Database for properties of nuclear reactor materials based on the ontology and NoSQL data format

Sergey A. Dyachkov[1,2][0000-0001-9606-2123], Adilbek O. Erkimbaev[1][0000-0002-5239-2208], Sergey Yu. Grigoryev[1,2][0000-0003-3492-3949], Pavel Yu. Korotaev[2][0000-0002-2617-6341], Andrey V. Kosinov[1][0000-0002-0290-0198], Pavel R. Levashov[1][0000-0002-8575-2847], Maxim A. Maltsev[1][0000-0001-6052-2319], Dmitry V. Minakov[1][0000-0002-6816-3768], Igor V. Morozov[1][0000-0002-0122-9400], Mikhail A. Paramonov[1][0000-0003-4105-6732], Aleksey V. Yanilkin[2][0000-0002-7457-9911], and Vladimir Yu. Zitserman[1][0000-0003-1714-4513]

[1] Joint Institute for High Temperatures of Russian Academy of Sciences, Moscow, Russia
[2] Dukhov Automatics Research Institute, Moscow, Russia

Abstract. In the paper, the development of a new generation database for the properties of nuclear reactor materials is discussed. The goal of this project is to accumulate experimental and theoretical data on the mechanical, thermophysical, and service properties of different materials, their microstructure, and transformations due to radiation-induced swelling, corrosion, and other degradation processes for future use in data analysis and computer simulations that can predict unknown material properties and therefore accelerate the development of new materials for contemporary and planned nuclear reactors. Due to the large variety of data standards, types, and formats, as well as the various physical quantities and their dimensions used in different data sources, the use of typical relational databases requires a huge amount of handwork for restructuring and fitting the data into the common tables. Moreover, adding new data types may require a redesign of the database architecture. In this case, NoSQL databases where the information is stored as a set of JSON documents might greatly improve the flexibility of the data storage, although searching, extracting, and merging the data from different documents becomes more complicated. As a solution for this problem, we propose using an ontology for data annotation and integration that allows one to store the datasets in their original forms and convert them on-the-fly when processing a particular user query.

Keywords: database · NoSQL · ontology · semantic search · machine learning · nuclear reactor materials

1 Introduction

Materials informatics is a rapidly developing branch of the material science concerned with the issues of data storage, annotation, and management suitable for further processing using contemporary statistical and machine learning techniques [5, 8, 14, 15]. Along with experiments and computer simulations, the machine learning algorithms has become an indispensable part of scientific research related to the discovery, optimization, and development of new materials. Perhaps one of the most prominent projects of such kind is the Materials Genome Initiative [21] which introduces a new paradigm in the material sciences. In particular, the creation of modern nuclear reactors relies greatly on advances in the material science, including studies of engineering materials, fuels, and coolants [19, 20].

At present, there are a few nuclear reactor material databases available [9], that have been built using the traditional relational scheme. Although these databases are very fast and reliable, they lack flexibility when a broad range of materials, properties, and experimental conditions are to be considered. It is of particular importance when storing large amount of raw experimental and simulation results for fundamental or applied research rather than developing a ready-to-use engineering database with complete and verified datasets. In the first case, the so-called non-relational or NoSQL databases [11] may be used, as demonstrated by such projects as Citrination [18] or NOMAD [12].

In document-oriented NoSQL databases, the data is stored in collections of documents or records, usually represented in the JSON format. Their advantages are as follows:

- the source data structure can be retained provided that it is represented as a set of JSON documents (which is possible for most of the data structures);
- a database can be appended by new data types without the need to redesign the structure of tables and their relations;
- updates and exchanges of data between different databases are fast and straightforward;
- NoSQL databases can be scaled almost infinitely with the growth of the information volume, particularly with the use of distributed computer systems, in contrast to the relational databases, where the extension is limited and strongly affects the performance;
- there is native support for storing binary files along with structured JSON documents for many of NoSQL databases (see, e.g. MongoDB [3]).

At the same time, searching and analyzing data in NoSQL databases can be quite tricky. As soon as the data is saved in different formats using various physical quantities and dimensions, the database becomes disintegrated unless special metadata or annotations are provided. In this work, we propose to use an ontology as a supplementary metadata subsystem.

The ontologies being generalized semantic data models are used in many scientific areas, including material sciences [6, 7, 17]. If the particular data items (JSON key-value pairs) are linked to the ontology classes, the data from different collections can be related to each other, and moreover, the internal links between ontology entities enable one to use a wide range of instruments for data processing and discovery without rearranging the data in the original storage. This way of data integration is discussed, e.g., in [13, 16]. Our idea of ontology-based data linkage is somewhat similar to the one presented in [6, 7], although it has many differences in its particular implementation.

In this paper, we briefly describe the idea and implementation of a new information system for nuclear reactor materials based on a NoSQL database, a custom-designed ontology, and a user-friendly web interface.

2 Database structure and ontology

The scheme of the information system is given in the figure 1. The users can interact with the database via a web interface, a Python API or command-line access (for administration and maintenance). The data is put into the storage subsystem, whose structure is presented in the figure 2.

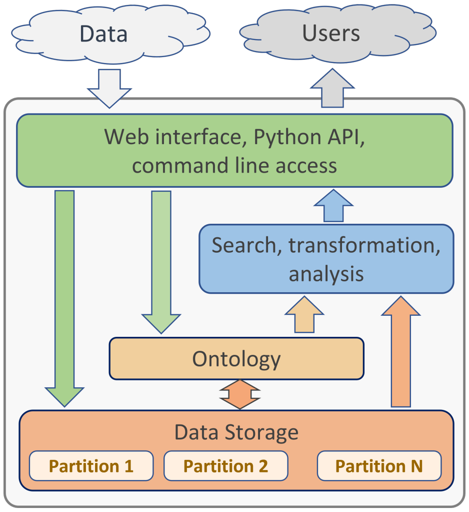

Data
Users
Web interface, Python API, command line access
Search, transformation, analysis
Ontology
Data Storage
Partition 1
Partition 2
Partition N

Fig. 1. The general scheme of the data flow in the information system for nuclear reactor materials.

The elementary data items in the database include key-value pairs, tables in the special format, and links to the binary files saved in the same storage. The hierarchical structure provided by the JSON format allows sets of data to be combined into objects that contain other objects, etc. The records represent the minimal amount of data that can be inserted into or extracted from the database. A set of records forms a collection that is somewhat similar to a table in a relational database (the records are similar to the table rows in this

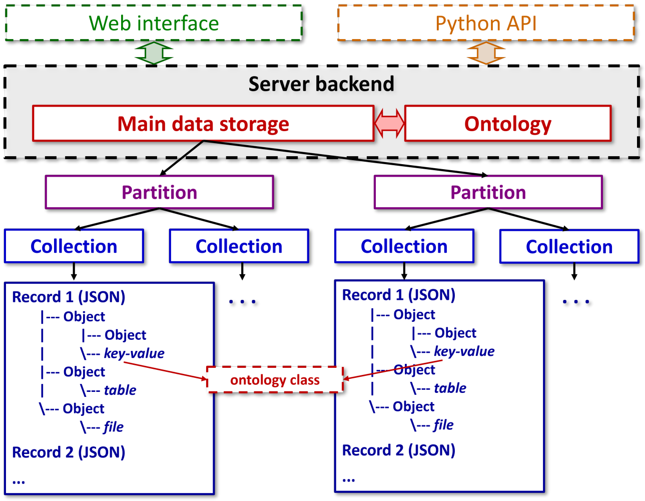


Fig. 2. The structure of the data storage subsystem. The red arrows show the linking of the data from different collections using the ontology class.

case). There are no constraints on the internal structure of JSON documents, but typically, all records in the same collection have similar JSON objects. For instance, when storing the results of a series of experiments or simulations of the same type, each record usually contains the results of a single experiment or simulation.

The partition contains one or more data collections. It is the minimal data structure for which the access rights of users and groups are defined. Typically, the partition is assigned to a specific organization or research group that provides the data. It can also be related to specific research or a literature review project.

The key feature of the developed system is the method of linking data from different collections. In our case, it is done by changing the names of JSON keys and/or values to the names of the corresponding ontology classes. It can be done for the keys that contain numerical or literal data as well as for the dimensions. Therefore, the ontology is used to annotate all or a part of the data in the collections. The linkage process is described in section 4 in more detail.

The fragment of the ontology is shown in the figure 3. It is usually represented as a set of classes and their subclasses that form an acyclic graph of linked entities. Each class can contain instances, e.g., the entity “Oxygen” is an instance of the classes “Atom”, “Molecule” and “ChemicalSubstance”. In course of this project, a special ontology for nuclear reactor materials was developed that contains more than 1000 classes and more than 3000 class instances describing various substances, engineering (structural) materials, devices, material properties (thermophysical, mechanical, microstructural, etc.), radiation-induced and general degradation processes, experiment and simulation types, as well as some general concepts. An early concept of such an ontology was presented in [10].

The upper level of the developed ontology contains the following classes: “Object”, “Process”, “Quality”, and “Unit”. The “Object” class describes both material and information objects, where the material objects include elementary particles, chemical substances, complex materials like alloys, fuels, etc., various samples, constructive objects, typical equipment of nuclear reactors and testing laboratories. The information objects represent auxiliary classes for describing different data types, tables, files, and software. The next upper-level class “Process” is responsible for the description of general chemical and physical processes, manufacturing processes, and material treatment during their operation or testing. According to the main purpose of the database, the focus is on material degradation processes like radiation swelling, embrittlement, or corrosion. The class “Quality” contains subclasses for the properties of material and information objects, as well as the process characteristics; it also includes most of the general physical quantities. Finally, the “Unit” class has subclasses for all types of physical quantities used in the database, grouped by their their units of measure. Each of these subclasses has a few instances representing individual units that can be converted to each other using the coefficients and offsets stored as numeral properties of the instance. In particular, the unit class instances are used by the web interface for conversion of units “on the fly” when retrieving information from the database.

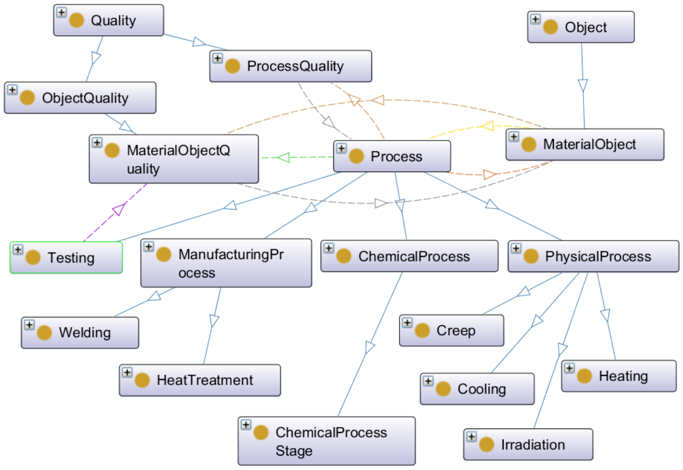


Fig. 3. A fragment of the ontology for nuclear reactor materials. The square boxes denote ontology classes; the solid lines correspond to the relations “class-subclass” while the dashed ones correspond to all other semantic relations.

The ontology classes can be linked to each other in different ways. Along with the links of the type “class-subclass” also known as “is a” relation, that form the main (vertical) structure of the ontology, there could be horizontal links between the classes that define their semantic relations. In this work, we used the following types of such relations (annotation properties): “has participant”, “determines”, “describes”, “has attribute”, has quality”, “is related to”, “has part”, “has unit” and their inverse counterparts. The two special annotation properties “has key”, and “has value” are reserved for special use when marking the collections in which the keys are linked to a particular class or a class instance.

As shown in the figure 1, the ontology is used for searching and analyzing the data. In the simplest case, the ontology can be used as a glossary of terms for a search query. An extended “semantic search” mode allows one to obtain semantically linked quantities, i.e., the terms linked with the original ones given in the search query, using one or more ontology class relations described above. For example, the semantic search for “Dislocation” retrieves the records that have the key "Dislocation” as well as the related keys “DislocationLoop”, “DislocationDensity”, “DislocationLoopDensity”, and “DislocationLoopSize”. The types of relations used for this search type are controlled by the user. With the help of ontology, it is also possible to get a list of materials that have the above mentioned properties or a list of processes that create or are affected by dislocations. Thus, the diversity of possible class links makes semantic search an efficient tool for immediate data comparison and later use in machine learning algorithms.

3 Implementation details

The software tools used in the project are shown in figure 4. The main data storage is maintained by the MongoDB database, while the ontology is stored in the graph-oriented RDF database AllegroGraph [1]. On one side, the use of two different engines for JSON and RDF databases causes some inconveniences related to ensuring the consistency of the data; each of these engines can be potentially used for both RDF and JSON databases. On the other side, the use of MongoDB provides the best scalability for storing JSON documents and binary files, while AllegroGraph has the best performance for querying the ontology using the SPARQL language.

An additional tiny SQL database is used for the user-specific data, access rights, and general description of the data partitions. The SQL database is also used for some technical purposes, such as configuring caches and web pages. In particular, we use SQLite for a compact standalone installation of the system and PosgreSQL for a multi-user access.

It should be noted, that some relational DBMS allow have extension to operate with graphs and therefore they could be used for storing the ontology which avoid the use of AllegroGraph or another RDF engines. Their testing and implementation in this project are left for future work.

The user interface is based on the Django framework [2] using the Webix JavaScript library [4] on the client side.

4 User interface and access rights

The database can be accessed from either a local or remote host via both the Python API and the web interface. A fragment of the web interface is shown in figure 5. The main menu in the navigation panel directs the user to sections “Home”, “Search”, “Data”, “Ontology”, “Simulation”, and to the administrative panel (via the context menu when clicking on the user name). The language button allows one to switch between Russian and English. All the sections are described in detail below.

The “Home” section has a few pages with a general description of the project and the database summary.

The “Search” section is responsible for searching the data using different search modes and displaying the results in the form of customizable tables or plots. The search is based on the query string, which in a simple mode contains a number of JSON key names, optionally followed by relation singes, values, and units. These elements can be combined using “AND” and “OR” operators (“AND” is used by default), e.g., “Temperature $>$ 500[K] Temperature $<$ 800[K] (HeatCapacity OR Enthalpy)”. The key names are usually taken from the ontology, but it is not obligatory, so that custom user keys not linked to the ontology can be used as well. If the units of the retrieved numerical values are defined and linked to the ontology, they are automatically converted to the units selected by the user in the search results table. It is especially important when retrieving values from different data collections where the dimensions of the values can differ.

In the semantic search mode, the query contains the element in the form “ClassA ClassB ... options” where the ontology classes “ClassA”, “ClassB“ can be supplemented by options like types of the semantic relations described above, the depth of subclasses to be searched, and exclusions of selected classes. The semantic and simple search queries can be combined. Therefore, the data can be retrieved from the database even without using the ontology (in the simple search mode), but the use of the ontology extends the capabilities of the search significantly.

The “Data” section is used for importing, editing, or exporting the data, as well as linking it to the ontology. The screenshot given in figure 5 shows the data editor page for a particular data collection. On the left side, the JSON structure of one of the records is displayed, where some of the JSON keys are highlighted in blue, indicating that this data is linked to the corresponding ontology class.

During the linkage of a JSON key that can represent a key–value pair or an object, the user invokes a context menu by clicking on the corresponding key. If the key name coincides with the name of an ontology class (e.g., “Temperature”), the web interface suggests the corresponding link, and after user approval, the triplet “CollectionName – has key – Temperature” is added to the ontology, where “CollectionName” is the name of the instance of the special class “DataCollection”. Such instances are created for all the collections at their creation. After the insertion of this triplet, all keys named “Temperature” in the current collection are considered to be linked to the ontology. This rule obviously re-

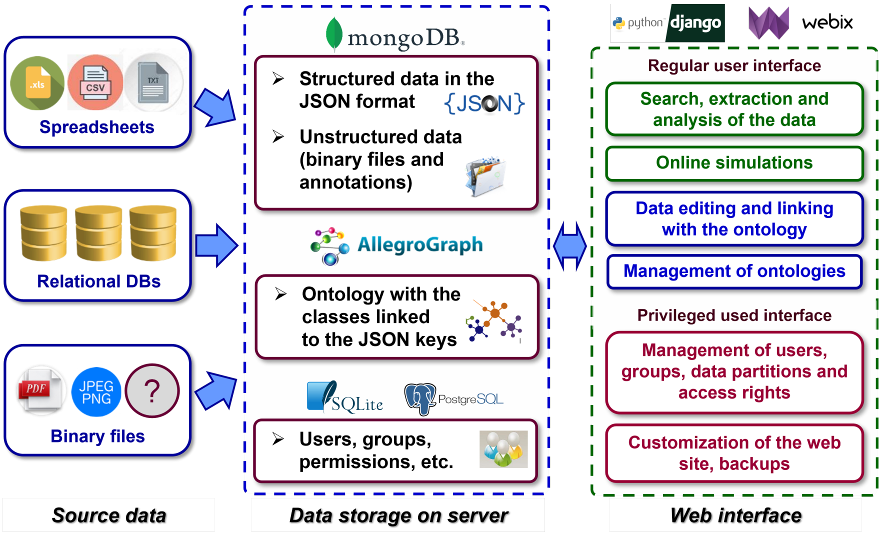
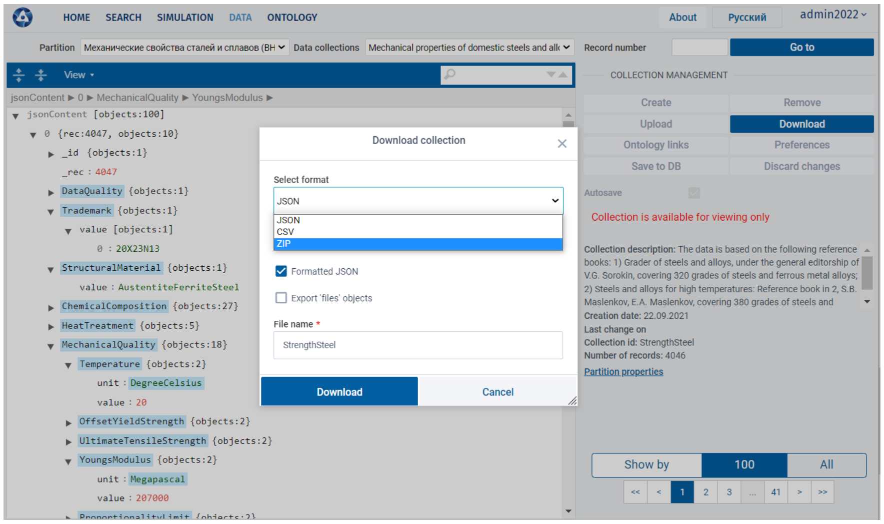

Spreadsheets
Relational DBs
JPEG
PNG
?
Binary files
Source data
mongoDB.
Structured data in the JSON format {JSON}
Unstructured data (binary files and annotations)
AllegroGraph
Ontology with the classes linked to the JSON keys
SQLite
PostgreSQL
Users, groups, permissions, etc.
Data storage on server
python
django
webix
Regular user interface
Search, extraction and analysis of the data
Online simulations
Data editing and linking with the ontology
Management of ontologies
Privileged used interface
Management of users, groups, data partitions and access rights
Customization of the web site, backups
Web interface

Fig. 4. Software components and their interactions.



Partition: Механические свойства сталей и сплавов (ВН

Data collections: Mechanical properties of domestic steels and allo

Record number

jsonContent ▶ 0 ▶ MechanicalQuality ▶ YoungsModulus ▶

```
▼ jsonContent [objects:100]
  ▼ 0 {rec:4047, objects:10}
    ▶ _id {objects:1}
      _rec : 4047
    ▶ DataQuality {objects:1}
    ▼ Trademark {objects:1}
      ▼ value [objects:1]
          0 : 20X23N13
    ▼ StructuralMaterial {objects:1}
        value : AustentiteFerriteSteel
    ▶ ChemicalComposition {objects:27}
    ▶ HeatTreatment {objects:5}
    ▼ MechanicalQuality {objects:18}
      ▼ Temperature {objects:2}
          unit : DegreeCelsius
          value : 20
      ▶ OffsetYieldStrength {objects:2}
      ▶ UltimateTensileStrength {objects:2}
      ▼ YoungsModulus {objects:2}
          unit : Megapascal
          value : 207000
      ▶ ProportionalityLimit {objects:2}
```

Download collection

Select format: JSON

JSON
CSV
ZIP

☑ Formatted JSON

☐ Export 'files' objects

File name *: StrengthSteel

Autosave ☑

Collection is available for viewing only

Collection description: The data is based on the following reference books: 1) Grader of steels and alloys, under the general editorship of V.G. Sorokin, covering 320 grades of steels and ferrous metal alloys; 2) Steels and alloys for high temperatures: Reference book in 2, S.B. Maslenkov, E.A. Maslenkov, covering 380 grades of steels and
Creation date: 22.09.2021
Last change on
Collection id: StrengthSteel
Number of records: 4046
Partition properties

Fig. 5. A sample view of the web data editor with an opened pop-up window for downloading the collection.

stricts the use of arbitrary JSON key names but greatly simplifies the linkage process. If the original key name is different from the destination ontology class name, it is to be renamed within the whole collection. Alternatively, for one-to-one linkage, the user can simply start automatic linking of all the keys that are named properly.

After the linkage, the key–value pairs are converted into objects, where the value is stored in a special "value" key. There are also rules for automatic extraction of the dimensions of physical quantities and storing them in the special "unit" key.

The mapping of key values and units to the ontology classes can be done in a similar way, except that the relations "has value" and "has unit" are used instead of "has key" and the information of the JSON path within the record is also stored in the ontology. Thus, if the user links the value "AustentiteFerriteSteel" of the key "StructuralMaterial" (see figure 5), it does not affect the values of other JSON keys or the values of the keys "StructuralMaterial" located in nested JSON objects.

The data can be entered via the web editor shown in figure 5 or imported from a JSON, CSV, or ZIP file, where ZIP archive is used for importing binary files that are to be attached to the collection. The imported records can be added to an existing collection. In this case, all ontology links are applied to these new records automatically.

On the right side of figure 5, one can see the collection management tools (some of which are disabled for the current user). In the front, a pop-up window is open, showing the dialogue box for exporting the collection in the different formats mentioned above.

The "Ontology" section allows browsing, searching, and modifying the ontology. In the current version of the system, modification of the ontology via the web interface is limited to changing the properties of existing classes without the possibility to add or remove classes. A new version of the ontology as a whole can be uploaded as an OWL file using the administrative interface. As the ontology has many classes that are still not linked to any data and are reserved for future use, we expect that its extension may not be needed too frequently.

The "Simulation" section contains a set of simulation modules in Python that can perform calculations for particular problems using the information from the database. This section is still under development. At present, it includes web forms for the calculation of yield strength, tensile strength, and heat resistance; the calculation of swelling depending on the radiation dose and temperature; the calculation of the parabolic corrosion rate constant; and the search for new compositions with lower corrosion rates using Bayesian optimization. In this version, the forms use predefined models obtained via machine learning techniques using preliminary selected data sets. In the future, it is planned to extend the capabilities of this module to use arbitrary data from the database.

The administrative panel can be used to maintain users, user groups, access rights, data partitions, and site preferences. Access rights to a particular data partition can be assigned to a group of users. The types of these rights include “none”, “read”, “read/write”, and “read/write/manage”. The latest type enables the users of the group to create, delete, or edit the properties of data collections. There are also five user roles: “guest”, “reader”, “editor”, “admin”, and “superuser”. The “superuser” has full access to the data, and moreover, he is responsible for managing data partitions, the creation and removal of user groups, the editing of user group rights, and the designation of a group administrator (“admin”). The “admin” user can add or remove other users within the same group and change their roles. The “guest” user has access only to special information resources, not to the main data storage. The access policy for different users depending on their role is given in table 1.

Table 1. Assess rights to a data partition for the users depending on their role and the access level of the corresponding group of users.

User role	Access level of a group of users to a data partition			
	none	read	read/write	read/write/manage
guest	no access			
reader	no access	read	read	read
editor	no access	read	read/write	read/write/manage
admin	no access	read	read/write	read/write/manage
superuser	full access			

5 Conclusions

The concept and implementation of the information system based on the NoSQL database and ontology metadata are presented. Although relational databases provide better performance for highly structured data, semi-structured data in different formats is more convenient to store as collections of JSON documents. It ensures better flexibility, scalability, and simple updates. At the same time, the ontology is found to be an appropriate instrument for linking the data from different collections, which results in overall data integrity and enables advanced search and analysis instruments based on the semantic relations between the ontology classes. The data can be accessed via both the web interface and the Python API. The system is still under development, although all the main features are implemented. The future plans are related to the extension of the search possibilities and adding machine learning and other algorithms that will make full use of the ontology metadata.

Acknowledgements This work is supported by the Russian State Atomic Energy Corporation “Rosatom”.

References

1. AllegroGraph graph database project web site, https://allegrograph.com/ [Accessed: 31 May 2023]
2. Django framework web site, https://www.djangoproject.com/ [Accessed: 31 May 2023]
3. MongoDB nosql database project web site, https://www.mongodb.com/ [Accessed: 31 May 2023]
4. Webix Javascript UI library web site, https://ru.webix.com/ [Accessed: 31 May 2023]
5. Agrawal, A., Choudhary, A.: Perspective: Materials informatics and big data: Realization of the "fourth paradigm" of science in materials science. APL Materials **4**(5), 053208 (2016)
6. Ashino, T.: Materials ontology: An infrastructure for exchanging materials information and knowledge. Data Science Journal **9**, 54–61 (2010)
7. Ashino, T., Nishikawa, N., Kadohira, T.: Data analysis environment for materials science and engineering integrating heterogeneous data resources. In: Elizarov, A., Novikov, B., S, S. (eds.) Data Analytics and Management in Data Intensive Domains: Proccesings of XXI International Conference DAMDID/RCDL'2019 (October 15–18, 2019, Kazan, Russia). p. 420. Kazan Federal University, Kazan (2019)
8. Austin, T.: Towards a digital infrastructure for engineering materials data. Materials Discovery **3**, 1–12 (2016)
9. Belov, G.V., Aristova, N.M.: Databases on the properties of materials and substances for nuclear power. Matematicheskoe modelirovanie **29**(6), 135–142 (2017)
10. Chusov, I.A., Kirillov, P.L., Pronyaev, V.G., Erkimbaev, A.O., Zitserman, V.Y., Kobzev, G.A., Fokin, L.R.: Ontologies and databases on thermophysical properties of nuclear reactor materials (in russian). Izvestiya vuzov. Yadernaya Energetika (1), 5–18 (2019)
11. Davoudian, A., Chen, L., Liu, M.: A survey on nosql stores. ACM Computing Surveys (CSUR) **51**(2), 1–43 (2018)
12. Draxl, C., Scheffler, M.: The NOMAD laboratory: from data sharing to artificial intelligence. Journal of Physics: Materials **2**(3), 036001 (2019)
13. Erkimbaev, A.O., Zitserman, V.Y., Kobzev, G.A., Kosinov, A.V.: Integration of information resources containing data on the properties of substances and materials: Practical implementation and existing tools. Automatic Documentation and Mathematical Linguistics **52**, 257–264 (2018)
14. Himanen, L., Geurts, A., Foster, A.S., Rinke, P.: Data-driven materials science: status, challenges, and perspectives. Advanced Science **6**(21), 1900808 (2019)
15. Kalidindi, S.R., De Graef, M.: Materials data science: current status and future outlook. Annual Review of Materials Research **45**, 171–193 (2015)
16. Kosinov, A., Erkimbaev, A., Zitserman, V.Y., Kobzev, G.: Ontology-based methods of thermophysical data integration. Journal of Physics: Conference Series **1385**(1), 012033 (2019)
17. Li, H., Armiento, R., Lambrix, P.: An ontology for the materials design domain. In: The Semantic Web–ISWC 2020: 19th International Semantic Web Conference, Athens, Greece, November 2–6, 2020, Proceedings, Part II 19. pp. 212–227. Springer (2020)
18. Michel, K., Meredig, B.: Beyond bulk single crystals: a data format for all materials structure–property–processing relationships. MRS Bulletin **41**, 617–623 (2016)

19. Morgan, D., Pilania, G., Couet, A., Uberuaga, B.P., Sun, C., Li, J.: Machine learning in nuclear materials research. Current Opinion in Solid State and Materials Science **26**(2), 100975 (2022)
20. Novoselov, I.I., Savin, D.I., Yanilkin, A.V.: The effect of irradiation conditions on generation of defects and their clusters. Journal of Nuclear Materials **546**, 152762 (2021)
21. de Pablo, J.J., Jackson, N.E., Webb, M.A., Chen, L.Q., Moore, J.E., Morgan, D., Jacobs, R., Pollock, T., Schlom, D.G., Toberer, E.S., et al.: New frontiers for the materials genome initiative. npj Computational Materials **5**(1), 41 (2019)