparators with different feature generation algorithms are implemented. In this case, complex comparators take into account the scale value that a user can set for objects when generating features. The scale is inherited by Functional object ->

Calculation -> Grid -> Geometry, which simplifies to set it for a group of similar calculations.

The simple comparator written in the form of a predicate logic formula is given below. Fig. 4 shows an example explaining the second part of the formula that is designed to work with an unnamed first-level node.

$$\forall s,p,o\left(\begin{array}{c} associated\_comparator(p, simple\_comp) \wedge p(s,o) \wedge uri(s) \\ \Rightarrow has\_feature(s, feature\_generated\_comp(p, o)) \end{array}\right)$$

$$\vee$$

$$\forall par,s,\ p,o,type\left(\begin{array}{c} associated\_comparator(p, simple\_comp) \\ \wedge p(s,o) \wedge type(s) \wedge has(par,s) \wedge blankNode(s) \\ \Rightarrow has\_feature(par, feature\_generated\_comp(p, o, type)) \end{array}\right)$$

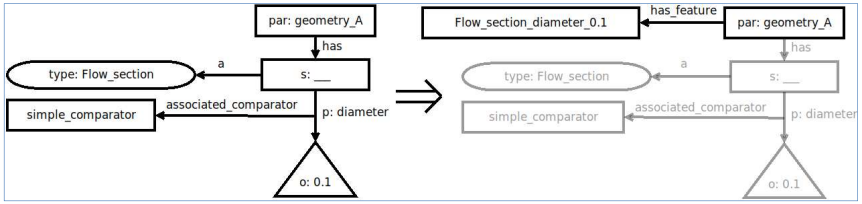

Fig. 4. A diagram illustrating the principle of operation of a comparator with a blank node

The use of the scale mechanism and comparator mechanism potentially make it possible to carry out the comparative analysis of calculations relating to different objects by reducing comparison criteria to dimensionless complexes.

6. Technical Implementation of the Repository

In the current version, the repository is a set of scripts written in bash and python, which interact (using a number of system and third-party programs) with a user and data contained in the repository. Data is stored as text files in the turtle format. To process them, these files are uploaded to the local Apache Fuseki server [12], which provides access to them via a SPARQL. The repository is managed via the command line. To visualize data, a browser (to display html files), the graphviz graph builder with its xdot, gnuplot shell and direct data output to the console are used.

The ontology described in the previous sections is also loaded into Fuseki; however, universal inference machines are not used, because they are not currently required to provide the repository functionality. Instead, a simplified inference machine based on user rules is used. The comparator mechanism is implemented as separate python scripts.

Simplified work of a user with the repository can be represented as follows.

First, it is necessary to initiate the repository, i.e. start the Fuseki server and enter data into it. The line:

```
> rep start
```

The next step is work with the repository.

To add new data:

```
> rep add sample.ttl
> rep add
```

To display repository content:

```
> rep diplay all calculations
> rep display calculation_original
```

To compare and analyze various data:

```
> rep diagram
> rep compare calculation_original calculation_a
```

If new data has been entered before the analysis, it is necessary to re-initiate the system of comparators:

```
> rep comparators
```

After work is finished, a server can be stopped by the command

```
> rep stop
```

To ensure reliability in terms of the safety of user data at the current stage of development of a software product, no data changes that occur in Fuseki affect the actual file storage of data. Thus, possible script errors or erroneous user requests to the server do not spoil user data.

7. Usage Example

In the last section of this article, the author provides a practical example of using the repository to solve the applied problem of profiling channels in the cylinder head of a piston engine.

On the one hand, this task is characterized by a large number of independent parameters that describe the complex spatial shape of a cast channel, on the other hand, by very small boundaries of parameter changes according to the design considerations, customer requirements, and common sense. (Sometimes there is no reason to change the existing design – when the design documentation has been already prepared, and the production process has already been adjusted – if the potential positive effect is not too great). In such conditions, the formal plans of experiment are of little use, and the profiling process is intuitive and unsystematic at least at the initial stage.

In the described case, the steady state flow of exhaust gases through the exhaust port of a head with two valves of a piston engine with a fixed valve position is shown in fig. 5. The supersonic pressure differential was used as boundary conditions, because most exhaust gases leave a cylinder in such flow conditions for this engine. The purpose of the work was to increase the flow through the channel. The rigid dimensional requirements and inadmissibility of redesigning the gas air duct determined the boundary narrowness of the permissible change in geometry, which can be provided in various ways. The shape and area of the outlet section from the head, the shape and area of the intermediate section in the middle of the channel, as well as a number of structural elements, were used as main parameters to variate.

Also, several additional valve lifts were considered and grid convergence was evaluated. To simulate the flow, the NSF-3 software package was used, which is based on the modification of the method of large particles.

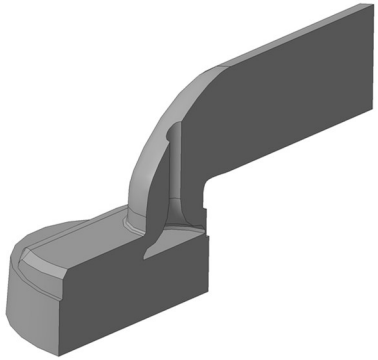

Fig. 5. 3D-model of the computational domain (cutting)

Fig. 1 shows the table of the calculations that records this initial stage of profiling. It is not difficult to notice that, despite its small size, the analysis of this table is very difficult. To test the work of the repository, the data presented in it were converted to the turtle format and uploaded to the repository. An example of the calculation description is as follows:

```
:c1_opt7_b
:mass_flow "0.156";
:obtained_with_program :NSF_2017;
:ted_to :single_exhaust_port;
:for environment: air_politropic;
:with_boundary_conditions
                [:located_in :Output_section;
                 :pressure 100000];
:with_boundary_conditions
                      [:located_in :Cylinder_cut;
                     :total_pressure 300000;
                     :total_temperature 1300];
:calculated_for_grid: gr_opt7_b
.
:gr_opt7_b
:built_from_geometry:g_opt7_b
.
:g_opt7_b
:valve_lift "0.007";
:contains[a :Output_section;
            :nominal_diameter 38;
            :shape :circle];
:conrains [a :Intermediate_section;
             :nominal_diameter 35;
             :shape :circle];
```

Figs. 6, 7, and 8 show the results of processing these data using scripts with minimal prepress of screenshots.

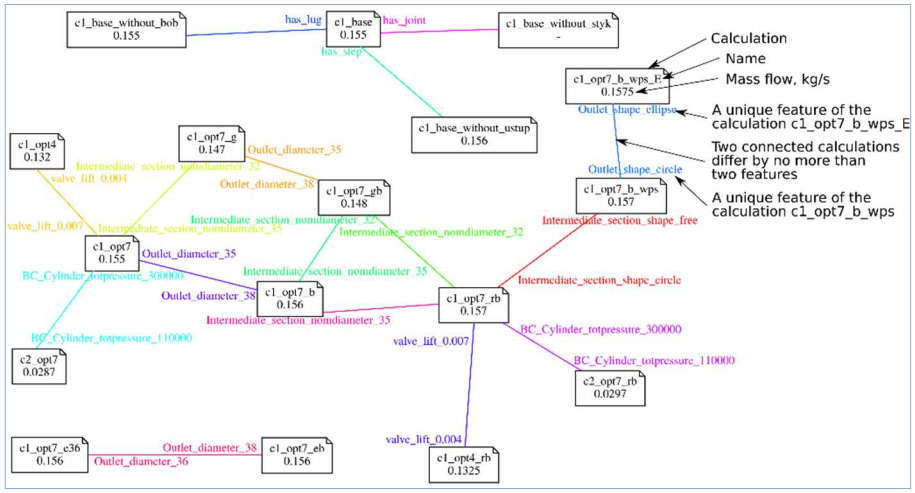

Fig. 6. An example of a proximity graph of calculations automatically generated from the knowledge base

Fig. 6 shows the proximity graph contracted by the program, where the connecting lines are drawn according to the criteria of "no more than two different points in the list of features". The program displays these different points at the corresponding end of the connecting line. At the graph nodes, in addition to the calculation name, the value of the flow through the channel is displayed. Analysis of the graph by a user makes it easy to select ways to increase consumption, which is the main purpose of profiling, and to identify parameters that do not affect this value.

If it is necessary to detail the differences between the calculations for two objects not connected by the graph edges, this can be done using the command to compare two calculations. The screenshot is shown in fig. 7.

Name of calculation	Unique features	General features
c1_base	• has_lug • has_joint • has_step • Intermediate_section_shape_free • Intermediate_section_nomdiameter_34.8	• valve_lift_0.007 • Outlet_shape_circle • Outlet_nomdiameter_35 • BC_Cylinder_totpressure_300000 • with_turbulence_model_mu_const_0.01 • with_soft_NSF_2017 • for_substance_air_politrop • BC_Outlet_pressure_100000
c1_opt7	• Intermediate_section_shape_circle • Intermediate_section_nomdiameter_35	

Fig. 7. An example of a table comparing two calculations on lists of automatically generated features

Another typical question that a calculation specialist has when analyzing the results is the degree of influence of a parameter on the simulation results. The simplest way to answer this question is to conduct a separate experiment, despite the fact that the required information may already be implicitly contained in previous calculations (and sometimes even explicitly). Using a typical method of logging an experiment, it may be more difficult to reveal this information than to recalculate. Using the repository allows performing the appropriate analysis automatically. The corresponding script splits the bank of calculations into pairs, one of which contains the desired feature, and the other does not. The script sorts them in the order of proximity of two members in a couple to each other. An example of such analysis for this project is shown in fig. 8.

```
Feature analysis :Outlet_nomdiameter_35

Similarity      Calc with a feature    Calc without a feature
Similarity 2 :  c1_opt7                c1_opt7_b
Similarity 2 :  c1_opt7_g              c1_opt7_gb
Similarity 3 :  c1_opt4                c1_opt4_rb
Similarity 3 :  c1_opt7                c1_opt7_rb
Similarity 3 :  c1_opt7_g              c1_opt7_rb
Similarity 3 :  c2_opt7                c2_opt7_rb
Similarity 4 :  c1_opt4                c1_opt7_b
Similarity 4 :  c1_opt7                c1_opt7_gb
Similarity 4 :  c1_opt7_g              c1_opt7_b
Similarity 4 :  c2_opt7                c1_opt7_b

Shown 10 nearest pairs of 72
```

Fig. 8. The result of the script that analyzes the presence of a particular feature in the knowledge base.

As one can see, the representation of the bank of calculations in the knowledge base format provides ample opportunities for analyzing and visualizing calculations, which are impossible (or extremely time-consuming) when using simpler methods of modeling logging.

8. Conclusion

As the result, a software package has been created that implements the functioning of the repository for CFD calculations based on a semantic knowledge base. To ensure its correct operation, an ontology has been also created, which formulates the information model of calculation. A number of scripts have been written to support a user during the process of analyzing data in a database.

Currently, there is the developed prototype of this project and the first version (0.1) is released. All scripts are published under the GNU GPL Version 3 open source license on the bitbucket platform [13]. The author plans to work on the project to develop the user interface and provide more repository capabilities.

References

[1]. Junker U., Mailharro D. The Logic of ILOG (J) Configurator: Combining Constraint Programming with a Description Logic. In Proc. of the 18th International Joint Conference on Artifical Intelligence (IJCAI-03), Configuration Workshop, 2003, pp. 13-20.

[2]. A. Felfernig, G. Friedrich, D. Jannach, M. Stumptner. Consistency-based diagnosis of configuration knowledge bases. Artificial Intelligence. Vol. 152. Issue 2. 2004. pp. 213-234. DOI: 10.1016/S0004-3702(03)00117-6.

[3]. Blondet G., Le Duigou J., Boudaoud N. ODE: An Ontology for Numerical Design of Experiments. Procedia CIRP, vol. 50. 2016, pp. 496-501. 10.1016/j.procir.2016.04.199.

[4]. Ajit S., Sleeman D., Fowler D., Knott, D. Constraint capture and maintenance in engineering design. Artificial intelligence for engineering design analysis and manufacturing, vol. 22, 2008, pp 325-343. DOI:10.1017/S089006040800022X.

[5]. M. Ito, D. Ishihara, M. Tsuchiya, M. Otsuka, K. Tsuchimoto, Y. Miyazaki. Development of CAE Standardization System for Motorcycle Parts with Navigating Function for Designers. Honda R&D Technical Review, vol. 23, № 2. 2011. pp 90-96.

[6]. SHustova D.V., Borgest N.M. Development of the semantic bases of information systems in the design and manufacture of engineering products. Otkrytye semanticheskie tekhnologii proektirovaniya intellektual'nykh sistem [Open semantic technologies for the design of intelligent systems], №7, 2017, pp. 293-296. (in Russian)

[7]. Nazarov D.M., Borgest N.M. Managing a CFD model using a linked external database. XIII Korolyovskie chteniya. Mezhdunarodnaya molodyozhnaya nauchnaya konferentsiya, sbornik trudov [XIII Korolev readings. International Youth Scientific Conference, proceedings], 2015, p. 114. (in Russian)

[8]. CML-Bench™ – computer activity management system. Available at: http://fea.ru/article/cml-bench, accessed 27.08.2018. (in Russian)

[9]. ANSYS EKM: Simulation Data and Process Management Available at: https://www.ansys.com/products/platform/ansys-ekm, accessed 27.08.2018

[10]. Richard Cyganiak, David Wood, Markus Lanthaler. RDF 1.1 Concepts and Abstract Syntax. W3C Recommendation, 25 February 2014. Available at: http://www.w3.org/TR/2014/REC-rdf11-concepts-20140225, accessed 27.08.2018

[11]. ISO 10303-1:1994 Industrial automation systems and integration - Product data representation and exchange - Part 1: Overview and fundamental principles.

[12]. Apache Jena - Apache Jena Fuseki. Available at: https://jena.apache.org/documentation/fuseki2/index.html, accessed 27.08.2018

[13]. Zenkin / repCAE – Bitbucket. Available at: https://bitbucket.org/zenkin/repcae, accessed 29.10.2018

Онтологический репозиторий для CFD-расчетов

В.А. Зенкин <vl.zenkin@gmail.com>
Федеральное государственное бюджетное образовательное учреждение высшего образования «Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет)»
105005, Россия, г. Москва, ул. 2-я Бауманская, д.5. стр.1

Аннотация. На основе RDF-хранилища создан программный комплекс для формирования базы знаний, содержащей информацию о проведенных гидрогазодинамических расчетах. Комплекс представляет собой набор скриптов, написанных на языке bash и python, которые опубликованы под открытой лицензией GNU GPL 3. Инструмент предназначен для поддержки расчетчика при проведении поисковых исследований, не имеющих строгого, наперед заданного плана эксперимента или алгоритма решения задачи. Для формализации описания расчета, хранящегося в базе знаний, создана онтология, служащая его информационной моделью. В качестве вспомогательного средства для проведения автоматизированного сравнения расчетов друг с другом разработан механизм «компараторов» и «особенностей», также описанный в статье. Помимо систематизированного хранения данных комплекс обеспечивает возможность их автоматического и полуавтоматического анализа, в том числе представление банка расчетов в форме неориентированного графа, построение плоских и пространственных зависимостей, поиск сходных расчетов и т.п. В статье приводятся примеры обработки данных проекта по профилированию каналов в головке цилиндров поршневого двигателя.

Ключевые слова: методология CAE; онтология; управление знаниями; вычислительная газодинамика; семантические технологии

DOI: 10.15514/ISPRAS-2018-30(5)-15

Для цитирования: Зенкин В.А. Онтологический репозиторий для CFD-расчетов. Труды ИСП РАН, том 30, вып. 5, 2018 г., стр. 249-264 (на английском языке). DOI: 10.15514/ISPRAS-2016-30(5)-15

Список литературы

[1]. Junker U., Mailharro D. The Logic of ILOG (J) Configurator: Combining Constraint Programming with a Description Logic. In Proc. of the 18th International Joint Conference on Artifical Intelligence (IJCAI-03), Configuration Workshop, 2003, pp. 13-20.

[2]. A. Felfernig, G. Friedrich, D. Jannach, M. Stumptner. Consistency-based diagnosis of configuration knowledge bases. Artificial Intelligence. Vol. 152. Issue 2. 2004. pp. 213-234. DOI: 10.1016/S0004-3702(03)00117-6.

[3]. Blondet G., Le Duigou J., Boudaoud N. ODE: An Ontology for Numerical Design of Experiments. Procedia CIRP, vol. 50. 2016, pp. 496-501. 10.1016/j.procir.2016.04.199.

[4]. Ajit S., Sleeman D., Fowler D., Knott, D. Constraint capture and maintenance in engineering design. Artificial intelligence for engineering design analysis and manufacturing, vol. 22, 2008, pp 325-343. DOI:10.1017/S089006040800022X.

[5]. M. Ito, D. Ishihara, M. Tsuchiya, M. Otsuka, K. Tsuchimoto, Y. Miyazaki. Development of CAE Standardization System for Motorcycle Parts with Navigating Function for Designers. Honda R&D Technical Review, vol. 23, № 2. 2011. pp 90-96.

[6]. Шустова Д.В., Боргест Н.М. Разработка семантических основ информационных систем при проектировании и производстве машиностроительных изделий. Открытые семантические технологии проектирования интеллектуальных систем, №7, 2017, стр. 293-296.

[7]. Назаров Д.М., Боргест Н.М. Управление CFD-моделью при помощи связанной внешней базы данных. XIII Королёвские чтения. Международная молодёжная научная конференция, сборник трудов, 2015, стр. 114.

[8]. CML-Bench™ – система управления деятельностью в области компьютерного инжиниринга URL: http://fea.ru/article/cml-bench. (Дата обращения 27.08.2018)

[9]. ANSYS EKM: Simulation Data and Process Management URL: https://www.ansys.com/products/platform/ansys-ekm. (Дата обращения 27.08.2018)

[10]. Richard Cyganiak, David Wood, Markus Lanthaler. RDF 1.1 Concepts and Abstract Syntax. W3C Recommendation, 25 February 2014. URL: http://www.w3.org/TR/2014/REC-rdf11-concepts-20140225. (Дата обращения 27.08.2018)

[11]. Государственный стандарт Российской Федерации, ГОСТ Р ИСО 13030-1-99, "Системы автоматизации производства и их интеграция. Представление данных об изделии и обмен этими данными. Часть 1. Общие представления и основополагающие принципы". ИПК Издательство стандартов, 1999.

[12]. Apache Jena - Apache Jena Fuseki URL: https://jena.apache.org/documentation/fuseki2/index.html. (Дата обращения 27.08.2018)

[13]. Zenkin / repCAE - Bitbucket URL: https://bitbucket.org/zenkin/repcae. (Дата обращения 29.10.2018)