The Materials Design Ontology

Patrick Lambrix [a,b,c,*], Rickard Armiento [c,d], Huanyu Li [a,c], Olaf Hartig [a], Mina Abd Nikooie Pour [a,c] and Ying Li [a,c]
[a] *Department of Computer Science, Linköping University, Sweden*
E-mails: patrick.lambrix@liu.se, huanyu.li@liu.se, olaf.hartig@liu.se, mina.abd.nikooie.pour@liu.se, ying.li@liu.se
[b] *Department of Building Engineering, Energy Systems and Sustainability Science, University of Gävle, Sweden*
E-mail: patrick.lambrix@liu.se
[c] *The Swedish e-Science Research Centre, Linköping University, Sweden*
E-mails: patrick.lambrix@liu.se, rickard.armiento@liu.se, huanyu.li@liu.se, mina.abd.nikooie.pour@liu.se, ying.li@liu.se
[d] *Department of Physics, Chemistry and Biology, Linköping University, Sweden*
E-mail: rickard.armiento@liu.se

Abstract. In the materials design domain, much of the data from materials calculations is stored in different heterogeneous databases with different data and access models. Therefore, accessing and integrating data from different sources is challenging. As ontology-based access and integration alleviates these issues, in this paper we address data access and interoperability for computational materials databases by developing the Materials Design Ontology. This ontology is inspired by and guided by the OPTIMADE effort that aims to make materials databases interoperable and includes many of the data providers in computational materials science. In this paper, first, we describe the development and the content of the Materials Design Ontology. Then, we use a topic model-based approach to propose additional candidate concepts for the ontology. Finally, we show the use of the Materials Design Ontology by a proof-of-concept implementation of a data access and integration system for materials databases based on the ontology. [1]

Keywords: Ontology, Ontology development, Data access, Data integration, Materials science, Materials Design Ontology

1. Introduction

Materials design and materials informatics is central for technological progress, not the least in the green engineering domain. Many traditional materials contain toxic or critical raw materials, whose use should be avoided or eliminated. Also, there is an urgent need to develop new environmentally friendly energy technology. Presently, relevant examples of materials design challenges include energy storage, solar cells, thermoelectrics, and magnetic transport [8, 10, 24].

The space of potentially useful materials yet to be discovered — the so-called *'chemical white space'* — is immense. The possible combinations of, say, up to six different elements, constitute many billions. The space is further extended by possibilities of different phases, low-dimensional systems, nanostructuring, and so forth, which adds several orders of magnitude. This space was traditionally explored by experimental techniques, *i.e.*, materials synthesis and subsequent experimental characterization. Parsing and searching the full space of possibilities this

*Corresponding author. E-mail: patrick.lambrix@liu.se.
[1] This paper is an extension of [30] with results from [1] and currently unpublished results regarding an application using the ontology.

way is, however, hardly practical. Recent advances in condensed matter theory and materials modeling make it possible to generate reliable materials data by means of computer simulations based on quantum mechanics [28]. High-throughput simulations combined with machine learning can speed up progress significantly and also help to break out of local optima in composition space to reveal unexpected solutions and new chemistries [21]. The progress brought by the combination of machine learning models and databases of materials data, is now so rapid that it can be discussed as a lead-up to a *singularity* for the field of materials design [6].

This development has led to several global efforts to assemble and curate databases that combine experimentally known and computationally predicted materials properties, along with a desire to make them interoperable. These efforts have collectively been referred to as the Materials Genome Initiative (https://www.mgi.gov/). A central idea is that materials design challenges can be addressed by searching these databases for entries with desired combinations of properties. Nevertheless, these data sources also open up for *materials informatics*, i.e., the use of big data methodology and data mining techniques to discover new physics from the data itself. A workflow for such a discovery process can be based on a typical data mining process, where key factors are identified, reduced and extracted from heterogeneous databases, similar materials are identified by modeling and relationship mining and properties are predicted through evaluation and understanding of the results from the data mining techniques [3]. The use of the data in such a workflow requires addressing problems with data integration, provenance, and semantics.

Even when a new material has been invented and synthesized in a lab, much work remains before it can be deployed. Production methods allowing manufacturing the material at large scale in a cost effective manner need to be developed, and integration of the material into the production must be realized. Furthermore, life-cycle aspects of the material need to be assessed. Today, this post-invention process takes typically about two decades [24, 32]. Shortening this time is in itself an important strategic goal, which could be realized with the help of an integrated informatics approach [24].

It is clear that materials data, experimental as well as simulated, has the potential to speed up progress significantly in many steps in the chain starting with materials discovery, all the way to marketable product. However, the data needs to be suitably organized and easily accessible, which in practice is highly nontrivial to achieve. It requires a multidisciplinary effort and the various conventions and norms in use need to be integrated. Materials data is highly heterogeneous [32].

In this paper we address the data access and interoperability issue by developing an ontology suitable for the OPTIMADE (Open Databases Integration for Materials Design, https://www.optimade.org/) effort. The OPTIMADE consortium aims to make materials databases interoperable by developing a specification for a common REST API [5]. The consortium includes many of the data providers in computational materials science. However, although a first version of a common API has been defined, there is no semantic and integrated access support yet. Therefore, in this paper, we develop the Materials Design Ontology (MDO) that covers the content and is guided and inspired by the databases in OPTIMADE. Further, we provide a proof-of-concept implementation of a data access and integration system that currently covers two of the databases in the OPTIMADE consortium. The used framework is general and in the future other databases as well as the OPTIMADE API will be added to the implementation.

The paper is organized as follows. In section 3 we describe the development of MDO while the ontology itself is described in section 4. In section 5 we propose new concepts for an extension of MDO. Currently, these concepts are under discussion. In section 6 we show the use of MDO in our MDO proof-of-concept implementation of a data access and integration system for materials science databases. The paper concludes in section 7. We start with some background in section 2.

2. Background

2.1. Ontologies in Materials Science

A number of ontologies in materials science have been developed and we show some characteristics from the knowledge representation and the materials science perspectives in Table 1.

EMMO (earlier known as European Materials & Modelling Ontology, and recently renamed Elemental Multi-perspective Material Ontology, https://github.com/emmo-repo/EMMO) is a top level ontology with the purpose to develop a standard representational ontology framework based on knowledge of materials modeling and characterization. Another top level ontology that models materials is the Basic Formal Ontology (BFO). In BFO, material entities are continuants that include some portion of matter as part [39].

Most ontologies, however, are domain ontologies that focus on specific sub-domains of the materials field (Domain in Table 1) and have been developed with a specific use in mind (Application Scenario in Table 1). MatOnto [9], based on the upper ontology DOLCE [20], aims to represent structured knowledge, properties and processing steps relevant to materials for data exchange, reuse and integration. MatOWL [44] is extracted from MatML schema data to enable ontology-based data access. The Materials Ontology in [7] was designed for data exchange among thermal property databases, particularly focusing on representing knowledge relevant to material processing, measurement methods and manufacturing processes. The NanoParticle Ontology [41], based on the upper ontology BFO, and the eNanoMapper ontology [23] are two ontologies in the nanotechnology domain. The former represents properties of nanoparticles to design new nanoparticles, while the latter focuses on assessing risks caused by the use of nanomaterials in engineering. Extensions to these ontologies are computed in [29]. The MMOY ontology [45] captures metal materials knowledge from Yago. The Materials and Molecules Basic Ontology (MAMBO, [26]) focuses on concepts and relations emerging on materials where the relationship between individual molecules and molecular aggregation is relevant to the properties of the system, such as in molecular materials and nanomaterials. MAMBO integrates with EMMO, CheBI [12] and MDO. The Dislocation Ontology [43] focuses on representing knowledge related to crystalline materials and reuses some concepts in MDO. The Platform MaterialDigital Ontology (PMD, [4]) is a prototype to describe materials science experiments. The Materials Design Ontology (MDO, [30], https://w3id.org/mdo/), which is the focus of this paper, is inspired by OPTIMADE, and aims to enable semantic and integrated querying over multiple heterogeneous materials databases such as Materials Project [24], OQMD [36], NOMAD [16] and AFLOW [11].

From the knowledge representation perspective, the basic terms defined in materials ontologies involve materials, properties, performance, and processing in specific sub-domains. All presented ontologies use OWL as a representation language (Language in Table 1). The number of OWL classes ranges from a few to several thousands (Ontology Metrics in Table 1). Some ontologies have more concepts than relationss (e.g., MatOnto, Materials Ontology, NanoParticle Ontology, MMOY and EMMO), while some have much more properties (e.g., MDO). Several ontologies are developed in a modular fashion (Modularity in Table 1).

2.2. Ontology development

In Section 3 we describe the development of the Materials Design Ontology (MDO). Although, we could have used a more modern approach such as the eXtreme Design methodology [34] or its extension that integrates debugging [15], as our initial ontology was expected to be of a smaller size and given our earlier experience with the NeOn methodology for ontology engineering, we decided to use NeOn.

NeOn [40] is a methodology for ontology engineering that proposes nine scenarios which commonly occur, including Scenario 1: From Specification to Implementation, Scenario 2: Reusing and re-engineering non-ontological resources, Scenario 3: Reusing ontological resources, Scenario 4: Reusing and re-engineering ontological resources, Scenario 5: Reusing and merging ontological resources, Scenario 6: Reusing, merging, and re-engineering ontological resources, Scenario 7: Reusing ontology design patterns (ODPs), Scenario 8: Restructuring ontological resources, and Scenario 9: Localizing ontological resources. Depending on different background knowledge resources and purposes of the ontology, developers can make use of different scenarios or combinations of the scenarios. Scenario 1 is necessary in any ontology development and should always be included. The detailed use of NeOn for the development of MDO is described in Section 3.

Further, we also used two tools for detecting defects in the ontology during the development. The first tool, OOPS! [33], helps to detect some of the most common pitfalls appearing within ontology development. The second tool, RepOSE [25], allows to debug an ontology and proposes additional knowledge that could be interesting to add to the ontology.

Table 1
Characteristics of some materials ontologies

Ontologies	Knowledge Representation Perspective: Ontology Metrics	Language	Modularity	Materials Science Perspective: Domain	Application Scenario
EMMO	309 concepts, 35 relations, 3 instances	OWL	✓	Materials science	Upper ontology
MatOnto [9]	78 concepts, 10 relations, 24 instances	OWL	✓	Crystals	Materials discovery
MatOWL [44]	(not available)	OWL		Materials	Semantic querying
Materials Ontology [7]	606 concepts, 31 relations, 488 instances	OWL	✓	Thermal properties	Data exchange, search
ELSSI-EMD ontology [18]	35 concepts, 37 relations, 33 instances	OWL	✓	Materials testing	Standardization
NanoParticle Ontology [41]	1904 concepts, 81 relations	OWL		Nanotechnology	Data integration, search
eNanoMapper [23]	12781 concepts, 5 relations 464 instances	OWL	✓	Nanotechnology	Data integration
MMOY [45]	2325 concepts, 9 relations, 1738 instances	OWL		Metals	Knowledge extraction
MAMBO [26]	26 concepts, 33 relations	OWL	✓	Molecules-based materials	Knowledge representation
Dislocation Ontology [43]	18 concepts, 16 relations	OWL	✓	Crystalline Materials	Knowledge representation
PMD [4]	13 concepts, 7 relations	OWL	✓	Materials experiments	Knowledge representation, Data curation
MDO	37 concepts, 64 relations	OWL	✓	Materials design	Semantic querying over multiple databases

2.3. Ontology extension

In Section 5 we describe work on generating new concepts that may be added to MDO. The new concepts are, however, not yet included in the public version of MDO as discussions regarding the scope and the use of the extension are ongoing.

We used the phrase-based topic model generation approach we presented in [29], shown in Figure 1. Given a corpus of documents related to the domain of interest and the number of requested topics, a phrase-based topic model is created using an extended version of the ToPMine [17] system as presented in [1].

First, frequent contiguous phrases are mined, which consists of collecting aggregate counts for all contiguous words satisfying a user-defined minimum support threshold. In our extended version we also introduce a user-defined maximum threshold for word occurrences to, if so desired, remove very general words. Then the documents are segmented based on the frequent phrases. Further, an agglomerative phrase construction algorithm merges the frequent phrases guided by a significance score.

After this phrase mining, the system performs topic modelling by computing representations of latent topics in the documents. Topics are generated using a variant of Latent Dirichlet Allocation, called PhraseLDA, that deals with phrases, rather than words. Essentially, topics can be seen as a probability distribution over words or phrases.

The phrases as well as the topics are suggestions that a domain expert should validate or interpret and relate to concepts in the ontology. Based on the validations and interpretations of the domain expert, concepts and axioms are added to the ontology. To help a domain expert with the validation we implemented a tool of which an early version is described in [2]. The tool follows the iterative workflow of validation supporting the different phases. In the *Phrases* phase, the domain expert can look for sub- and super-phrases in the list as well as use string matching to find existing concepts in the ontology with similar names (Figure 2) . This is useful to obtain an overview of the phrases as well as to find initial connections to the concepts in the ontology. In the *From Phrase to Concept* phase, the domain expert processes the frequent phrases and decides for each phrase whether one or more concepts related to the phrase can be defined in the ontology. The tool supports the addition of these concepts (Figure 3). In the *Concept* phase new concepts (also unrelated to the phrases) can be added (Figure 4). In our experiments the domain

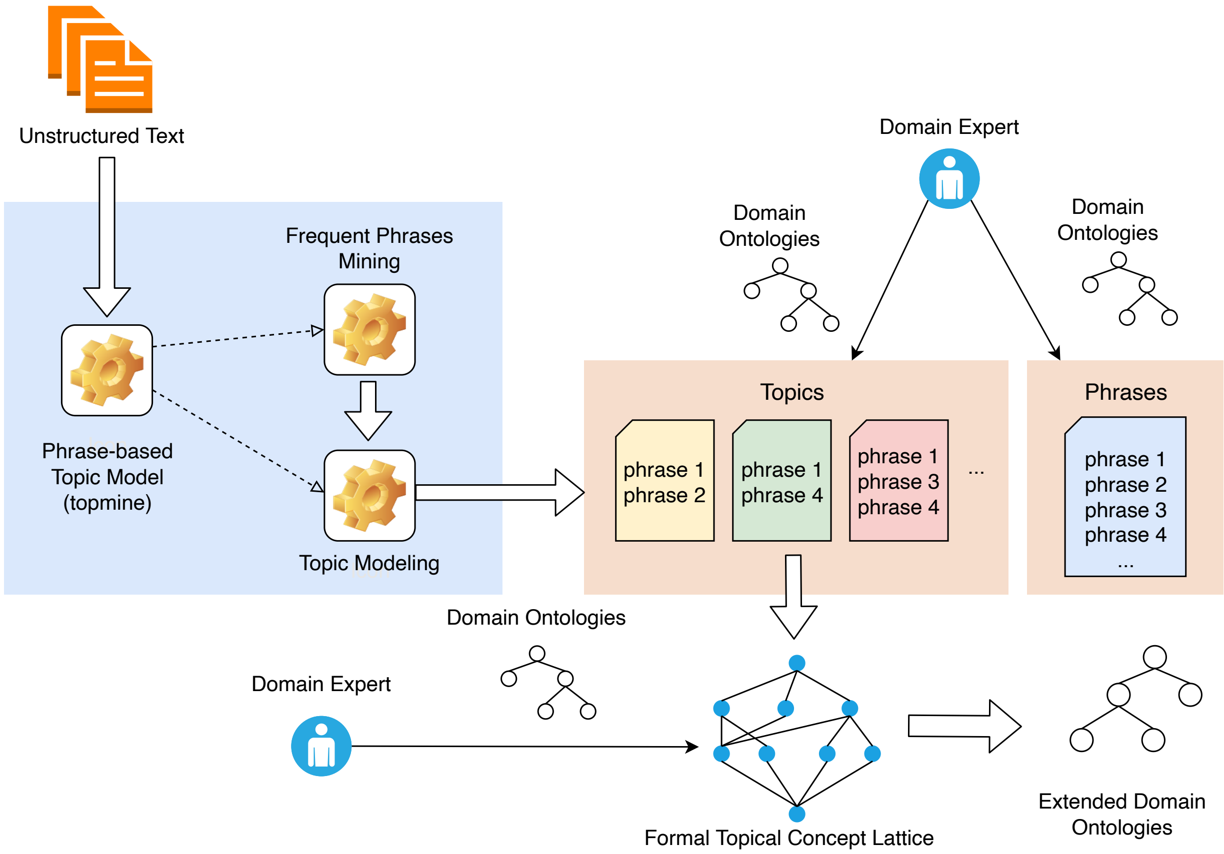


Figure 1. Approach: The upper part of the figure shows the creation of a phrase-based topic model with unstructured text as input and phrases and topics as output. The lower part shows the formal topical concept analysis with as input topics and as output a topical concept lattice. In both parts a domain expert validates and interprets the results. [29]

experts requested such functionality as they often thought of related concepts during the validation process. Finally, in the *From Concept to Axiom* phase, support for adding axioms related to the concepts is provided (Figure 5).

3. Developing MDO

The development of MDO followed the NeOn ontology engineering methodology [40]. We focused on applying scenario 1 (*From Specification to Implementation*), scenario 2 (*Reusing and re-engineering non-ontological resources*), scenario 3 (*Reusing ontological resources*) and scenario 8 (*Restructuring ontological resources*). We did not re-engineer or merge ontological resources, i.e., scenarios 4-6 (but did reuse some concepts from other ontologies) and did not translate MDO into another natural language (scenario 9). Further, we did not use existing ontology design patterns (scenario 7), as the only one we are aware of in the materials science field is about materials transformation [42] that is not covered by MDO.

We used OWL2 DL as the representation language for MDO. During the whole process, two knowledge engineers, and one domain expert from the materials design domain were involved. In the remainder of this section, we introduce the key aspects of the development of MDO.

3.1. Requirements Analysis.

During this step, we clarified the requirements by proposing Use Cases (UC), Competency Questions (CQ) and additional restrictions.

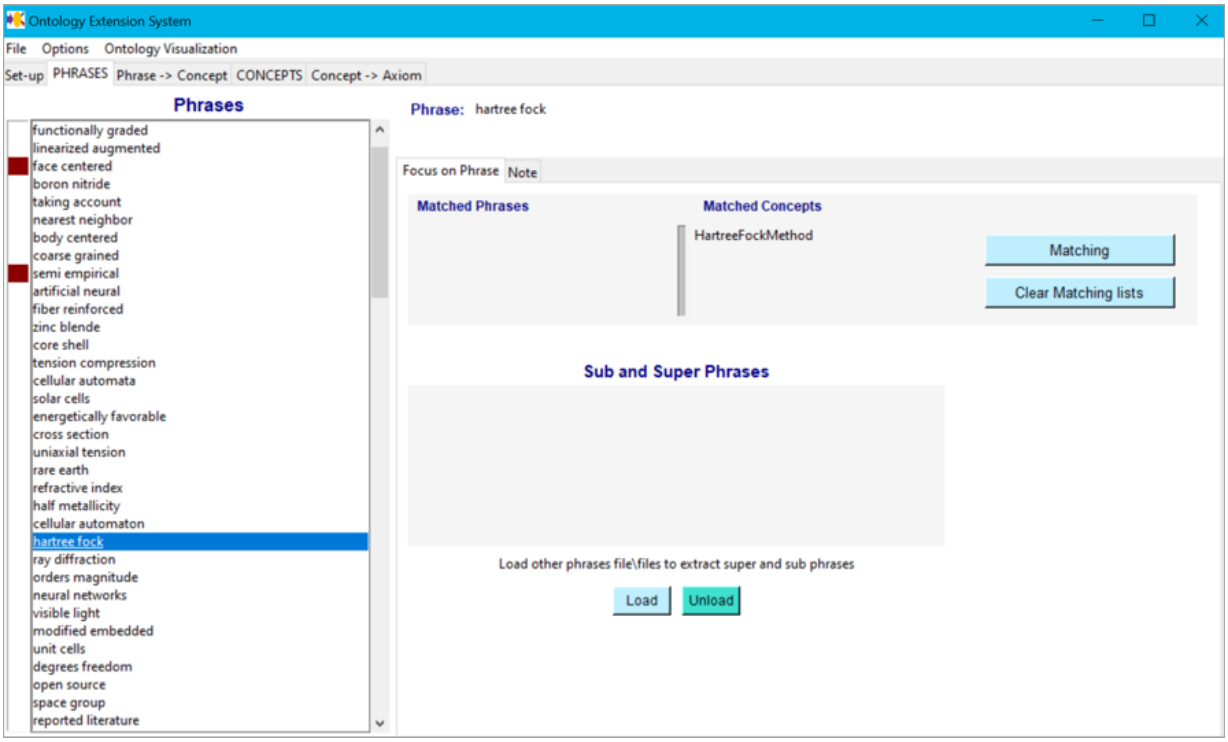


Figure 2. Tool - *Phrases* tab.

The use cases, which were identified through literature study and discussion between the domain expert and the knowledge engineers based on experience with the development of OPTIMADE and the use of materials science databases, are listed below.

– UC1: MDO will be used for representing knowledge in basic materials science such as solid-state physics and condensed matter theory.
– UC2: MDO will be used for representing materials calculation and standardizing the publication of the materials calculation data.
– UC3: MDO will be used as a standard to improve the interoperability among heterogeneous databases in the materials design domain.
– UC4: MDO will be mapped to OPTIMADE's schema to improve OPTIMADE's search functionality.

The competency questions are based on discussions with domain experts and contain questions that the databases currently can answer as well as questions that experts would want to ask the databases. For instance, CQ1, CQ2, CQ6, CQ7, CQ8 and CQ9 cannot be asked explicitly through the database APIs, although the original downloadable data contains the answers.

– CQ1: What are the calculated properties and their values produced by a materials calculation?
– CQ2: What are the input and output structures of a materials calculation?
– CQ3: What is the space group type of a structure?
– CQ4: What is the lattice type of a structure?
– CQ5: What is the chemical formula of a structure?
– CQ6: For a series of materials calculations, what are the compositions of materials with a specific range of a calculated property (e.g., band gap)?
– CQ7: For a specific material and a given range of a calculated property (e.g., band gap), what is the lattice type of the structure?

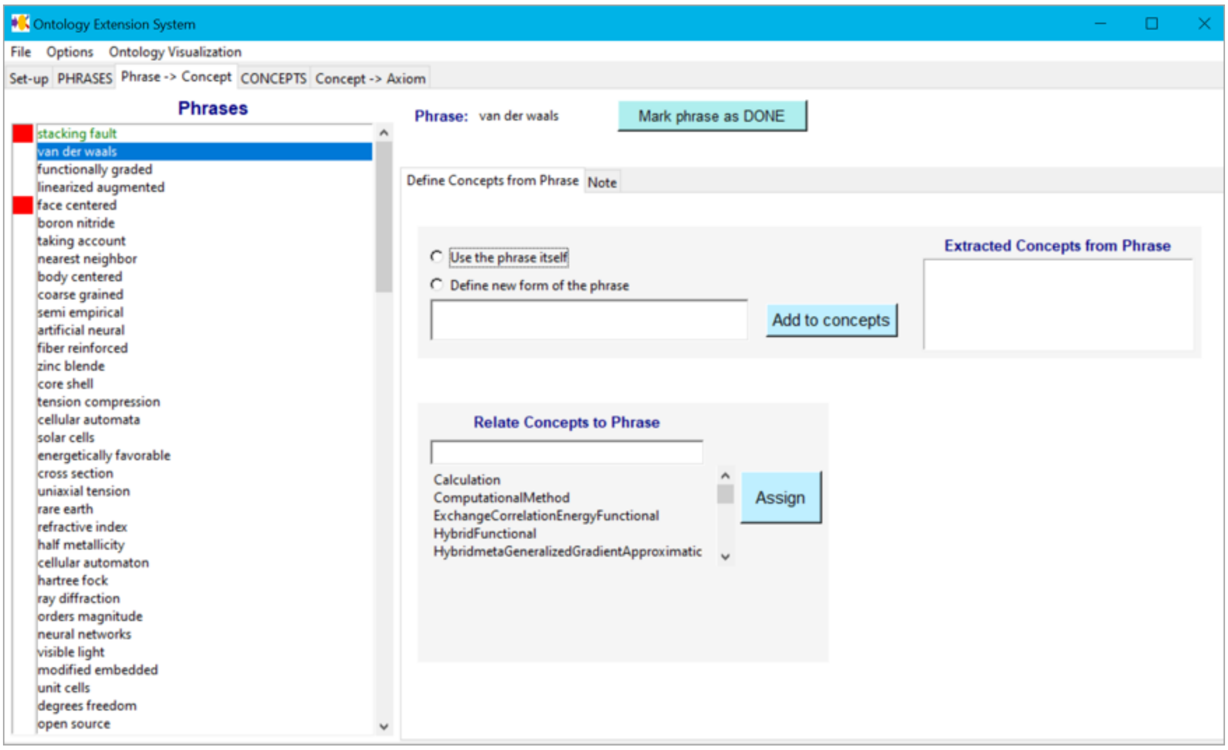


Figure 3. Tool - *From Phrases to Concepts* tab.

- CQ8: For a specific material and an expected lattice type of output structure, what are the values of calculated properties of the calculations?
- CQ9: What is the computational method used in a materials calculation?
- CQ10: What is the value for a specific parameter (e.g., cutoff energy) of the method used for the calculation?
- CQ11: Which software produced the result of a calculation?
- CQ12: Who are the authors of the calculation?
- CQ13: Which software or code does the calculation run with?
- CQ14: When was the calculation data published to the database?

Further, we proposed a list of additional restrictions that help in defining concepts.

- AR1: A materials property can relate to a structure.
- AR2: A materials calculation has exactly one corresponding computational method.
- AR3: A structure corresponds to one specific space group.
- AR4: A calculation is performed by some software programs or codes.
- AR5: A structure is a part of some materials.
- AR6: A calculation is achieved by a specific computational method.
- AR7: A structure and a property can be published by references which could be databases or publications.
- AR8: A calculation can take some structures as input.
- AR9: A calculation can take some properties as input.

3.2. *Reusing and re-engineering non-ontological resources.*

To obtain the knowledge for building the ontology, we followed two steps: (1) the collection and analysis of non-ontological resources that are relevant to the materials design domain, and (2) discussions with the domain expert regarding the concepts and relationships to be modeled in the ontology. The collection of non-ontological resources

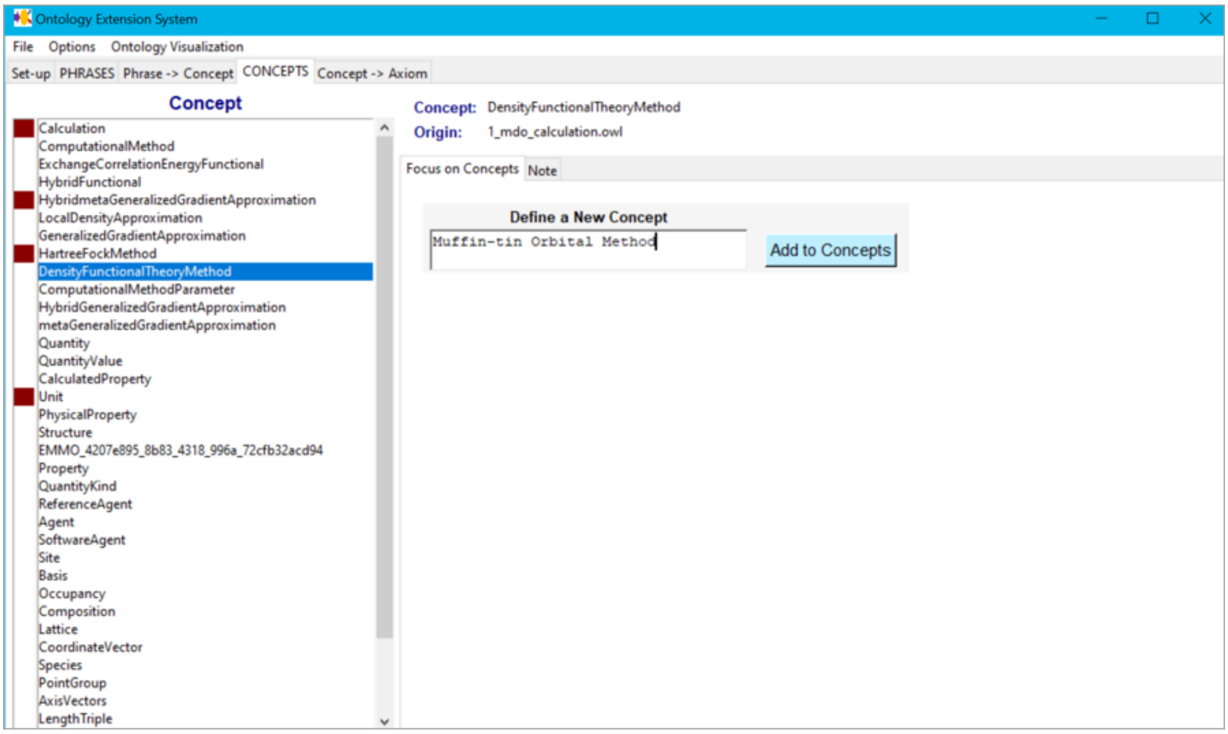


Figure 4. Tool - *Concepts* tab.

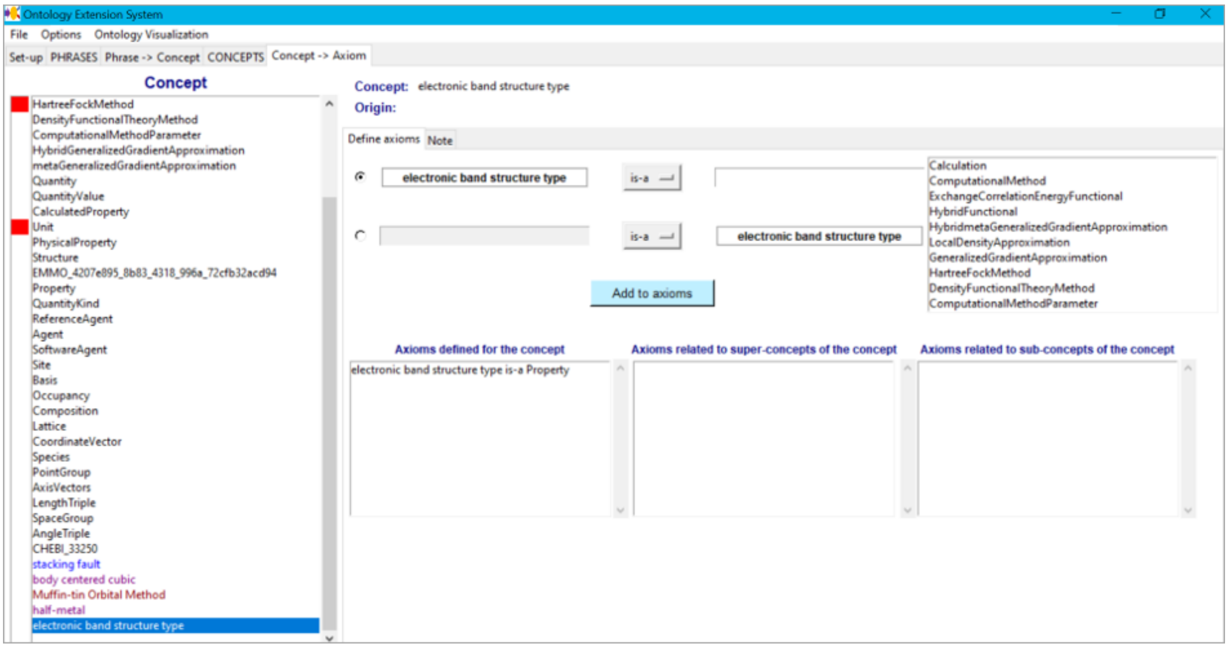


Figure 5. Tool - *From Concepts to Axioms* tab.

comes from: (1) the dictionaries of CIF (https://www.iucr.org/resources/cif) and International Tables for Crystallography (https://it.iucr.org/); (2) the APIs from different databases (e.g., Materials Project, AFLOW, OQMD) and OPTIMADE.

3.3. Modular development aiming at building design patterns.

We identified a pattern related to provenance information in the repository of Ontology Design Patterns (ODPs) that could be reused or re-engineered for MDO. This has led to the reuse of entities in PROV-O [27]. Further, we built MDO in modules.

3.4. Connection and Integration of Existing Ontologies.

MDO is connected to EMMO by reusing the concept 'Material', and to ChEBI by reusing the concept 'atom'. Further, we reuse the concepts 'Agent' and 'SoftwareAgent' from PROV-O. In terms of representation of units we reuse the 'Quantity', 'QuantityValue', 'QuantityKind' and 'Unit' concepts from QUDT (Quantities, Units, Dimensions and Data Types Ontologies) [22]. We use the metadata terms from the Dublin Core Metadata Initiative (DCMI, http://purl.org/dc/terms/) to represent the metadata of MDO.

4. Description of MDO

MDO consists of one basic module, *Core*, and two domain-specific modules, *Structure* and *Calculation*, importing the *Core* module. In addition, the *Provenance* module, which also imports *Core*, models provenance information. In total, the OWL2 DL representation of the ontology contains 37 concepts, 32 object properties, and 32 data properties. Figure 14 shows an overview of the ontology. The ontology specification is also publicly accessible at w3id.org at https://w3id.org/mdo/full/1.0/. The competency questions can be answered using the concepts and relations in the different modules (CQ1 and CQ2 by *Core*, CQ3 to CQ8 by *Structure*, CQ9 and CQ10 by *Calculation*, and CQ11 to CQ14 by *Provenance*).

The **Core** module as shown in Figure 6, consists of the top-level concepts and relations of MDO, which are also reused in other modules. Figure 7 shows the description logic axioms for the *Core* module. The module represents general information of materials calculations. The concepts *Calculation* and *Structure* represent materials calculations and materials' structures, respectively, while *Property* represents materials properties. *Property* is specialized into the disjoint concepts *CalculatedProperty* and *PhysicalProperty* (Core1, Core2, Core3). *Property*, which can be viewed as a quantifiable aspect of one material or materials system, is defined as a sub-concept of *Quantity* from QUDT (Core4). *Properties* are also related to *structures* (Core5). When a calculation is applied on materials structures, each *calculation* takes some *structures* and *properties* as input, and may output *structures* and *calculated properties* (Core6, Core7). Further, we use EMMO's concept *Material* and state that each *structure* is related to some *material* (Core8).

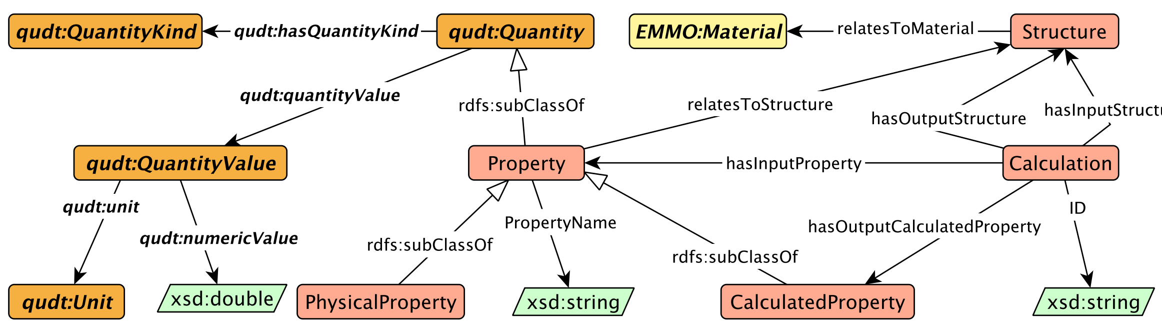


Figure 6. Concepts and relations in the Core module.

$$\textit{(Core1) CalculatedProperty} \sqsubseteq \textit{Property}$$
$$\textit{(Core2) PhysicalProperty} \sqsubseteq \textit{Property}$$
$$\textit{(Core3) CalculatedProperty} \sqcap \textit{PhysicalProperty} \sqsubseteq \bot$$
$$\textit{(Core4) Property} \sqsubseteq \textit{Quantity}$$
$$\textit{(Core5) Property} \sqsubseteq \forall\, \textit{relatesToStructure.Structure}$$
$$\textit{(Core6) Calculation} \sqsubseteq \exists\, \textit{hasInputStructure.Structure} \sqcap \forall\, \textit{hasInputStructure.Structure} \sqcap \forall\, \textit{hasOutputStructure.Structure}$$
$$\textit{(Core7) Calculation} \sqsubseteq \exists\, \textit{hasInputProperty.Property} \sqcap \forall\, \textit{hasInputProperty.Property}$$
$$\sqcap \forall\, \textit{hasOutputCalculatedProperty.CalculatedProperty}$$
$$\textit{(Core8) Structure} \sqsubseteq \exists\, \textit{relatesToMaterial.Material} \sqcap \forall\, \textit{relatesToMaterial.Material}$$

Figure 7. Description logic axioms for the Core module.

The **Structure** module as shown in Figure 8, represents the structural information of materials. Figure 9 shows the description logic axioms for the *Structure* module. Each *structure* has exact one *composition* which represents what chemical elements compose the structure and the ratio of elements in the *structure* (Struc1). The *composition* has different representations of chemical formulas. The *occupancy* of a structure relates the *sites* with the *species*, i.e. the specific chemical elements, that occupy the *site* (Struc2 - Struc5). Each *site* has at most one representation of coordinates in Cartesian format and at most one in fractional format (Struc6, Struc7). The spatial information regarding structures is essential to reflect physical characteristics such as melting point and strength of materials. To represent this spatial information, we state that each *structure* is represented by some *bases* and a (periodic) *structure* can also be represented by one or more *lattices* (Struc8). Each *basis* and each *lattice* can be identified by one *axis-vectors* set or one *length triple* together with one *angle triple* (Struc9, Struc10). An *axis-vectors* set has three connections to *coordinate vector* representing the coordinates of three translation vectors respectively, which are used to represent a (minimal) repeating unit (Struc11). These three translation vectors are often called a, b, and c. Point groups and space groups are used to represent information of the symmetry of a structure. The *space group* represents a symmetry group of patterns in three dimensions of a *structure* and the *point group* represents a group of linear mappings which correspond to the group of motions in space to determine the symmetry of a *structure*. Each *structure* has one corresponding *space group* (Struc12). Based on the definition from International Tables for Crystallography, each *space group* also has some corresponding *point groups* (Struc13).

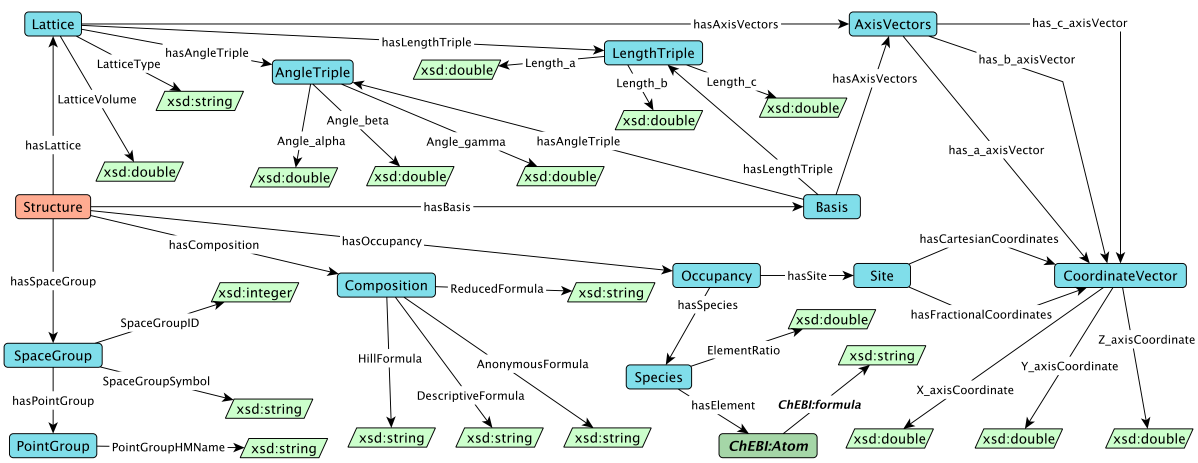


Figure 8. Concepts and relations in the Structure module.

The **Calculation** module as shown in Figure 10, represents the classification of different computational methods. Figure 11 shows the description logic axioms for the *Calculation* module. Each *calculation* is achieved by a specific *computational method* (Cal1). Each *computational method* has some *parameters* (Cal2). In the current version of

(Struc1) $Structure \sqsubseteq =1\ hasComposition.Composition \sqcap \forall\ hasComposition.Composition$
(Struc2) $Structure \sqsubseteq \exists\ hasOccupancy.Occupancy \sqcap \forall\ hasOccupancy.Occupancy$
(Struc3) $Occupancy \sqsubseteq \exists\ hasSpecies.Species \sqcap \forall\ hasSpecies.Species$
(Struc4) $Occupancy \sqsubseteq \exists\ hasSite.Site \sqcap \forall\ hasSite.Site$
(Struc5) $Species \sqsubseteq =1\ hasElement.Atom$
(Struc6) $Site \sqsubseteq \leqslant 1\ hasCartesianCoordinates.CoordinateVector \sqcap \forall\ hasCartesianCoordinates.CoordinateVector$
(Struc7) $Site \sqsubseteq \leqslant 1\ hasFractionalCoordinates.CoordinateVector \sqcap \forall\ hasFractionalCoordinates.CoordinateVector$
(Struc8) $Structure \sqsubseteq \exists\ hasBasis.Basis \sqcap \forall\ hasBasis.Basis \sqcap \forall\ hasLattice.Lattice$
(Struc9) $Basis \sqsubseteq =1\ hasAxisVectors.AxisVectors \sqcup (=1\ hasLengthTriple.LengthTriple \sqcap =1\ hasAngleTriple.AngleTriple)$
(Struc10) $Lattice \sqsubseteq =1\ hasAxisVectors.AxisVectors \sqcup (=1\ hasLengthTriple.LengthTriple \sqcap =1\ hasAngleTriple.AngleTriple)$
(Struc11) $AxisVectors \sqsubseteq =1\ has\_a\_axisVector.CoordinateVector$
$\sqcap =1\ has\_b\_axisVector.CoordinateVector$
$\sqcap =1\ has\_c\_axisVector.CoordinateVector$
(Struc12) $Structure \sqsubseteq =1\ hasSpaceGroup.SpaceGroup \sqcap \forall\ hasSpaceGroup.SpaceGroup$
(Struc13) $SpaceGroup \sqsubseteq \exists\ hasPointGroup.PointGroup \sqcap \forall\ hasPointGroup.PointGroup$

Figure 9. Description logic axioms for the Structure module.

this module, we represent two different methods, the *density functional theory method* and the *HartreeFock method* (Cal3, Cal4). In particular, the density functional theory method is frequently used in materials design to investigate the electronic structure. Such method has at least one corresponding *exchange correlation energy functional* (Cal5) which is used to calculate the exchange-correlation energy of a system. There are different kinds of functionals to calculate exchange–correlation energy (Cal6 - Cal11).

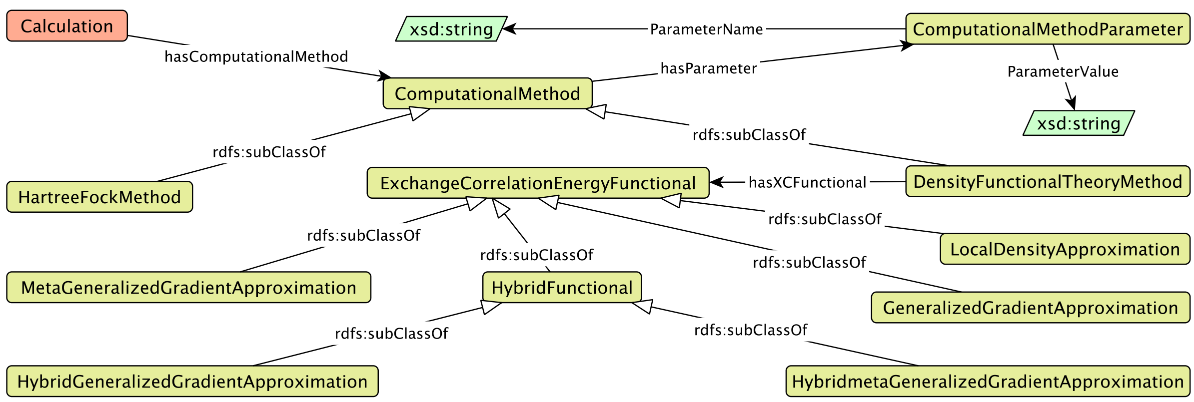


Figure 10. Concepts and relations in the Calculation module.

The **Provenance** module as shown in Figure 12, represents the provenance information of materials data and calculation. Figure 13 shows the description logic axioms for the *Provenance* module. We reuse part of PROV-O and define a new concept *ReferenceAgent* as a sub-concept of PROV-O's agent (Prov1). We state that each *structure* and *property* can be published by *reference agents* which could be databases or publications (Prov2, Prov3). Each *calculation* is produced by a specific *software* (Prov4).

In Figure 15 we exemplify the use of MDO to represent a specific materials calculation and related data in an instantiation. The example is from one of the 85 stable materials published in Materials Project in [19]. The calculation is about one kind of elpasolites, with the composition $Rb_2Li_1Ti_1Cl_6$. To not overcrowd the figure, we only show the instances corresponding to the calculation's output structure, and for multiple calculated properties, species and sites, we only show one instance respectively. Connected to the instances of the Core module's concepts,

$$
\begin{gathered}
(Cal1)\ Calculation \sqsubseteq\ = 1\ hasComputationalMethod.ComputationalMethod \\
(Cal2)\ ComputationalMethod \sqsubseteq \exists\ hasParameter.ComputationalMethodParameter \sqcap \forall \\
hasParameter.ComputationalMethodParameter \\
(Cal3)\ DensityFunctionalTheoryMethod \sqsubseteq ComputationalMethod \\
(Cal4)\ HartreeFockMethod \sqsubseteq ComputationalMethod \\
(Cal5)\ DensityFunctionalTheoryMethod \sqsubseteq \exists\ hasXCFunctional.ExchangeCorrelationEnergyFunctional \\
\sqcap \forall\ hasXCFunctional.ExchangeCorrelationEnergyFunctional \\
(Cal6)\ GeneralizedGradientApproximation \sqsubseteq ExchangeCorrelationEnergyFunctional \\
(Cal7)\ LocalDensityApproximation \sqsubseteq ExchangeCorrelationEnergyFunctional \\
(Cal8)\ metaGeneralizedGradientApproximation \sqsubseteq ExchangeCorrelationEnergyFunctional \\
(Cal9)\ HybridFunctional \sqsubseteq ExchangeCorrelationEnergyFunctional \\
(Cal10)\ HybridGeneralizedGradientApproximation \sqsubseteq HybridFunctional \\
(Cal11)\ HybridmetaGeneralizedGradientApproximation \sqsubseteq HybridFunctional
\end{gathered}
$$

Figure 11. Description logic axioms for the Calculation module.

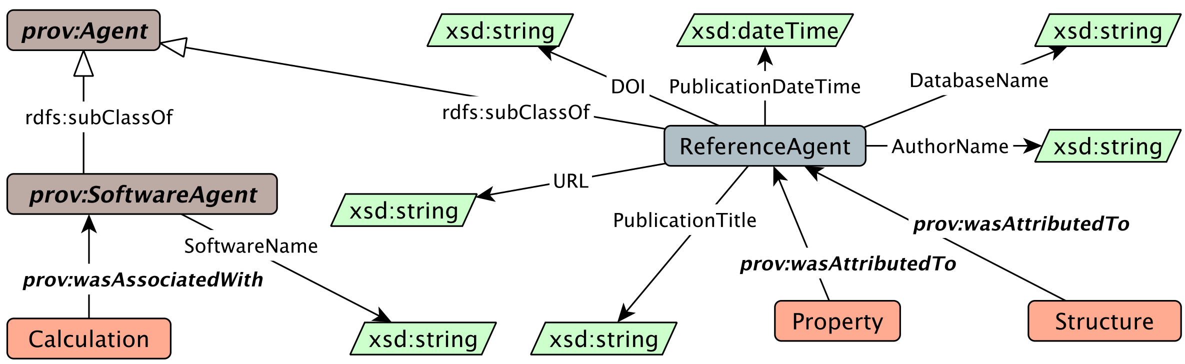


Figure 12. Concepts and relations in the Provenance module.

$$
\begin{gathered}
(Prov1)\ ReferenceAgent \sqsubseteq Agent \\
(Prov2)\ Structure \sqsubseteq \forall\ wasAttributedTo.ReferenceAgent \\
(Prov3)\ Property \sqsubseteq \forall\ wasAttributedTo.ReferenceAgent \\
(Prov4)\ Calculation \sqsubseteq \exists\ wasAssociatedwith.SoftwareAgent
\end{gathered}
$$

Figure 13. Description logic axioms for the Provenance module.

are instances representing the structural information of the output structure, the provenance information of the output structure and calculated property, and the information about the computational method used for the calculation.

5. Extending MDO

In this section we use the approach in [29] to propose new concepts for MDO. The result of this work is a list of proposed concepts that are validated by a domain expert to be relevant to the domain. However, at this point the concepts are not yet included in the public version of MDO. Discussions are ongoing regarding the scope of the extension of MDO with respect to the domain and intended use of MDO.

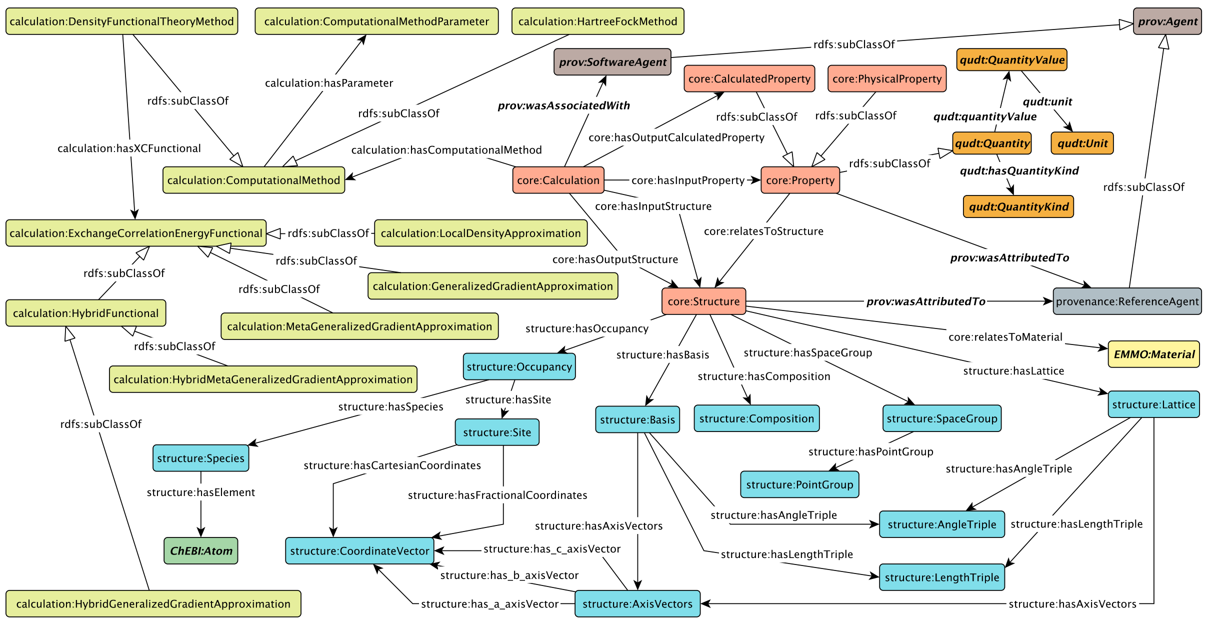


Figure 14. An overview of MDO.

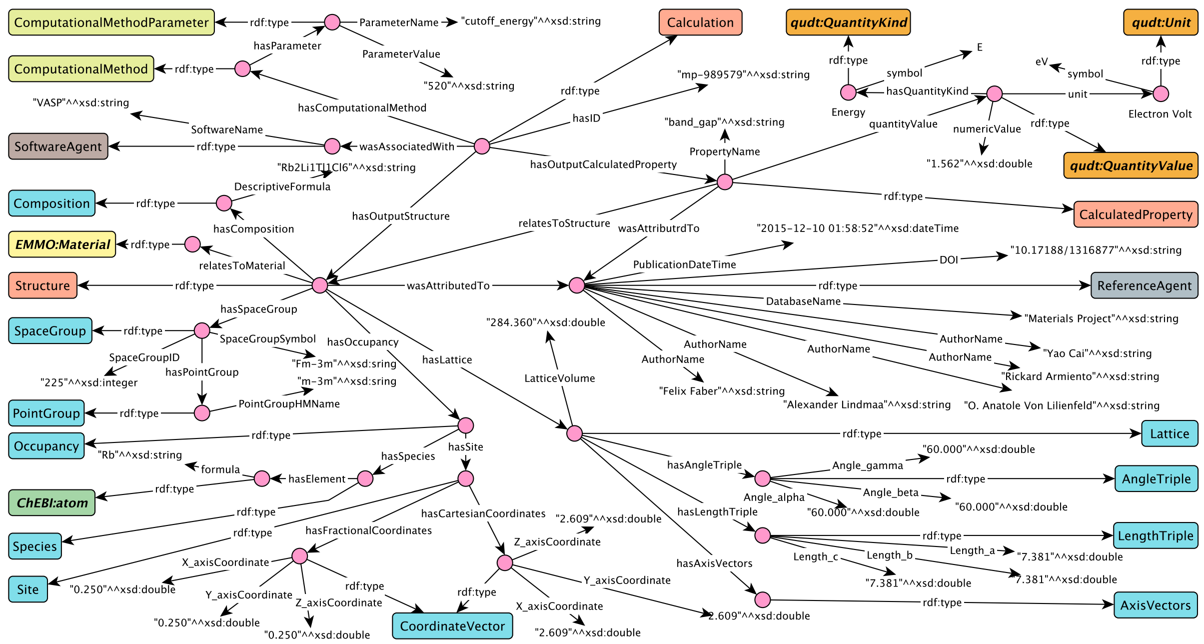


Figure 15. An instantiated materials calculation.

A first step in the approach in [29] is to collect the corpus that is used as input. To be able to find as relevant information for MDO as possible, we used MDO as a seed for querying journal databases. The 37 concepts of MDO were used as search phrases for two journals in the field of materials design, NPJ Computational Materials (https://www.sciencedirect.com/journal/computational-materials-science) and Computational Materials Science

Table 2
The distribution of word frequency after preprocessing.

Frequency	Percentage of words
less than 10	72.27
10-30	13.25
31-100	7.76
101-500	5.25
501-1000	0.83
1001-2000	0.44
2001-3000	0.12
More than 3000	0.08

Table 3
Number of frequent phrases for *min_support* 10, 15, 20, 25 and 30 respectively, and three different versions of the ToPMine algorithm.

min_support	original TopMine	ToPMine without stemming	ToPMine with stemming
10	6,901	6,478	5,452
15	3,826	3,578	3,022
20	2,542	2,402	2,046
25	1,816	1,722	1,477
30	1,375	1,298	1,119

(https://www.nature.com/npjcompumats/), to find relevant articles of which we retrieved the titles and abstracts. The final corpus contained titles and abstracts from 403 articles of NPJ Computational Materials and 8,193 from Computational Materials Science.

In the preprocessing step characters were set to lower case and punctuations were removed. Further, we removed words of length one or two. One consequence is that often materials symbols are removed. An advantage is that the phrases and words are usually not material dependent, but we miss cases where this is interesting.

After preprocessing there were 21,548 distinct words which together occur 808,862 times. An overview of the frequency of the words is presented in Table 2. Most of the words (72.27%) occur less than 10 times, while there are 17 words that occur more than 3000 times. These are 'based', 'properties', 'method', 'calculations', 'phase', 'materials', 'study', 'structure', 'temperature', 'density', 'results', 'energy', 'electronic', 'model', 'molecular', 'simulations', 'surface'.

5.1. Frequent phrases

Given a minimum support threshold *min_support*, we say that phrases that occur at least *min_support* times are *frequent phrases*. ToPMine generates frequent phrases of a length up to a maximum length that is given as an input parameter. In our experiments this was set to 10. Further, ToPMine does not generate all frequent phrases but uses a method based on partitioning documents and using a significance score for deciding which words likely belong together, to produce high-quality frequent phrases [17].

The second column of Table 3 shows the number of frequent phrases that ToPMine generates for different values of *min_support*. The higher the *min_support*, the fewer frequent phrases are generated.

We also define a maximum support threshold *max_support_word*. Words that occur more than *max_support_word* times were removed. These words were usually very general terms that are not interesting for an ontology or that would not be interesting for a domain ontology, but possibly for an upper ontology. We do note, however, that some of these words could be useful such as 'method', 'electronic', 'model', and 'molecular'. The second column in Table 4 shows how *max_support_word* influences the number of generated frequent phrases with a constant *min_support* of 10. The higher *max_support_word*, the more frequent phrases are generated. Note that no word

Table 4

Number of frequent phrases for *min_support* 10 and for *max_support_word* 500, 1000, 3000, 5000, and 8000, respectively for two different versions of the ToPMine algorithm.

max_support_word	ToPMine without stemming	ToPMine with stemming
8,000	6,478	5,452
5,000	5,947	5,023
3,000	4,692	4,090
1,000	1,878	1,692
500	932	866

occurs more than 8000 times in our corpus, so setting *max_support_word* to 8000 allows all words (or, in other words, *max_support_word* is not used).

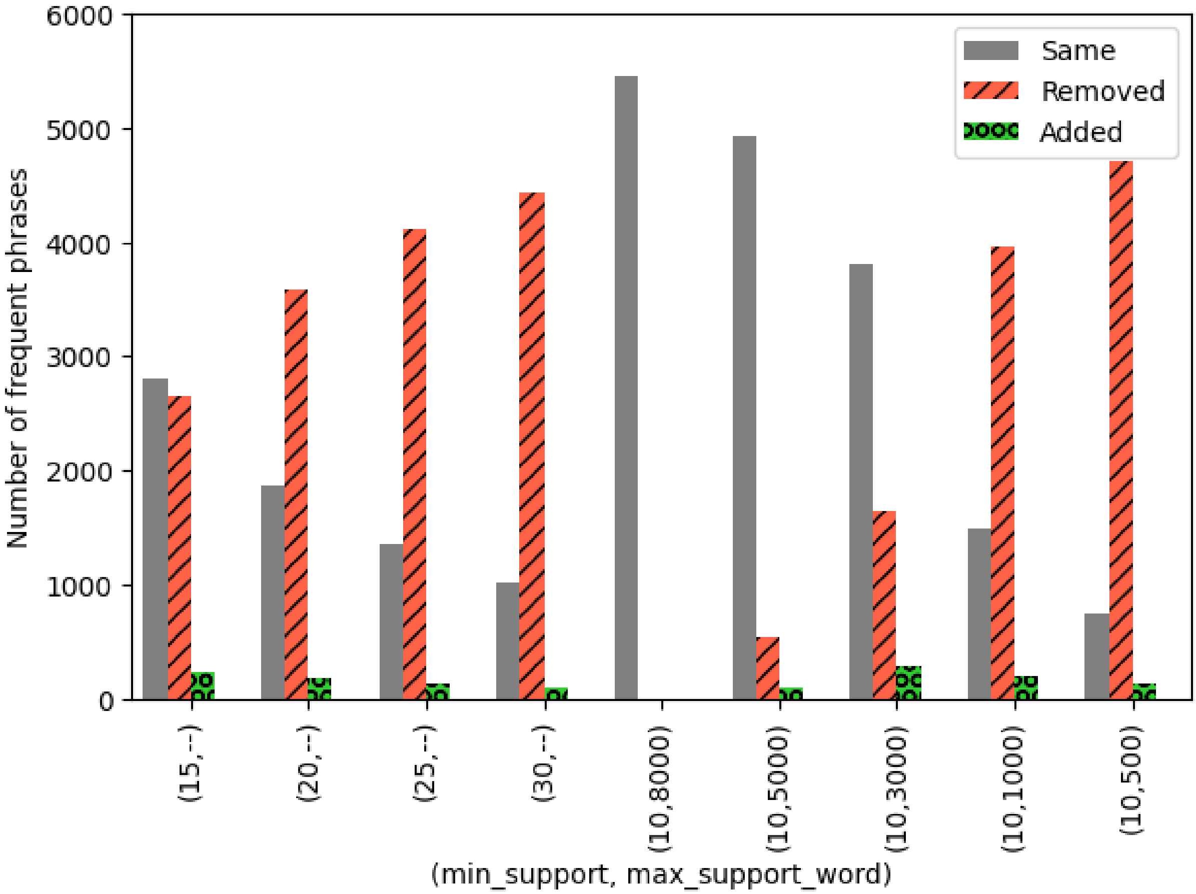


Figure 16. Comparison of the frequent phrases of ToPMine with *min_support* 10 (and *max_support_word* 8000) to settings with *min_support* in 15, 20, 25 and 30, respectively, and settings with *min_support* 10 and *max_support_word* 500, 1000, 3000, 5000, respectively.

Another way to look at the influence of *min_support* and *max_support_word* is to compare how many of the frequent phrases are the same and different for different settings. In Figure 16 we show this comparison of different settings to the base setting where *min_support* is 10 and *max_support_word* is 8000 (i.e., *max_support_word* is not used) which is shown in the middle of the figure. The 'Same' bars show how many generated phrases occur both in the base setting and the compared setting. The 'Removed' bars show how many frequent phrases occur in the base setting, but not in the compared setting. For the cases where we change *min_support*, these would be phrases that are frequent phrases for *min_support* 10, but not for the higher *min_support* in the compared setting. For example 'computational screening' is removed for *min_support* 15. For the cases where we change the *max_support_word*, these would be phrases with words that occur more often than the *max_support_word* in the compared setting. For instance, 'sheet metal forming' contains the word 'metal' with frequency 3457 and would be removed for *max_support_word* 1000. The 'Added' bars show which frequent phrases occur newly in the compared settings. This happens, as stated before, because ToPMine does not generate all frequent phrases, but focuses on high-quality frequent phrases. As an example, 'exchange correlation potential' appears at least 10 times and less than 30 times and 'exchange correlation' appears at least 30 times. Both are frequent phrases for *min_support* 10. However, ToPMine does not generate 'exchange correlation' for *min_support* 10, but it does generate 'exchange correlation potential'. For *min_support* 30 'exchange correlation potential' is not a frequent phrase, while 'exchange correlation' is, and ToPMine does generate 'exchange correlation' as a frequent phrase.

Further, we investigated the influence of using stemming on the frequent phrases. For instance, the phrases 'molecular dynamics simulations', 'molecular dynamics simulation', 'molecular dynamic simulations' and 'molecular dynamic simulation' have the same stem 'molecular dynam simul'. Stemming allows for removing redundant phrases and thus reduces the work of the domain expert. The influence on the number of generated phrases can be seen by comparing the last two columns in Tables 3 and 4. A disadvantage is that in some cases possible concept candidates may be removed. To alleviate this problem we show the domain expert for each of the stemmed frequent phrases the list of corresponding original phrases. This also helps the domain expert to choose terms to be added to the ontology.

In Table 5, we show the candidate concepts based on the validation of a domain expert on the frequent phrases from the experiment with *min_support* 30 and *max_support_word* 500. In total, 88 candidate concepts are suggested based on 81 out of 131 frequent phrases generated by the experiment. Some candidate concepts can be added into MDO as sub-concepts of existing concepts. For instance, 'Linearized Augmented Plane Wave Method' is a sub-concept of 'Density Functional Theory Method'. Some candidate concepts are relevant to the materials design domain but may be not interesting for data access or data integration over materials design databases. For instance, 'Covalent Bond' is a bonding type that can be used to describe materials structures.

5.2. Topics

After the phrase mining we generated topics. The number of topics (*num_topic*) is an input parameter to ToPMine. Each topic contains a set of phrases and these sets do not have to be disjoint. For instance, Figure 17 shows the overlap of phrases between topics for different settings of input parameters. In general, when we increase the number of topics, the number of frequent phrases in each topic decreases and the overlap between topics decreases as well.

The domain expert validates these topics and if possible, labels them to generate concepts for the ontology. In Table 6, we show the domain expert validation on 10 topics generated by ToPMine with stemming, *min_support* 30 and *max_support_word* 500. Among these topics, there are two topics (topics 0 and 9) that are interpreted with multiples labels, i.e., the domain expert divided the topic in different parts. The other topics received one label. Further, representative phrases are given for each topic. The labels and the representative phrases can all lead to new concepts.

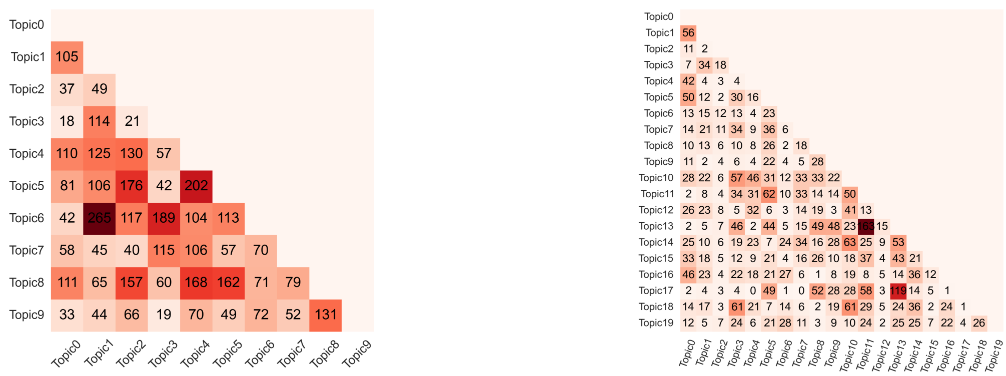
6. Using MDO in ontology-based access to materials databases

In this section we show how MDO can be used for providing semantic and integrated access to materials databases. As a proof of concept we implemented data integration over two data sources, Materials Project [24] and OQMD [36] using a new GraphQL-based framework for data access and integration. This framework is introduced in [31] and illustrated in Figure 18. The framework generates a GraphQL server that provides integrated access to data from heterogeneous data sources. These data sources may be based on different schemas and formats and may be accessed in different ways (e.g., tabular data accessed via SQL queries or JSON-formatted data accessed via a REST API). To address the heterogeneity, the framework relies on an ontology that provides an integrated

Table 5

Candidate concepts based on domain expert validation on the experiment with *min_support* 30 and *max_support_word* 500.

Stacking Fault	Stone-wales Defect	Cement Paste
Van der Waals Force	Covalent Bond	Perdew-Burke-Ernzerhof (PBE) Exchange-Correlation Functional
Functionally Graded Material	Symmetric Tilt Grain Boundary Structure	Fatigue Limit
Linearized Augmented Plane Wave Method	Asymmetric Tilt Grain Boundary Structure	Edurance Limit
Face Centered Cubic	Rock Salt Structure	Porous Media
Boron Nitride	Rock Salt	Microstructural Features
Nearest Neighbor	Projector Augmented Wave Method	Hall-Petch Relation
Body Centered Cubic	Iron	Conduction Band
Coarse Grained Model	Cahn–Hilliard Equation	Slip Plane
Fiber Reinforced	Cauchy-Born Rule	Vapor Deposition
Zinc Blende	Domain Wall	Spinodal Decomposition
Core Shell	Armchair	Spontaneous Polarization
Rare Earth	Zigzag	Absorption Spectrum
Refractive Index	Double Walled Nanotube	Charpy Impact Test
Half metallicity	Power Factor	Alkaline Earth Metal
X-ray diffration	Carbon Nanotube (cnt)	Contact Angle
Modified Embedded Atom Method	Mixed Mode Fracture	Vickers Hardness
Unit Cell	Homo-lumo Energy Gap	Rutile Titanium Dioxide (TiO_2)
Absorption Spectra	Stainless Steels	Kinematic Hardening
Glass Formation	Vibrational Modes	Hexagonal Close Packed (hcp)
Brillouin Zone	Domain Switching	Anomalous Hall Effect
Lennard Jones	Sound Velocity	Valence Band
Dispersion Curves	Anatase (TiO_2)	Voight Model
Cohesive Zone Model	Austenitic Stainless Steel	Reuss Model
Quasi-harmonic Debye Model	Crystallographic Orientation	Solute Segregation
Additive Manufacturing	Brittle Transition	Directional Solidification
Real Space Methods	Ductile Transition	Muffin-tin Orbital method
Quasi-harmonic Model	Brittle-Ductile Transition	Muffin-tin Orbital Approximation
Quantum Dot	Modified Becke-Johnson Exchange-Correlation Functional	
Hexagonal Boron Nitride	Kohn-Sham	



(a) *min_support* 10, *num_topic* 10

(b) *min_support* 10, *num_topic* 20

Figure 17. Number of common phrases between pairs of topics.

view of the data from the different sources, and corresponding semantic mappings that define how the data from the underlying data sources is represented as instances of the ontology (arrows ***(a)***) and ***(b)***). Furthermore, two processes

Table 6

Topic labelling based on domain expert validation on the experiment with *min_support* 30 and *max_support_word* 500 (Up to five representative phrases are selected for each label).

Topic NO.	Topic Labels	Representative Phrases
0	Computational Method Categories	Linearized Augmented Plane Wave Method
		Hartree-Fock Method
		Perdew-Burke-Ernzerhof (PBE) Exchange-Correlation Functional
		Modified Becke-Johnson Exchange Correlation Functional
		Kohn-Sham
	Materials Properties and Features	Absorption Spectrum
		Refractive Index
		Homo-lumo Energy Gap
		Alkaline Earth Metal
		Dispersion curves
	Electronic Structure Features	Conduction Band
		Valence Band
	Materials Categorizations	Half metallicity
		Rare Earth
	Experimental Method Categories	X-ray Diffraction
	Specific Materials	Zinc Blende
	Applications	Optoelectronic Devices
1	Hardness-related Materials Concepts	Quasi-harmonic Debye Model
		Quasi-harmonic Model
		Rock Salt
		Sound Velocity
		Zinc Blende
2	Materials Strength-related Concepts	Stacking Fault
		Van der Waals Force
		Tension Compression
		Uniaxial Tension
		Symmetric Tilt Grain Boundary Structure
3	Materials Fatigue/Fracture-related Concepts	Functionally Graded Material
		Fiber Reinforced
		Cohesive Zone Model
		Unit Cell
		Cement Paste
4	Materials Synthesis Concepts	Additive Manufacturing
		Vapor Deposition
		Directional Solidification
		Microstructural Features
		Crystallographic Orientations
5	Battery-related Materials Concepts	Ion Batteries
		Anatase (TiO_2)
		Lithium Ion Batteries
		Rutile Titanium Dioxide (TiO_2)
		Boron Nitride
6	Materials Structural Categorizations	Face Centered Cubic
		Body Centered Cubic
		Coarse Grained Model
		Hexagonal Close Packed (hcp)
		Iron
7	Nanotube-related Concepts	Armchair
		Boron Nitride
		Hexagonal Boron Nitride
		Carbon Nanotube (cnt)
		Cross Section
8	Artificial Intelligence-Methods (NO)	Artificial Neural
		Neural Networks
		Open Source
		Degrees Freedom
		Artificial Neural Networks
9	Materials Concepts for Solar-cells	Solar Cells
		Quantum Dots
		Domain Wall
		Power Factor
		Electric Fields
	Materials Magnetism Concepts	Domain Switching
		Anomalous Hall Effect
	Materials Polarization Concepts	Spontaneous Polarization

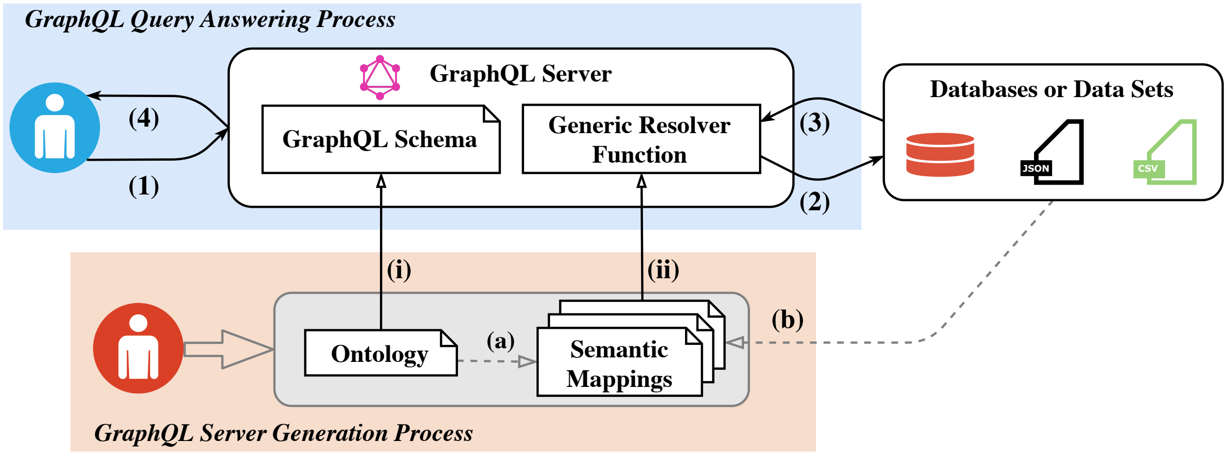


Figure 18. Framework of ontology-based GraphQL server generation (OBG-gen).

are defined. The first process generates the GraphQL server. This includes generating both a GraphQL schema for the API provided by the server (arrow ***(i)***) and a generic resolver function (arrow ***(ii)***). This process does not need to be repeated unless the ontology or the mappings change. After this generation process, the GraphQL server can be set up. The second process deals with query answering and is performed after the GraphQL server is set up. During this process the query is validated against the GraphQL schema (arrow ***(1)***); the underlying data sources are accessed via resolver functions, the retrieved data is combined, and the data is structured according to the schema (arrows ***(2)*** and ***(3)***), and finally the query result is returned (arrow ***(4)***). Details are available in [31].

6.1. An MDO-based data access and integration system

In our proof of concept implementation we use MDO as the ontology to generate the GraphQL server. The GraphQL server contains a GraphQL schema generated based on MDO, and a generic resolver function that allows for accessing underlying data sources and restructuring the obtained data according to the GraphQL schema. This generic resolver function is implemented based on RML [13, 14] semantic mappings defined using MDO terminologies.

In a GraphQL API, the GraphQL schema defines types, their fields, and the value types of the fields. An object type represents a list of fields and each field has a value of a specific type such as object type or scalar type. A scalar is used to represent a value such as a string. An input object type can be used to define an input object with a set of input fields; the input fields are either scalars, or other input objects. A GraphQL schema also supports defining types that represent operations such as query and mutation. The schema presumes *Query* type as the query root operation type. The part of the final GraphQL schema shown in Listing 1 contains two basic object type definitions which are *Calculation* and *Structure*. Both have field definitions which represent the relationships to scalar types or to other object types. For instance, the *Calculation* type has a field definition *ID* of which the value type is *String*, and a field definition *hasOutputStructure* of which the value type is *Structure*. The *Query* type has a field definition which is *CalculationList*. This definition allows users to write a GraphQL query that accesses all the entities of *Calculation* type, as shown in the example in Listing 12. Further, the schema contains four input object type definitions which are *CalculationFilter*, *CalculatedPropertyFilter*, *StringFilter* and *FloatFilter*, for capturing notions of filtering conditions that should be taken into account in the evaluation of the GraphQL query. As an example, the *CalculationFilter* can be used as an argument of the *CalculationList* query. For instance, in the query example of Listing 12, the argument of the query from line 3 to line 10 is used to represent a conjunctive filter expression with the meaning ‘property name is band gap’ and ‘the value of the property is equal to 5’. In our generic implementation of GraphQL resolver functions, a filter expression represented by an input object type will be parsed and evaluated against the underlying data.

Listing 2 shows an example of mappings in RML related to ‘band gap’ which is a *CalculatedProperty*. In general, an RML document has one or more *Triples Maps* which declare how input data is mapped into triples of the form (subject, predicate, object). A *Triples Map* contains the following three components; *Logical Source*, *Subject Map* and a set of *Predicate-Object Maps*. A logical source declares the source of input data to be mapped (e.g., line 2 to line 6). It contains definitions of *source* locating the input data source, *reference formulation* declaring how to refer to the input data, and *logical iterator* declaring the iteration loop used to map the input data. A subject map declares the rule for generating subjects when mapping input data into triples (e.g, line 7 to line 10). A predicate-object map consists of one or more predicate maps declaring how to generate predicates of triples (e.g., line 12), one or more object maps or referencing object maps defining how to generate objects of triples (e.g., line 19 to line 25). An object map can be a reference-valued term map (e.g., line 46 to line 48) or a constant-valued term map (e.g., line 13 to line 15).

In Listing 3, we show an excerpt of the JSON response based on Materials Project for the query to retrieve the data in which the task_id of the calculation is mp-989579.

Listing 1: An excerpt of the GraphQL schema generated based on MDO

```
type Query {
    CalculationList(filter: CalculationFilter): [Calculation]
}

type Calculation {
    hasOutputStructure: [Structure]
    hasOutputCalculatedProperty: [CalculatedProperty]
    ID: String
}

type Structure {
    StructureID: String
    hasComposition: Composition
}

input CalculationFilter {
    hasOutputCalculatedProperty: CalculatedPropertyFilter
    ID: StringFilter
    _and: [CalculationFilter]
    _or: [CalculationFilter]
    _not: CalculationFilter
}

input CalculatedPropertyFilter {
    PropertyName: StringFilter
    numericalValue: FloatFilter
    _and: [CalculatedPropertyFilter]
    _or: [CalculatedPropertyFilter]
    _not: CalculatedPropertyFilter
}

input StringFilter {
    _eq: String
    _gt: String
    _in: [String]
}

input FloatFilter {
    _eq: Float
    _gt: Float
    _in: [Float]
}
```

Listing 2: An excerpt of the RML mappings defined based on MDO.

```
<BandGapPropertyMapping>
 rr:logicalSource [
   rml:source "http://example.com/mp-989579_Rb2LiTlCl6.json";
   rml:referenceFormulation ql:JSONPath;
   rml:iterator "$.data[*]";
 ];
 rr:subjectMap [
   rr:template "http://example.com/mdo/bandgapproperty/{task_id}";
   rr:class core:CalculatedProperty;
 ];
 rr:predicateObjectMap [
   rr:predicate core:hasPropertyName;
   rr:objectMap [
     rr:constant "band_gap";
   ];
 ];
 rr:predicateObjectMap [
   rr:predicate qudt:quantityValue;
   rr:objectMap [
     rr:parentTriplesMap <BandGapQuantityValueMapping>
     rr:joinCondition [
      rr:child "task_id";
      rr:parent "task_id";
     ];
   ];
 ];

<BandGapQuantityValueMapping>
 rr:logicalSource [
   rml:source "http://example.com/mp-989579_Rb2LiTlCl6.json";
   rml:referenceFormulation ql:JSONPath;
   rml:iterator "$.data[*]";
 ];
 rr:subjectMap [
   rr:template "http://example.com/mdo/bandgapquantityvalue/{task_id}";
   rr:class qudt:QuantityValue;
 ];
 rr:predicateObjectMap [
   rr:predicate qudt:unit;
   rr:objectMap [
     rr:constant qudt_unit:EV;
   ];
 ];
 rr:predicateObjectMap [
   rr:predicate qudt:numericalValue;
   rr:objectMap [
     rr:reference "BandGap";
   ];
 ].
```

Listing 3: An excerpt of the JSON response based on Materials Project API.

```
{
  "data": [
    {
      "band_gap": 1.5623,
      "density": 3.474406325286245,
      "elements": ["Rb", "Li", "Tl", "Cl"],
      "final_energy": -31.6397249,
      "full_formula": "Rb2Li1Tl1Cl6",
      "spacegroup": {
        "symprec": 0.1,
        "source": "spglib",
        "symbol": "Fm-3m",
        "number": 225,
        "point_group": "m-3m",
        "crystal_system": "cubic"
    },
    "task_id": "mp-989579",
    "volume": 284.3605319552886
    }
  ]
}
```

All material needed to generate the server is available online at https://github.com/LiUSemWeb/OBG-gen (including, e.g., files with the source code, mappings, the queries and documentation).

6.2. Comparison

We compare our tool, OBG-gen (Ontology-Based GraphQL Server Generation) with three systems: morph-rdb [35], HyperGraphQL [37], and UltraGraphQL [38]. Morph-rdb is a tool that can access a relational database by translating SPARQL queries into SQL queries based on R2RML mappings. HyperGraphQL and its extension UltraGraphQL are GraphQL interfaces to query Linked Data that may be provided by local RDF files and remote SPARQL endpoints.

The semantic mappings (for all the systems) are based on the MDO. OBG-gen generates the GraphQL schema based on MDO. For UltraGraphQL and HyperGraphQL we use a modified version of the generated schema since they require directive definitions, as additional configurations for object type or field definitions, to specify the context information when translating a GraphQL query to SPARQL query (e.g., for an object type in the GraphQL schema, what is the URL of the object type's corresponding class in the RDF data.).

6.2.1. Data

The data from Materials Project and OQMD represents five different types of entities (Calculation, Structure, Composition, Band Gap and Formation Energy). We collected data in the sizes of 1K (i.e., 1000 entries), 2K, 4K, 8K, 16K and 32K from each database for populating the five entity types. We represented this data in different formats, i.e., tabular data for relational databases and for CSV files, and JSON-formatted data for JSON files. Additionally, for the RDF-based systems in our evaluation, we created an RDF file based on RML mappings and MDO for each dataset setting. We used six dataset settings for the experiments, which are 1K-1K, 2K-2K, 4K-4K, 8K-8K, 16K-16K and 32K-32K. Taking 32K-32K as an example, for each entity type, the test data contains the 32K data from Materials Project and the 32K data from OQMD, respectively.

6.2.2. Systems

Morph-rdb is served with data stored in a single database instance containing data from Materials Project and OQMD in separate tables. HyperGraphQL and UltraGraphQL are served with the same RDF data for each dataset setting. We use OBG-gen with two input settings. OBG-gen-rdb is served with two MySQL database instances hosting data from Materials Project and OQMD respectively. Conceptually, OBG-gen-mix is also served with two database instances. However, each instance contains different formats of data such as data in MySQL database, CSV or JSON files.

Table 7
Query Characteristics.

Query	CQ	DI	Filter	Query	CQ	DI	Filter
Q1	CQ5			Q7	CQ5		✓
Q2	CQ2, CQ5	✓		Q8	CQ5	✓	✓
Q3	CQ1	✓		Q9	CQ6, CQ7	✓	✓
Q4	CQ1, CQ2, CQ5	✓		Q10	CQ6, CQ7	✓	✓
Q5	CQ1, CQ2, CQ5			Q11	CQ1,CQ2, CQ5		✓
Q6			✓	Q12	CQ5	✓	✓

6.2.3. Queries

The queries that are used in our experiments are listed in Appendix A. We describe their characteristics in Table 7. The 'CQ' column describes which competency questions from Section 3.1 are covered by the queries. As the selected data covers competency questions CQ1-2 and CQ5-7, these are the ones that are covered. However, the other competency questions would in principle be easily covered with other or extended data sets. The 'DI' column shows which queries are of particular interest in the domain, i.e., these are often used queries to the materials databases. The other queries are mainly used to evaluate system performance on technically difficult queries. The 'Filter' column indicates whether the query contains filters.

As example, query Q9 in Listing 12 requests all the entities of Calculation type of which the value of the band gap property is larger than 5 electron volt. For such calculation entities, the query requests the corresponding values of ID, and reduced formula of the composition of the output structure. Query Q12 in Listing 15 requests all the entities of type Structure which contain the silicon element.

6.2.4. Experiments and measurements

We evaluate the query execution time (QET) of the different systems over the six dataset settings. For each query separately, we run the query four times and always consider the first run as a warm-up, then take the averaged value of the remaining three runs. Figure 19 and Figure 20 illustrate the measurements for all data sizes and all queries. The measures for all data sizes and all queries are available online at https://github.com/LiUSemWeb/OBG-gen/evaluation/README.md. For UltraGraphQL, we have measurements only for queries Q1–Q4 because UltraGraphQL does not support queries with filtering conditions. For HyperGraphQL answering queries with filter expressions, we just have the measurement for Q7 because the system can only deal with simple filter expressions.

6.2.5. Results and discussion

We observe that both GraphQL servers generated by OBG-gen-rdb and OBG-gen-mix can answer all the 12 queries corresponding to competency questions of MDO.

We also observe that increasing data set sizes lead to increasing QETs (cf. Figure 20). For all systems there is a clear increase of QET for queries Q1–Q5 and Q10–Q12, while OBG-gen-rdb and morph-rdb have stable QETs for queries Q6–Q9. This result is what we anticipate. It is reasonable that systems have similar greater increases of QETs for queries of which the result sizes increase as the data set grows and have stable QETs for queries of which the result sizes do not change as the data set grows.

Another observation is that OBG-gen-rdb and morph-rdb have similar QETs for queries Q6–Q12 (cf. Figure 19). We explain it by the fact that filtering conditions are written within *WHERE* clauses in SQL queries that will be evaluated against the back-end databases in both OBG-gen-rdb and morph-rdb. In general, in the scenario of data integration, a system has to rely on the filtering functionalities provided by the underlying data sources for answering queries with filter expressions.

Additionally, morph-rdb outperforms the other systems in terms of QET for queries Q1–Q5 (cf. Figure 19). These queries are mainly regarding relationship traversal between different entity types, without filter expressions.

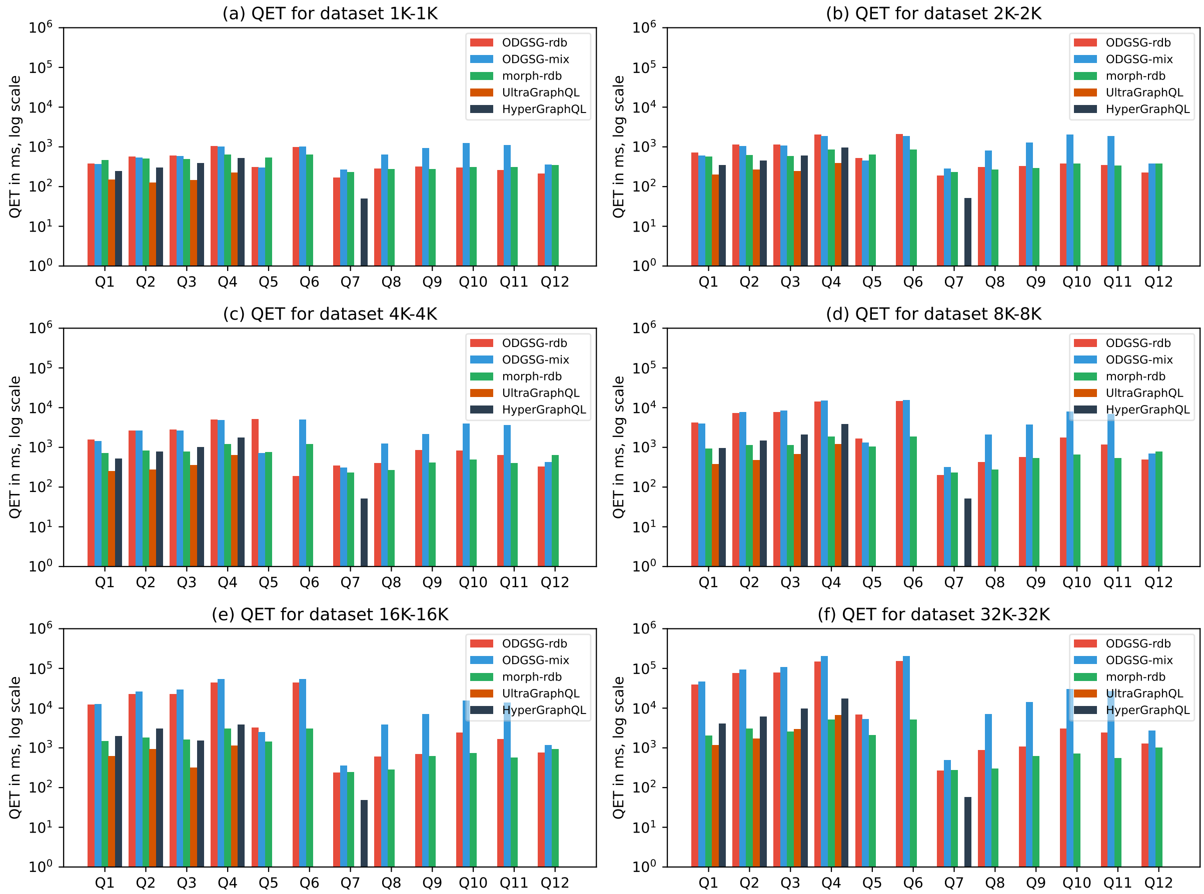


Figure 19. Query Execution Time (QET) per data size on materials dataset.

To answer these queries, the servers generated by OBG-gen first need to access multiple sources to get and combine the data for the same entity type, then structure the data of different entity types based on the relationships stated in referencing-object map definitions in semantic mappings. For UltraGraphQL and HyperGraphQL, the underlying data is materialized RDF data. For morph-rdb, the tables for storing data of different entity types are in the same database instance. In our experiments, combining and structuring retrieved data from heterogeneous underlying data sources is time consuming.

7. Conclusion

In this work we addressed the data access and interoperability issue for computational materials databases by developing MDO and providing a proof-of-concept implementation of an MDO-based data access and integration system for computational materials databases. We have described MDO and a possible extension and showed that the proof-of-concept implementation can answer all competency questions for MDO, while not all of these could be answered by using the underlying databases' APIs.

One direction of future work is to extend the current proof-of-concept implementation in different ways. We want to integrate more databases as well as the OPTIMADE API. Further, as many end users in this domain may be more comfortable with form-based user interfaces, we will look into providing a form-based user interface or one that aids users to pose queries.

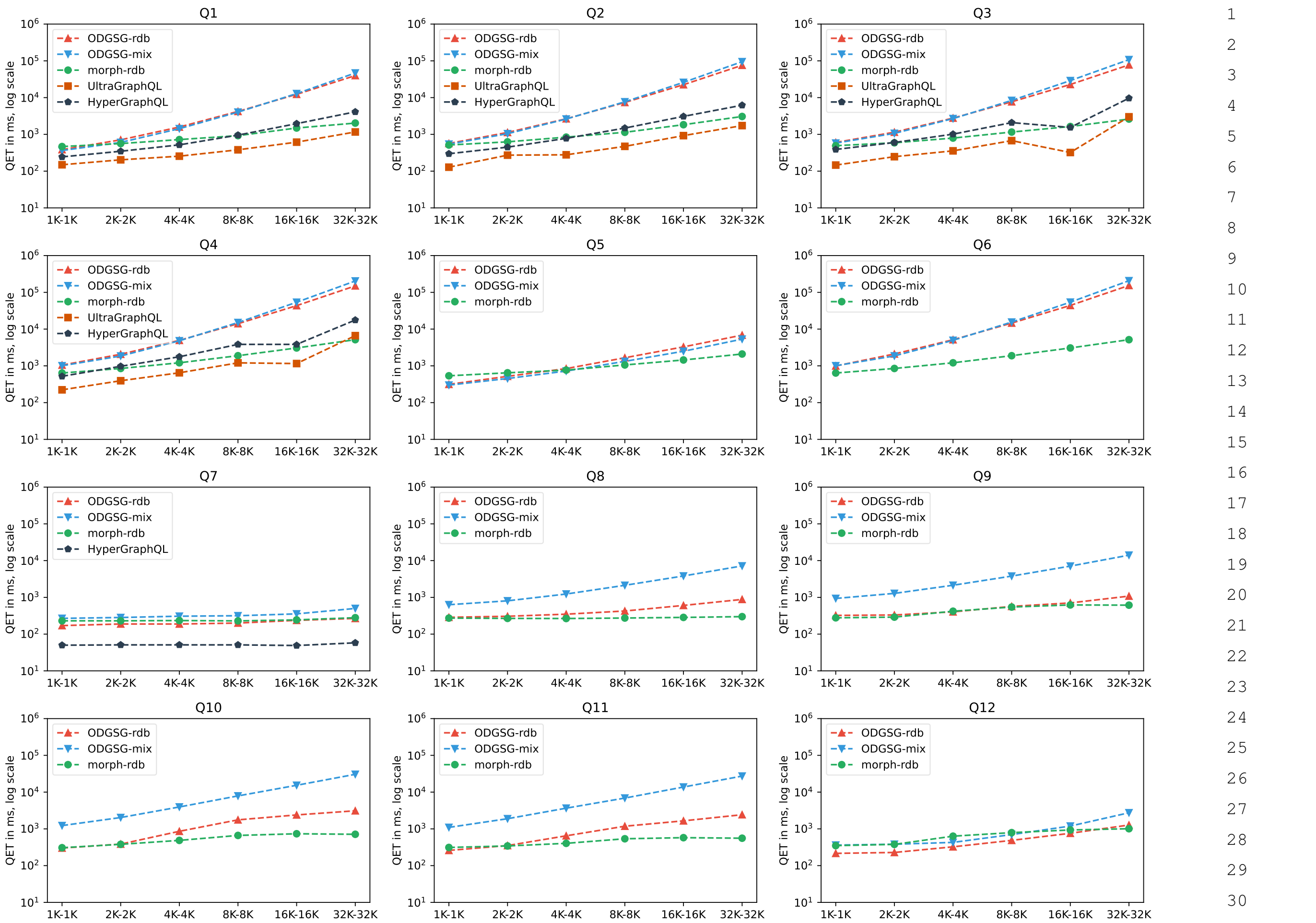


Figure 20. Query Execution Time (QET) per query on materials dataset.

After discussion with domain experts we will extend the public version of MDO with the concepts and relations they deem appropriate. We will also use MDO in related domains as in a newly started project on interoperability of simulation systems.

We will also investigate the connection of MDO with other ontologies, and in particular upper level ontologies. This is, for instance, one of the topics of a recently accepted OntoCommons (https://ontocommons.eu/) demonstrator that we will lead.

Acknowledgements. This work has been financially supported by the Swedish e-Science Research Centre (SeRC), the Swedish National Graduate School in Computer Science (CUGS), the Swedish Research Council (Vetenskapsrådet, dnr 2018-04147), and the Swedish Agency for Economic and Regional and Growth (Tillväxtverket).

References

[1] Abd Nikooie Pour, M., Li, H., Armiento, R., Lambrix, P.: A First Step towards Extending the Materials Design Ontology. In: Chiacchiera, S., Horsch, M.T., Francisco Morgado, J., Goldbeck, G. (eds.) ESWC Workshop on Domain Ontologies for Research Data Management in Industry Commons of Materials and Manufacturing. pp. 1–11 (2021)

[2] Abd Nikooie Pour, M., Li, H., Armiento, R., Lambrix, P.: A first step towards a tool for extending ontologies. In: Lambrix, P., Pesquita, C., Wiens, V. (eds.) Proceedings of the Sixth International Workshop on the Visualization and Interaction for Ontologies and Linked Data co-located with the 20th International Semantic Web Conference (ISWC 2021), Virtual Conference, 2021. CEUR Workshop Proceedings, vol. 3023, pp. 1–12. CEUR-WS.org (2021), http://ceur-ws.org/Vol-3023/paper2.pdf

[3] Agrawal, A., Choudhary, A.: Perspective: materials informatics and big data: realization of the Fourth paradigm of science in materials science. APL Materials **4**, 053208:1–10 (2016).

[4] Alam, M., Birkholz, H., Dessì, D., Eberl, C., Fliegl, H., Gumbsch, P., von Hartrott, P., Mädler, L., Niebel, M., Sack, H., Thomas, A.: Ontology modelling for materials science experiments. In: Joint Proceedings of the Semantics co-located events: Poster & Demo track and Workshop on Ontology-Driven Conceptual Modelling of Digital Twins co-located with Semantics 2021. CEUR Workshop Proceedings, vol. 2941. CEUR-WS.org (2021), http://ceur-ws.org/Vol-2941/paper11.pdf

[5] Andersen, C.W., Armiento, R., Blokhin, E., Conduit, G.J., Dwaraknath, S., Evans, M.L., Fekete, Á., Gopakumar, A., Gražulis, S., Merkys, A., Mohamed, F., Oses, C., Pizzi, G., Rignanese, G.M., Scheidgen, M., Talirz, L., Toher, C., Winston, D., Aversa, R., Choudhary, K., Colinet, P., Curtarolo, S., Di Stefano, D., Draxl, C., Er, S., Esters, M., Fornari, M., Giantomassi, M., Govoni, M., Hautier, G., Hegde, V., Horton, M.K., Huck, P., Huhs, G., Hummelshøj, J., Kariryaa, A., Kozinsky, B., Kumbhar, S., Liu, M., Marzari, N., Morris, A.J., Mostofi, A.A., Persson, K.A., Petretto, G., Purcell, T., Ricci, F., Rose, F., Scheffler, M., Speckhard, D., Uhrin, M., Vaitkus, A., Villars, P., Waroquiers, D., Wolverton, C., Wu, M., Yang, X.: OPTIMADE, an API for exchanging materials data. Scientific Data **8**, 217 (2021).

[6] Armiento, R.: Database-driven high-throughput calculations and machine learning models for materials design. In: Schütt, K.T., Chmiela, S., von Lilienfeld, O.A., Tkatchenko, A., Tsuda, K., Müller, K.R. (eds.) Machine Learning Meets Quantum Physics, pp. 377–395. Springer International Publishing, Cham (2020).

[7] Ashino, T.: Materials Ontology: An Infrastructure for Exchanging Materials Information and Knowledge. Data Science Journal **9**, 54–61 (2010), doi: 10.2481/dsj.008-041

[8] Ceder, G., Persson, K.: How Supercomputers Will Yield a Golden Age of Materials Science. Scientific American **309** (2013)

[9] Cheung, K., Drennan, J., Hunter, J.: Towards an Ontology for Data-driven Discovery of New Materials. In: Semantic Scientific Knowledge Integration AAAI/SSS Workshop. pp. 9–14 (2008)

[10] Curtarolo, S., Hart, G., Buongiorno-Nardelli, M., Mingo, N., Sanvito, S., Levy, O.: The high-throughput highway to computational materials design. Nature Materials **12**, 191–201 (2013).

[11] Curtarolo, S., Setyawan, W., Wang, S., Xue, J., Yang, K., Taylor, R., Nelson, L., Hart, G., Sanvito, S., Buongiorno-Nardelli, M., Mingo, N., Levy, O.: AFLOWLIB.ORG: A distributed materials properties repository from high-throughput ab initio calculations. Computational Materials Science **58**(Supplement C), 227–235 (2012).

[12] de Matos, P., Dekker, A., Ennis, M., Hastings, J., Haug, K., Turner, S., Steinbeck, C.: ChEBI: a chemistry ontology and database. Journal of cheminformatics **2**(S1), P6:1 (2010), doi: 10.1186/1758-2946-2-S1-P6

[13] Dimou, A., Vander Sande, M., Colpaert, P., Verborgh, R., Mannens, E., Van de Walle, R.: RML: A Generic Language for Integrated RDF Mappings of Heterogeneous Data. In: Proceedings of the Workshop on Linked Data on the Web co-located with the 23rd International World Wide Web Conference (WWW 2014). vol. 1184 (2014), http://ceur-ws.org/Vol-1184/ldow2014_paper_01.pdf

[14] Dimou, A., Vander Sande, M., Slepicka, J., Szekely, P., Mannens, E., Knoblock, C., Van de Walle, R.: Mapping Hierarchical Sources into RDF Using the RML Mapping Language. In: 2014 IEEE International Conference on Semantic Computing. pp. 151–158 (2014).

[15] Dragisic, Z., Lambrix, P., Blomqvist, E.: Integrating ontology debugging and matching into the extreme design methodology. In: Blomqvist, E., Hitzler, P., Krisnadhi, A., Narock, T., Solanki, M. (eds.) Proceedings of the 6th Workshop on Ontology and Semantic Web Patterns. CEUR Workshop Proceedings, vol. 1461 (2015), http://ceur-ws.org/Vol-1461/WOP2015_paper_1.pdf

[16] Draxl, C., Scheffler, M.: NOMAD: The FAIR concept for big data-driven materials science. MRS Bulletin **43**(9), 676–682 (2018), doi: 10.1557/mrs.2018.208

[17] El-Kishky, A., Song, Y., Wang, C., Voss, C.R., Han, J.: Scalable Topical Phrase Mining from Text Corpora. Proceedings of the VLDB Endowment **8**(3), 305–316 (2014).

[18] European Committee for Standardization: A guide to the development and use of standards compliant data formats for engineering materials test data (2010)

[19] Faber, F.A., Lindmaa, A., Von Lilienfeld, O.A., Armiento, R.: Machine learning energies of 2 million elpasolite (a b c 2 d 6) crystals. Physical review letters **117**(13), 135502 (2016).

[20] Gangemi, A., Guarino, N., Masolo, C., Oltramari, A., Schneider, L.: Sweetening ontologies with DOLCE. In: Gómez-Pérez, A., Benjamins, V.R. (eds.) Knowledge Engineering and Knowledge Management. Ontologies and the Semantic Web, 13th International Conference, EKAW 2002, Siguenza, Spain, October 1-4, 2002, Proceedings. Lecture Notes in Computer Science, vol. 2473, pp. 166–181. Springer (2002).

[21] Gaultois, M.W., Oliynyk, A.O., Mar, A., Sparks, T.D., Mulholland, G.J., Meredig, B.: Perspective: Web-based machine learning models for real-time screening of thermoelectric materials properties. APL Materials **4**(5), 053213 (May 2016).

[22] Haas, R., Keller, P.J., Hodges, J., Spivak, J.: Quantities, units, dimensions and data types ontologies (QUDT). http://qudt.org, accessed: 2022-02-010

[23] Hastings, J., Jeliazkova, N., Owen, G., Tsiliki, G., Munteanu, C.R., Steinbeck, C., Willighagen, E.: eNanoMapper: harnessing ontologies to enable data integration for nanomaterial risk assessment. Journal of Biomedical Semantics **6**, 10:1–15 (2015), doi: 10.1186/s13326-015-0005-5

[24] Jain, A., Ong, S.P., Hautier, G., Chen, W., Richards, W.D., Dacek, S., Cholia, S., Gunter, D., Skinner, D., Ceder, G., Persson, K.A.: Commentary: The Materials Project: A materials genome approach to accelerating materials innovation. APL Materials **1**(1), 011002 (Jul 2013).

[25] Lambrix, P., Ivanova, V.: A unified approach for debugging is-a structure and mappings in networked taxonomies. Journal of Biomedical Semantics **4**, 10 (2013).

[26] Le Piane, F., Baldoni, M., Gaspari, M., Mercuri, F.: Introducing mambo: Materials and molecules basic ontology. In: Proceedings of the Workshop on Domain Ontologies for Research Data Management in Industry Commons of Materials and Manufacturing (DORIC-MM 2021) co-located with the 18th European Semantic Web Conference (ESWC 2021). pp. 28–39 (2021), http://purl.org/net/epubs/work/50300311

[27] Lebo, T., Sahoo, S., McGuinness, D., Belhajjame, K., Cheney, J., Corsar, D., Garijo, D., Soiland-Reyes, S., Zednik, S., Zhao, J.: PROV-O: The PROV Ontology. W3C recommendation (2013)

[28] Lejaeghere, K., Bihlmayer, G., Björkman, T., Blaha, P., Blügel, S., Blum, V., Caliste, D., Castelli, I.E., Clark, S.J., Corso, A.D., Gironcoli, S.d., Deutsch, T., Dewhurst, J.K., Marco, I.D., Draxl, C., Dulak, M., Eriksson, O., Flores-Livas, J.A., Garrity, K.F., Genovese, L., Giannozzi, P., Giantomassi, M., Goedecker, S., Gonze, X., Grånäs, O., Gross, E.K.U., Gulans, A., Gygi, F., Hamann, D.R., Hasnip, P.J., Holzwarth, N.a.W., Iuşan, D., Jochym, D.B., Jollet, F., Jones, D., Kresse, G., Koepernik, K., Küçükbenli, E., Kvashnin, Y.O., Locht, I.L.M., Lubeck, S., Marsman, M., Marzari, N., Nitzsche, U., Nordström, L., Ozaki, T., Paulatto, L., Pickard, C.J., Poelmans, W., Probert, M.I.J., Refson, K., Richter, M., Rignanese, G.M., Saha, S., Scheffler, M., Schlipf, M., Schwarz, K., Sharma, S., Tavazza, F., Thunström, P., Tkatchenko, A., Torrent, M., Vanderbilt, D., van Setten, M.J., Speybroeck, V.V., Wills, J.M., Yates, J.R., Zhang, G.X., Cottenier, S.: Reproducibility in density functional theory calculations of solids. Science **351**(6280), aad3000 (Mar 2016).

[29] Li, H., Armiento, R., Lambrix, P.: A Method for Extending Ontologies with Application to the Materials Science Domain. Data Science Journal **18**(1) (2019).

[30] Li, H., Armiento, R., Lambrix, P.: An Ontology for the Materials Design Domain. In: Pan, J.Z., Tamma, V.A.M., d'Amato, C., Janowicz, K., Fu, B., Polleres, A., Seneviratne, O., Kagal, L. (eds.) The Semantic Web - ISWC 2020 - 19th International Semantic Web Conference, Athens, Greece, November 2-6, 2020, Proceedings, Part II. Lecture Notes in Computer Science, vol. 12507, pp. 212–227. Springer (2020).

[31] Li, H., Hartig, O., Armiento, R., Lambrix, P.: Ontology-Based GraphQL Server Generation for Data Access and Integration. Submitted

[32] Mulholland, G.J., Paradiso, S.P.: Perspective: Materials informatics across the product lifecycle: Selection, manufacturing, and certification. APL Materials **4**(5), 053207 (Apr 2016).

[33] Poveda-Villalón, M., Gómez-Pérez, A., Suárez-Figueroa, M.C.: OOPS! (OntOlogy Pitfall Scanner!): An On-line Tool for Ontology Evaluation. International Journal on Semantic Web and Information Systems (IJSWIS) **10**(2), 7–34 (2014)

[34] Presutti, V., Blomqvist, E., Daga, E., Gangemi, A.: Pattern-based ontology design. In: Suárez-Figueroa, M.C., Gómez-Pérez, A., Motta, E., Gangemi, A. (eds.) Ontology Engineering in a Networked World, pp. 35–64. Springer (2012).

[35] Priyatna, F.: morph-rdb, version 3.12.5. https://github.com/oeg-upm/morph-rdb/releases/tag/v3.12.5, accessed: 2021

[36] Saal, J.E., Kirklin, S., Aykol, M., Meredig, B., Wolverton, C.: Materials Design and Discovery with High-Throughput Density Functional Theory: The Open Quantum Materials Database (OQMD). JOM **65**, 1501–1509 (2013).

[37] Semantic Integration Ltd.: Hypergraphql, version 2.0.0. https://github.com/hypergraphql/hypergraphql/releases/tag/2.0.0, accessed: 2021

[38] Semantic Integration Ltd.: Ultragraphql, version 1.0.0. https://git.rwth-aachen.de/i5/ultragraphql, accessed: 2021

[39] Smith, B.: On classifying material entities in basic formal ontology. In: Proceedings of the Third Interdisciplinary Ontology Meeting. pp. 1–13 (2012)

[40] Suárez-Figueroa, M.C., Gómez-Pérez, A., Fernández-López, M.: The NeOn methodology for ontology engineering. In: Suárez-Figueroa, M.C., Gómez-Pérez, A., Motta, E., Gangemi, A. (eds.) Ontology engineering in a networked world, pp. 9–34. Springer (2012).

[41] Thomas, D.G., Pappu, R.V., Baker, N.A.: Nanoparticle ontology for cancer nanotechnology research. Journal of Biomedical Informatics **44**(1), 59–74 (2011), doi: 10.1016/j.jbi.2010.03.001

[42] Vardeman II, C., Krisnadhi, A., Cheatham, M., Janowicz, K., Ferguson, H., Hitzler, P., Buccellato, A.: An ontology design pattern and its use case for modeling material transformation. Semantic Web **8**(5), 719–731 (2017).

[43] Zainul Ihsan, A., Dessì, D., Alam, M., Sack, H., Sandfeld, S.: Steps towards a dislocation ontology for crystalline materials. In: Proceedings of the Second International Workshop on Semantic Digital Twins co-located with the 18th Extended Semantic Web Conference (ESWC 2021). CEUR Workshop Proceedings, vol. 2887. CEUR-WS.org (2021), http://ceur-ws.org/Vol-2887/paper4.pdf

[44] Zhang, X., Hu, C., Li, H.: Semantic query on materials data based on mapping matml to an owl ontology. Data Science Journal **8**, 1–17 (2009).

[45] Zhang, X., Pan, D., Zhao, C., Li, K.: MMOY: Towards deriving a metallic materials ontology from Yago. Advanced Engineering Informatics **30**, 687–702 (2016).

Appendix A. GraphQL queries

A.1. Queries without filter expressions

A.1.1. Query 1: List all the structures containing the reduced formula of each structure's composition.

Listing 4: Q1.

```
{
  StructureList{
    hasComposition{
      ReducedFormula
    }
  }
}
```

A.1.2. Query 2: List all the calculations containing the reduced formula of each output structure's composition.

Listing 5: Q2.

```
{
  CalculationList{
    hasOutputStructure{
      hasComposition{
        ReducedFormula
      }
    }
  }
}
```

A.1.3. Query 3: List all the calculations containing the name and value of each output calculated property.

Listing 6: Q3.

```
{
  CalculationList{
    hasOutputCalculatedProperty{
      PropertyName
      numericalValue
    }
  }
}
```

A.1.4. Query 4: List all the calculations containing the name and value of each output calculated property, the reduced formula of each output structure's composition.

Listing 7: Q4.

```
{
  CalculationList{
    hasOutputStructure{
```

```
        hasComposition{
          ReducedFormula
        }
      }
      hasOutputCalculatedProperty{
        PropertyName
        numericalValue
      }
    }
  }
```

A.1.5. Query 5: List all the calculations and structures.

Listing 8: Q5.

```
{
  ThingList{
    ... on Calculation{iri}
    ... on Structure{iri}
  }
}
```

A.2. Queries with filter expressions

A.2.1. Query 6: List all the calculations where the ID is in a given list of values.

Listing 9: Q6.

```
{
  CalculationList(
    filter: {
      _and: [
        {
          ID: {
            _in: ["6332", "8088", "21331", "mp-561628", "mp-614918"]
          }
        }
        {
          hasOutputStructure: {
            hasComposition: {
              ReducedFormula: {
                _in: ["MnCl2", "YClO"]
              }
            }
          }
        }
      ]
    }
  )
  {
    ID
    hasOutputCalculatedProperty {
```

```
        PropertyName
        numericalValue
      }
    }
  }
```

A.2.2. Query 7: List all the calculations where the ID is in a given list of values, and the reduced formula is in a given list of values.

Listing 10: Q7.

```
{
  CalculationList(
    filter: {
      _and: [
        {
          ID: {
           _in: ["6332", "8088", "21331", "mp-561628", "mp-614918"]
          }
        }
        {
          hasOutputStructure: {
            hasComposition: {
              ReducedFormula: {
                _in: ["MnCl2", "YClO"]
              }
            }
          }
        }
      ]
    }
  )
  {
    ID
    hasOutputCalculatedProperty {
      PropertyName
      numericalValue
    }
  }
}
```

A.2.3. Query 8: List all the calculations where the ID is in a given list of values, and the reduced formula is in a given list A or B.

Listing 11: Q8.

```
{
  CalculationList(
    filter: {
      _and: [
        {
          ID: {
            _in: ["6332", "8088", "21331", "mp-561628", "mp-614918"]
```

```
          }
        }
        {
          _or: [
            {
              hasOutputStructure: {
                hasComposition: {
                  ReducedFormula: { _in: ["MnCl2", "YClO"] }
                }
              }
            }
            {
              hasOutputStructure: {
                hasComposition: {
                  ReducedFormula: { _in: ["CeCrS2O", "SiO2", "O"] }
                }
              }
            }
          ]
        }
      ]
    }
  )
  {
    ID
    hasOutputCalculatedProperty {
      PropertyName
      numericalValue
    }
  }
}
```

A.2.4. Query 9: List all the calculations where the value of band gap property is higher than 5.

Listing 12: Q9.

```
{
  CalculationList(
    filter: {
      hasOutputCalculatedProperty: {
        _and: [
          { PropertyName: { _eq: "Band Gap" } }
          { numericalValue: { _gt: 5 } }
        ]
      }
    }
  )
  {
    ID
    hasOutputStructure {
      hasComposition {
        ReducedFormula
      }
    }
```

```
    }
  }
```

A.2.5. Query 10: List all the calculations where the value of band gap property is higher than 5, and the reduced formula in a given list of values.

Listing 13: Q10.

```
{
  CalculationList(
    filter: {
      _and: [
        {
          hasOutputStructure: {
            hasComposition: {
              ReducedFormula: { _in: ["MnCl2", "YClO"] }
            }
          }
        }
        {
          hasOutputCalculatedProperty: {
            _and: [
              { PropertyName: { _eq: "Band Gap" } }
              { numericalValue: { _gt: 5 } }
            ]
          }
        }
      ]
    }
  )
  {
    ID
    hasOutputStructure {
      hasComposition {
        ReducedFormula
      }
    }
    hasOutputCalculatedProperty {
      PropertyName
      numericalValue
    }
  }
}
```

A.2.6. Query 11: List all the calculations where the filter condition is complex that needs to be simplified.

Listing 14: Q11.

```
{
  CalculationList(
    filter: {
      _and: [
        {
```

```
            hasOutputCalculatedProperty: {
              _and: [
                { PropertyName: { _eq: "Band Gap" } }
                { numericalValue: { _gt: 4 } }
              ]
            }
          }
          {
            _or: [
              {
                hasOutputCalculatedProperty: {
                  _and: [
                    { PropertyName: { _eq: "Band Gap" } }
                    { numericalValue: { _gt: 4 } }
                  ]
                }
              }
              {
                hasOutputStructure: {
                  hasComposition: {
                    ReducedFormula: { _in: ["YClO", "CsCl"] }
                  }
                }
              }
            ]
          }
        ]
      }
    )
    {
      ID
    }
  }
```

A.2.7. *Query 12: List all the structures that contain Silicon element.*

Listing 15: Q12.

```
{
  StructureList(
    filter: {
      hasComposition: {
        ReducedFormula: { _like: "%Si%" }
      }
    }
  )
  {
    hasComposition {
      ReducedFormula
    }
  }
}
```