An Ontology and Semantic Web Service for Quantum Chemistry Calculations

Nenad Krdzavac,[†,‡] Sebastian Mosbach,[†,‡,¶] Daniel Nurkowski,[¶] Philipp Buerger,[†] Jethro Akroyd,[†,‡,¶] Jacob Martin,[†,‡] Angiras Menon,[†,‡] and Markus Kraft[*,†,‡,§,¶]

†*Department of Chemical Engineering and Biotechnology, University of Cambridge, Philippa Fawcett Drive, West Site, CB3 0AS Cambridge, UK*

‡*Cambridge Centre for Advanced Research and Education in Singapore (CARES), CREATE Tower, 1 Create Way, Singapore 138602*

¶*CMCL Innovations, Sheraton House, Castle Park, Castle Street, Cambridge, Cambridgeshire, CB3 0AX, United Kingdom*

§*School of Chemical and Biomedical Engineering, Nanyang Technological University, 62 Nanyang Drive, Singapore 637459*

E-mail: mk306@cam.ac.uk

Phone: +44 (0)1223 762784

Abstract

The purpose of this article is to present an ontology, termed OntoCompChem, for quantum chemistry calculations as performed by the Gaussian quantum chemistry software, as well as a semantic web service named MolHub. The OntoCompChem ontology has been developed based on the semantics of concepts specified in the CompChem convention of Chemical Markup Language (CML) and by extending the Gainesville Core (GNVC) ontology. MolHub is developed in order to establish semantic interoperability between different tools used in quantum chemistry and thermochemistry calculations, and as such is integrated into the J-Park Simulator (JPS) – a multi-domain interactive simulation platform and expert system. It uses the OntoCompChem ontology and implements a formal language based on propositional logic as a part of its query engine, which verifies satisfiability through reasoning. This paper also presents a NASA polynomial use-case scenario to demonstrate semantic interoperability between Gaussian and a tool for thermodynamic data calculations within MolHub.

Introduction

In recent years, quantum chemistry calculations accomplished by software tools such as Gaussian[1] have become more and more popular. The results of these calculations have been used, for example, for the derivation of molecular properties, in particular, in the absence of experimental data. Several large databases containing results of quantum chemistry calculations currently exist for a variety of purposes. These include, for example, the Computational Chemistry Comparison and Benchmark DataBase (CCCBDB) for thermochemical properties of species from NIST[2], the Alexandria library of molecular properties for force field development from van der Spoel et al[3], and the ANI-1 database containing non-equilibrium calculations on organic molecules meant as a training set for machine learning techniques[4]. Several other databases containing benchmark quantum calculations at different levels of theories can also be found in the literature, such as from Ramakrishnan et al[5], Simmie et al[6], Nakata and Shimazaki[7], and Head-Gordon and Hait[8] to name but a few. However, to our knowledge these databases do not provide advanced search engine capabilities based on ontologies and formal logic such as propositional logic in order to share these data between different computational tools applied in quantum chemistry.

Semantic Web[9] technologies have been used to improve collaboration between different disciplines. One recent example is the J-Park Simulator (JPS) project in which models from different domains, for example chemical engineering and electrical engineering, have been connected to study the impact of chemical plants on the electrical network in an industrial park[10,11]. We propose that Semantic Web technologies are also good candidates to be used in building web services that should make quantum chemistry calculations easily available to other software tools for further processing. Web Ontology Languages (OWL)[12] are suitable for modeling knowledge about quantum chemistry calculations. There is a need to automate chemical model development and to make these models easily available to other domains by bridging the semantic interoperability gap between tools for chemical big data analysis, advanced mathematical modeling, and real-time simulations. Semantic Web technologies and standards can solve these problems.

To resolve the semantic interoperability gap and to avoid the loss of expensive data, a number of other attempts have been made to alleviate this situation. An early effort to store quantum chemistry calculations was the introduction of the CompChem convention using Chemical Markup Language (CML)[13] which is based on eXtensible Markup Language (XML) standards[14]. It can be used to store complex chemical objects and validate the data entries. This was achieved through a web-based software called MolHub[15]. However, this platform did not establish semantic interoperability between different chemistry software tools. It also did not guarantee the satisfiability (consistency) of data and did not provide a query engine.

More recently several other initiatives to store, validate (by an inference engine), and query quantum chemistry calculations have been published. The closest effort to the work presented in this paper is the Chemical Semantics Framework (CSF)[16]. It uses ontologies to store quantum chemistry calculations[16–18]. The backbone of CSF is the Gainesville Core (GNVC) ontology[19]. This ontology defines theoretical models and types related to the specification of the quantum computation. For example, atom type and positions, numerical approximation parameters, and the corresponding results of the computation such as molecule geometry[16]. Furthermore, it allows users to query such data by using faceted search. The CSF offers clients cloud-like archiving of computational chemistry data. To do this, CSF uses the SPARQL endpoint Virtuoso[20]. It also uses inference to create new results based on existing data that are globally available. The CSF uses GAMESS[21] output files, converts them into JavaScript Object Notation for Linked Data (JSON-LD) format by using XML Stylesheet Language Transformation (XSLT)[18]. The JSON-LD files are converted into Turtle (TTL) files and later published on a web portal. These data are components of the Giant Global Graph (GGG). The CSF allows computer agents to access these data[18]. Although the CSF is a great step forward, there are still areas which can be improved. For example, the faceted search uses only a restricted number of features stored in the knowledge graph and does not guarantee the satisfiability of the query. In addition, the GNVC ontology does not support all features issued by the CompChem convention of CML that are also important for quantum chemistry calculations such as geometry type and spin multiplicity. The CSF project also does not offer a solution on how to establish semantic interoperability between software tools. And further, our analysis showed that the GNVC ontology (v0.7)[19] did not pass all evaluation tests and revealed some modeling errors.

The *e*Science[22] project aims to integrate scientific standards and tools in current working practices. The chemical data used in this project conform to existing Chemical Markup Language (CML)[14]. A meta-data schema is used to validate experimental data. The project tries to bridge the gap between lack of data standards in computational chemistry, and the usage of such data in heterogeneous applications. To reach that goal, OWL is used to express and model the logical structure of these data. It allows different frameworks to work together with data that conform to the data models. The *e*CCP framework[23], as a part of *e*Science project, works with domain specific ontologies that are used by *e*CCP tools. These ontologies have implemented terms such as *molecule*, *crystal*, *basis set*, *molecular orbital*, but it is not clear whether these ontologies have implemented all CML concepts as the authors were unable to download any *e*Science ontology from the web.

Finally, MongoChem[24] is a web application that offers a set of interfaces to integrate computational and experimental chemistry data, and includes semantic querying and reasoning. The web application has tools for interactive visualization and analysis of computational chemistry data stored in JavaScript Object Notation (JSON) format. This format is used in order to support computational chemistry calculations, and to encapsulate data from a computational chemistry software. The MongoChem server has three main components such as data organization and dissemination, user management and authentication, authorization management. The MongoChem client is a user interface that offers clients to display molecular structures, and select vibrational modes by using 3D charts. Whilst the MongoChem web application makes use of semantic web standards such as semantic querying and reasoning, it does not support semantic interoperability between different quantum chemistry computational tools.

To address some of the research challenges mentioned above, the purpose of this paper is:

- To describe a consistent OntoCompChem knowledge base for quantum chemistry calculations that extend the GNVC ontology and matches the CompChem convention of CML.

- To outline the implementation details of MolHub as a semantic web framework mainly designed for semantic interoperability between tools in computational chemistry.

- To derive thermodata in the form of NASA polynomials for other domains.
- To take steps toward semantic interoperability between different tools in the quantum chemistry domain by integrating propositional logic tools with semantic web tools as a component of the J-Park Simulator (JPS).

We will begin this work with an overview of the J-Park simulator motivating this work. The OntoCompChem knowledge base is then described along with an evaluation of GNVC ontology. We then describe the implementation of MolHub as a semantic web service. Finally we will show a case study of computing the NASA polynomials using the framework developed. The MolHub web service is available at `http://www.theworldavatar.com/molhub/`.

J-Park Simulator

The overall purpose of the JPS is to establish a consistent approach that allows the management of data from different domains and the semantic interoperability of different components within cyber-physical systems. JPS also provides a methodology to include mathematical models into a knowledge graph.[25]

The JPS utilizes ontologies that define concepts and relations of different domains that have been designed in a modular manner. For example, OntoCAPE, a large-scale ontology for chemical process engineering[26] is the first ontology that was integrated into the JPS. OntoCAPE has been extended to OntoEIP,[27] which describes eco-industrial parks and their networks. A dissimilar domain ontology, OntoCityGML, which is an ontological version of CityGML – a standardised format for 3D models of cities and landscapes – has also been integrated into the JPS.[28] Concepts of chemical kinetics are defined in OntoKin,[29] i.e. chemical mechanisms consisting of a set of chemical reactions occurring between species. A species is composed of elements (for example, water is composed of hydrogen and oxygen) and may exist in different phases (for example, water may exist in a gas phase or may be adsorbed on some surface phase). OntoEngine contains concepts that are needed to describe the operation of an internal combustion engine. The specification of the fuel used in the engine including the corresponding model of its combustion chemistry is defined in OntoKin format.

Since the JPS knowledge graph is based on the principles of linked data, the knowledge graph can be distributed on different hosts and distributed over the web. This design allows for independently controlled access privileges to different parts of the knowledge graph. The structure of the knowledge graph and the data stored therein is not static but evolves with time, such as during the operation of a chemical plant. The JPS realises this dynamic behavior using agents. Agents are software components working together and covering a broad range of functionality. The description of an agent is also part of the knowledge graph, which is specified in the agent ontology OntoAgent. For example, agents normally operate on specific parts of the knowledge graph and can also execute optimization algorithms. As a consequence the knowledge graph varies in time, both in terms of content and structure. In[10] we presented a cross domain air pollution scenario that employs a combustion simulation agent that in turn uses NASA polynomials to calculate heat capacities and other thermodynamic data. In this paper, we expand the JPS using and extending the work on computational chemistry mentioned in the introduction.

OntoCompChem Knowledge Base

This section illustrates how to encode the semantics of the CompChem convention of CML[13,15] in a OntoCompChem knowledge base. The main goal of this is to utilize semantic web open source tools in the implementation of a revised and extended MolHub framework. It will allow the integration of MolHub web services with other semantic web services. As a starting point, we evaluated the GNVC ontology[16,19].

An Evaluation of the GNVC Ontology

The categorisation of criteria for the GNVC ontology evaluation are given in Table 1. Full definition of given metrics can be found in table 1 of reference[30].

Table 1: Categorisation of the GNVC evaluation criteria (updated from[30]).

Evaluation perspective	Metrics
Ontology correctness	Accuracy Conciseness Completeness Consistency
Ontology quality	Computation efficiency

The GNVC ontology has implemented 502 classes, 173 object properties, and 86 data properties. It also has implemented 18092 axioms. The evaluation of the GNVC ontology is carried out by using Protégé[31] ontology editor. *Computational efficiency* test shows that description logic (DL) expressiveness of the GNVC ontology is equivalent to $\mathcal{SROIN}(\mathcal{D})$[32] DL. This means that HermiT[33,34] is an inference engine that is able to classify this ontology. The consistency of the GNVC ontology is checked programatically.

The GNVC ontology does not pass the *Accuracy* test. One of the errors detected is: *Illegal re-declarations of entities: reuse of entity.* For instance, the term *vibration count*, relevant for our work, is implemented as a type of *owl:DatatypeProperty* and *owl:ObjectProperty* by using the same URI (`http://purl.org/gc/hasVibrationCount`). There are more than ten violations of this nature.

Conciseness requires to test whether an ontology defines irrelevant elements with regards to the domain to be covered.[30] By comparing terms of the CompChem convention with terms implemented in the GNVC ontology, we discovered that there are terms that are redundant in the OntoCompChem knowledge base. For example, the GNVC ontology imports a Unit[35] ontology containing terms such as *currency, new Taiwan dollar, Pula*. These terms should not be part of a computational chemistry ontology.

We tested the *consistency* of the GNVC ontology by using the HermiT reasoner, including testing the consistency of all ontologies imported into the GNVC ontology. The reasoner detected that there are inconsistencies, for example one caused by OWL class `http://data.nasa.gov/qudt/owl/qudt#Dimension`. Instances of this class have more than one value in the range of the OWL property `http://data.nasa.gov/qudt/owl/qudt#literal`.

According to[30], *completeness* measures whether the domain of interest is appropriately covered. The GNVC ontology covers only a sub-domain of quantum chemistry calculations as some features defined in the CompChem are not part of the GNVC. For example, the term *spin multiplicity* can be found in CompChem[13,15] but not in GNVC.

The next section gives an overview of the OntoCompChem knowledge base that is developed by making use of some part of the GNVC ontology (version 0.7)[16,19].

OntoCompChem Knowledge Base: An extension of the GNVC ontology

To express the CompChem convention of CML in the OntoCompChem knowledge base we use DL[36] syntax. The DL syntax offers a more elegant way to express OntoCompChem knowledge base. To bring the OntoCompChem knowledge base alive, we use OWL[37]. Table 2 specifies a list of concepts supported by the CompChem convention of CML features, and their representation as concept inclusion (CI) and role inclusion (RI) axioms in the OntoCompChem knowledge base. Further, Tables 3, and 4 show domain and range restrictions on the new role names implemented in the OntoCompChem knowledge base. In the rest of this paper the term *gc* that appears in front of concept or role names denotes the namespace for the base URI in the GNVC ontology, while the term *ontocompchem* denotes the namespace for the base URI in the OntoCompChem knowledge base. In the following we explain important terms that are part of OntoCompChem knowledge base.

Table 2: Selected CI and RI axioms in the OntoCompChem knowledge base.

Term name	Description logic CI and RI axioms	Description
Model calculation	ontocompchem:ComputationModule $\sqsubseteq \top$	Iteration process of optimization and computation.
Model initialization	ontocompchem:InitializationModule $\sqsubseteq$ ontocompchem:ComputationModule	Initial parameters used in a calculation.
Geometry type	ontocompchem:GeometryType $\sqsubseteq$ gc:MolecularComputation	Derived from molecule geometry.
Rotational constants	ontocompchem:RotationalConstants $\sqsubseteq$ gc:MolecularComputation	Determine the magnitude of spacing between rotational energy levels.
Rotational symmetry	ontocompchem:RotationalSymmetry $\sqsubseteq$ gc:MolecularComputation	Distinct configuration of a molecular system.
Gaussian (g09) calculation	ontocompchem:G09 $\sqsubseteq$ gc:Gaussian-n	Name of a software.
Rotational constant count	ontocompchem:hasRotationalConstantsCount $\sqsubseteq$ ontocompchem:hasCount	The number of rotational constants.
Vibration count	gc:hasVibrationCount $\sqsubseteq$ ontocompchem:hasCount	The number of cycling oscillation about equilibrium point.
Run date	ontocompchem:hasRunDate $\sqsubseteq$ dc:date	Date when Gaussian program is used.
Level of theory	ontocompchem:LevelOfTheory $\sqsubseteq$ gc:MethodologyFeature	A set of approximations used to describe the chemical system.

Model calculation represents an iteration process of optimization and computation. For example, the MolHub representation of a Gaussian input file which forms an instance of the OntoCompChem knowledge base, specifies a geometry optimization for a molecule that consists of several iterations. The concept *model calculation* must satisfy a certain set of rules. For example, it can not contain more than one *molecule* child[13]. The OntoCompChem knowledge base does not implement rules applied in the validation of generated XML files against the CompChem convention of CML[14]. In the OntoCompChem knowledge base, the term *model calculation* is encoded as concept name *ontocompchem:ComputationModule*.

Model initialization in the CompChem convention of CML represents initial parameters used in a calculation (see Table 2). These parameters are grouped in parameter lists. *Model initialization* includes parameters such as *basis set*, and *level of theory*. Also, *model initialization* contains information about the molecule to be studied. In the OntoCompChem knowledge base, *model initialization* is implemented as concept name *ontocompchem:InitializationModule*, and it is subsumed by *ontocompchem:ComputationModule* concept name.

Geometry type classifies chemical species into three categories: atomic, linear and nonlinear[14]. The *geometry type* can be derived from a molecule's geometry or rather from molecule's rotational constants. For atomic species all rotational constants are zero. For linear molecules there are two rotational constants equal to each other, and one rotational constant equal to zero. For nonlinear molecules there are three non-zero rotational constants. Table 2 shows the encoding of *geometry type* as a concept name *ontocompchem:GeometryType*, while domain and range restrictions on the role name *ontocompchem:hasGeometryType* are given in Table 4.

The terms *rotational symmetry*, *geometry type*, *frequency*, *rotational constants*, *vibration*, and *spin multiplicity* belong to the parameter list of *model finalization*[13]. We implemented these terms as concept names (see Table 2), role names, domain and range restrictions on these role names (see Table 3, and Table 4). Although the term *atomic mass* is implemented in CompChem as an attribute in the parameter list of *model fi-*

nalization[13], we chose to import this term from the GNVC ontology into the OntoCompChem knowledge base.

Rotational constants are inversely proportional to moments of inertia and determine the magnitude of spacing between rotational energy levels. The rotational constants can be calculated from the molecule's geometry and atoms. In the OntoCompChem knowledge base, the concept name *ontocompchem:RotationalConstants* is subsumed by *gc:MolecularComputation* (see Table 2) and it is the domain restriction on role name *ontocompchem:hasRotationalConstants* (see Table 3).

Rotational symmetry is number of distinct configuration of a molecular system. The symmetry number or symmetry order of an object is the number of ways of achieving a given spatial orientation using only rotation and without breaking bonds. In statistical thermodynamics, the symmetry number corrects for any over counting of equivalent molecular conformations in the partition function. As an example, the Rigid Rotor Harmonic Oscillator (RRHO) approximation treats an object, in this case a molecule, as a rigid body which does not allow certain symmetry operations like reflections or inversion. These forbidden operations need to be subtracted from the point group order to obtain a correct symmetry number of an object that is subject to constraints. In Table 2, *rotational symmetry* is encoded as *ontocompchem:RotationalSymmetry* concept name that is subsumed by concept name *gc:MolecularComputation.* Table 3 shows the encoding of *rotational symmetry* as *ontocompchem:hasRotationalSymmetryNumber* role name, and provides the CI axioms as domain and range restrictions on this role name.

Frequency is defined as the number of occurrences of a repeating event per unit of time. In the OntoCompChem knowledge base, concept name *gc:Frequency* implements the *frequency* term. This concept name is imported from the GNVC ontology. Also the concept name *gc:Frequency* is the domain restriction on the role name *ontocompchem:hasFrequencies*, as shown in Table 4. We created role name *ontocom-*

pchem:hasVibrationCount that describes the number of vibrational frequencies (see Table 2).

Vibration is a cycling movement/oscillation about an equilibrium point. In quantum mechanics, vibrational analysis is used to decompose complex internal motion of groups of atoms in a molecule into patterns of motion (normal modes) where all parts of the system move sinusoidally with the same frequency and with a fixed phase relation. Vibrational analysis also identifies the nature of a stationary point on the molecular potential energy surface. The implementation of this term in the OntoCompChem knowledge base is imported from the GNVC ontology.

Term *spin multiplicity* is defined as the number of possible orientations (calculated as '2S+1') of the spin angular momentum corresponding to a given total spin quantum number (S), for the same spatial electronic wave-function. In the OntoCompChem knowledge base, *spin multiplicity* is implemented as the role name *ontocompchem:hasSpinMultiplicity.* As shown in Table 3, the domain restriction on this role name is the *gc:Molecule* concept name that is imported from the GNVC ontology. The range restriction on this role name is a number that represents the *spin multiplicity* value.

The name of the computational chemistry program being run, the version of that program, and the run date, are encoded in the OntoCompChem knowledge base by using *ontocompchem:hasProgram*, *ontocompchem:hasProgramVersion*, and *ontocompchem:hasRunDate* role names (see Table 3) respectively. The domain restriction for first two role names is the *gc:SourcePackage* concept name that is also imported from the GNVC ontology.

The term *formal charge* is the charge assigned to an atom in a molecule, assuming that electrons in a chemical bond are shared equally between atoms, regardless of relative electro negativity. This term is encoded in the OntoCompChem knowledge base as the

Table 3: Selected domain and range restrictions on new roles added to the OntoCompChem knowledge base.

Role name	Domain and range restrictions on new roles	Description
hasEnvironment	$\exists$ ontocompchem:hasEnvironment.$\top \sqsubseteq$ gc:MolecularMethodology	The environment that the job used or required[13].
hasInitialization	$\exists$ ontocompchem:hasInitialization.$\top \sqsubseteq$ gc:MolecularMethodology	Represents the concept of the model parameters and inputs for computational job[13].
hasRotationalConstantsCount	$\exists$ ontocompchem:hasRotationalConstantsCount.$\top$ $\sqsubseteq$ ontocompchem:RotationalConstants $\top \sqsubseteq \forall$ ontocompchem:hasRotationalConstantsCount.Datatypestring	The number of magnitude of spacing between rotational energy levels.
hasSpinMultiplicity	$\exists$ ontocompchem:hasSpinMultiplicity.$\top \sqsubseteq$ gc:Molecule $\top \sqsubseteq \forall$ ontocompchem:hasSpinMultiplicity. Datatypestring	The number of alpha electrons, minus beta electrons, plus one[13].
hasRotationalSymmetryNumber	$\exists$ ontocompchem:hasRotationalSymmetryNumber. $\top \sqsubseteq$ ontocompchem:RotationalSymmetry $\top \sqsubseteq \forall$ontocompchem:hasRotationalSymmetryNumber.Datatypestring	The number of different but indistinguishable (or equivalent) arrangements (or views) of the object.
hasRotationalConstants	$\exists$ ontocompchem:hasRotationalConstants.$\top$ $\sqsubseteq$ ontocompchem:RotationalConstants $\top \sqsubseteq \forall$ontocompchem:hasRotationalConstants. Datatypestring	See Table 2
hasProgramVersion	$\exists$ ontocompchem:hasProgramVersion.$\top$ $\sqsubseteq$ gc:SourcePackage $\top \sqsubseteq \forall$ontocompchem:hasProgramVersions. Datatypestring	Version of software used for comptutation.
hasProgram	$\exists$ ontocompchem:hasProgram.$\top$ $\sqsubseteq$ gc:SourcePackage $\top \sqsubseteq \forall$ontocompchem:hasProgram. Datatypestring	Name of software used for computation.
hasLevelOfTheory	$\exists$ ontocompchem:hasLevelOfTheory.$\top$ $\sqsubseteq$ ontocompchem:LevelOfTheory $\top \sqsubseteq \forall$ontocompchem:hasLevelOfTheory. Datatypestring	A set of underlying approximations used to describe the chemical system.

Table 4: Selected domain and range restrictions on new roles added to the OntoCompChem knowledge base (cont.).

Role name	Domain and range restrictions on new roles	Description
hasFrequencies	$\exists$ ontocompchem:hasFrequencies.$\top$ $\sqsubseteq$ gc:Frequency $\top \sqsubseteq \forall$ontocompchem:hasFrequencies. Datatypestring	The number of occurrences of a repeating event per unit of time.
hasGeometryType	$\exists$ ontocompchem:hasGeometryType.$\top$ $\sqsubseteq$ ontocompchem:GeometryType $\top \sqsubseteq \forall$ontocompchem:hasGeometryType. Datatypestring	Classifies chemical species into three categories: atomic, linear and nonlinear.
hasFormalCharge	$\exists$ gc:hasFormalCharge.$\top$ $\sqsubseteq$ (gc:Atom $\sqcup$ gc:Molecule) $\top \sqsubseteq \forall$gc:hasFormalCharge.Datatypestring	The formal charge is value assigned to an atom or molecule.

role name *gc:hasFormalCharge*. This role name is imported from the GNVC ontology. In the OntoCompChem knowledge base, the domain restriction on this role name is the concept (*gc:Atom* $\sqcup$ *gc:Molecule*). The domain restriction on the *gc:hasFormalCharge* role name implemented in the OntoCompChem knowledge base differs from the implementation of the domain restriction on the same role name in the GNVC ontology. In the GNVC ontology this domain restriction does not include the *gc:Molecule* concept name (see Table 4).

The term *level of theory* denotes underlying approximations used to describe a chemical system. Higher levels of theory are often more accurate. However, they come at much greater computational cost. This term is implemented in the OntoCompChem knowledge base as the role name *ontocompchem:hasLevelOfTheory* (see Table 3) and the concept name *ontocompchem:LevelOfTheory* (see Table 2) that is subsumed by the concept name *gc:MethodologyFeature*. The range of the property *ontocompchem:hasLevelOfTheory* is of the data type string which takes the value that is used for the description of level of theory in the Gaussian output file.

MolHub Framework: Design and Implementation

This section outlines the design and implementation of MolHub as a semantic web framework that is a component of JPS. It is a replacement of the online infrastructure previously developed by Phadungsukanan et al.[15] and extends the JPS knowledge graph with quantum chemistry calculations. The previous version of MolHub allowed users to upload Gaussian files, to transform them into XML and Resource Description Framework (RDF/XML) formats[38], to run thermochemistry calculations, and to provide a list of species names. However, that infrastructure lacked a query engine that can be used for the semantic query of quantum chemistry calculations. Also, it did not offer users to manipulate data by using visual events. For example, users did not have the option to measure any angle between atoms on the 3D model of a selected species. The novel MolHub framework inherits and shares common goals from the previous platform, but also adds some new features which were previously not supported. The aims of our additions to MolHub are:

- To base the new MolHub on the OntoCompChem ontology. The ontology is developed based on the OntoCompChem knowledge base and it is used to support semantic query and reasoning;
- To develop a query language for the OntoCompChem knowledge base entries. The query language uses in its syntax terms from the OntoCompChem ontology such as ”atom name” and ”number of atoms”. The query engine validates the query and checks its satisfiability. If both conditions are satisfied, then the query engine generates a number of SPARQL queries. These queries are performed on the remote knowledge graph repository;
- To parse uploaded Gaussian files, generate XML files, and to validate generated XML files against the extension of CompChem convention of CML;

- To transform the generated XML files into OWL files and check consistency of the OWL files by using the HermiT reasoner;
- To support semantic interoperability between different computational chemistry software tools. For example, for the selected set of species, the novel MolHub offers to humans or software tools to run NASA polynomial calculation. The JSON format is used to store the result of this calculation;
- To support 3D visualization of species, and to allow users to manipulate with 3D objects by using visual events. For example, users have option to use a ruler in order to measure distances between two atoms on selected species.

Implementation of MolHub Framework

The novel MolHub is implemented by using the model view controller (MVC) software architecture design pattern[39,40]. *Model* component consists of data storage (see modules A and B in Figure 1), a set of actions that users can take (see labels 1, 6, and 7 in Figure 1), including the business logic.

We use two type of storages to save quantum chemistry calculations performed by Gaussian. The first one is the remote knowledge graph repository used for processing RDF/XML data, and the second storage includes the set of folders on a server used for storing uploaded and generated files. Both stores are related to three actions that users can take. The first action, labeled with 1 in Figure 1, is named *upload action.* On HyperText Transfer Protocol (HTTP) request, the novel MolHub uploads on the server all selected Gaussian files. For every uploaded Gaussian file, the novel MolHub generates corresponding XML file (see label 2 in Figure 1), 3D image of a molecule (see label 3 in Figure 1), and OWL files (see label 4 in Figure 1). All of these files are stored in the folder that is named by using the Universal Unique Identifier (UUID). Additionally, the novel MolHub stores generated OWL files in the remote knowledge graph repository

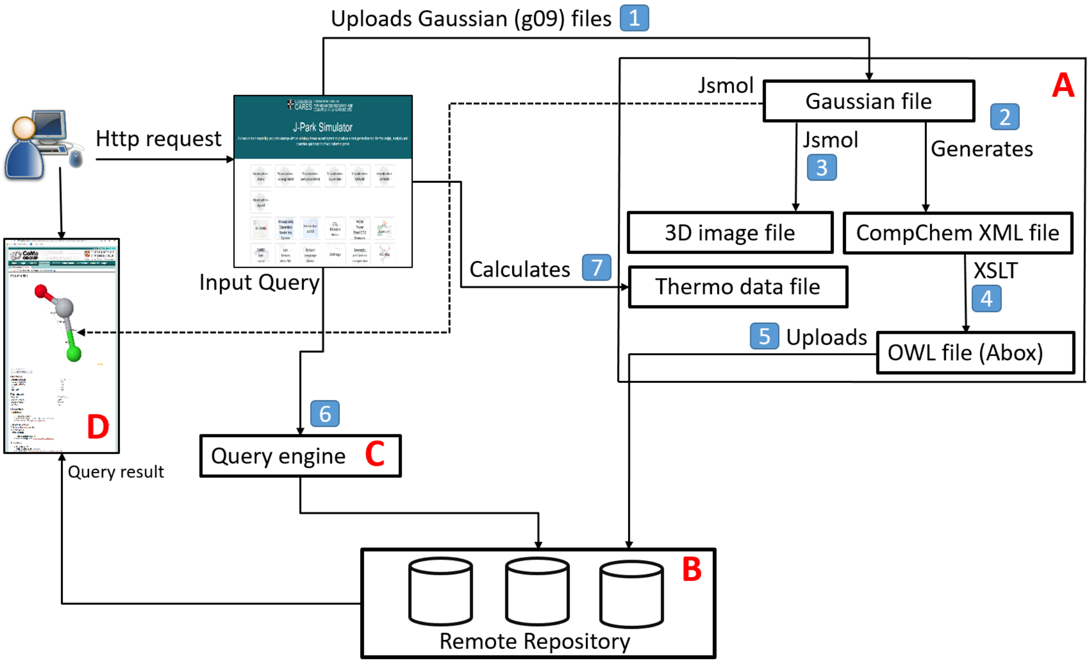


Figure 1: MolHub framework components and sequence of actions users are involved.

(see label 5 in Figure 1). The second action that users can take is named *query action*. When users submit a query, then the novel MolHub delivers that query (see label 6 in Figure 1) to the controller embedded in the query engine (module C in Figure 1). The controller validates that query and generates a number of SPARQL queries (component B in Figure 1). The business logic requires from users to take the *upload action* before taking the *query action*. If users run the *query action* before the *upload action* then the novel MolHub may deliver empty result to users. The third action that the novel MolHub supports is named *calculation action* (see label 7 in Figure 1). It runs different type of thermochemistry calculations by using data stored as OWL files or stored in the remote knowledge graph repository. This action depends on the *query action*. Users are not able to run any type of thermochemistry calculations before selecting species names by using the query engine.

In the novel MolHub, the *business logic* bridges data storage component and Java beans[40]. The business logic is implemented in modules A and C as shown in Figure 1.

It consists of three parts. Each of these three parts are related to exactly one action in the *Model*. When the novel MolHub receives an HTTP request then it takes at least one action implemented in the business logic. For example, the business logic related to *upload action*, validates each generated XML file against the CompChem convention of CML, and adds information about it to Java beans. A report about the validation status of the generated file is made available in *view model* of the novel MolHub. The business logic that corresponds to the *query action* populates Java beans with query results. The *calculation action* invokes the business logic that runs NASA polynomial calculations. As an input, this part of the business logic uses data that are the result of SPARQL queries.

View component (see module D in Figure 1) is implemented by modifying and adapting dual-MVC design pattern[39]. It mainly presents query results to users. The view component of the novel MolHub consists of two parts. The first part shows users a small set of query results in the form of plain text. The second part of the view component allows users to submit a request to view the full set of results of quantum chemistry calculations including 3D animation of molecules. It also offers users to download all available files generated by the novel MolHub.

Controller manipulates with all events that change *Model* or *View*[40]. The novel MolHub framework has implemented three controllers. The first one is responsible for manipulating with query requests, the second controller is responsible for the process of uploading Gaussian files and generating 3D image of species, and the third controller processes requests for thermochemistry calculations. All controllers are executed on server side. The controller embedded in the query engine (see Figure 2) maps HTTP request to a sequence of actions in order to query the remote knowledge graph repository, and returns results to users. The Davis-Putnam-Logemann-Loveland (DPLL) procedure[41,42] is used to check the satisfiability of the input query (see label 2 in Figure 2).

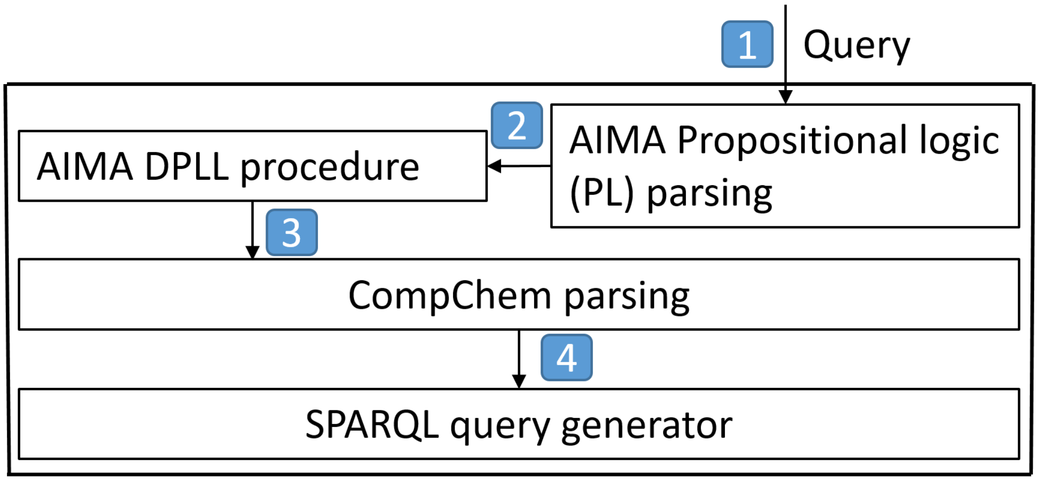


Figure 2: Component C: Query engine components.

The controller responsible for uploading Gaussian files (see Figure 3) maps the HTTP request for uploading selected Gaussian files to a set of actions. These actions upload Gaussian files on the server and generate XML, OWL, and png (jpg) files respectively. The Java Architecture for XML Binding (JAXB)[43] classes are integrated within the existing parser[44] and used to parse Gaussian files and generate XML files (see label 1 in Figure 3). These classes and interfaces are generated from the CompChem convention of CML[15]. One important task that controller takes is to map HTTP requests to a sequence of validation actions. Every generated XML file must pass validation against the CompChem convention of CML (see label 2 in Figure 3)[13]. After finishing the validation of the XML file, XSLT[45] script transforms that XML file into OWL file.

Generated OWL files are individual assertions[46] of the OntoCompChem ontology that is derived from the OntoCompChem knowledge base (see OntoCompChem Knowledge Base section). Before storing generated OWL files into the JPS knowledge graph the controller runs the HermiT inference engine to check the consistency (see[36], p. 28) of each generated OWL file (see label 3 in Figure 3). Inconsistent OWL file will not be stored into the JPS knowledge graph. In addition the controller uses the HermiT reasoner to check the consistency (see[36], p. 28) of generated OWL files before invoking

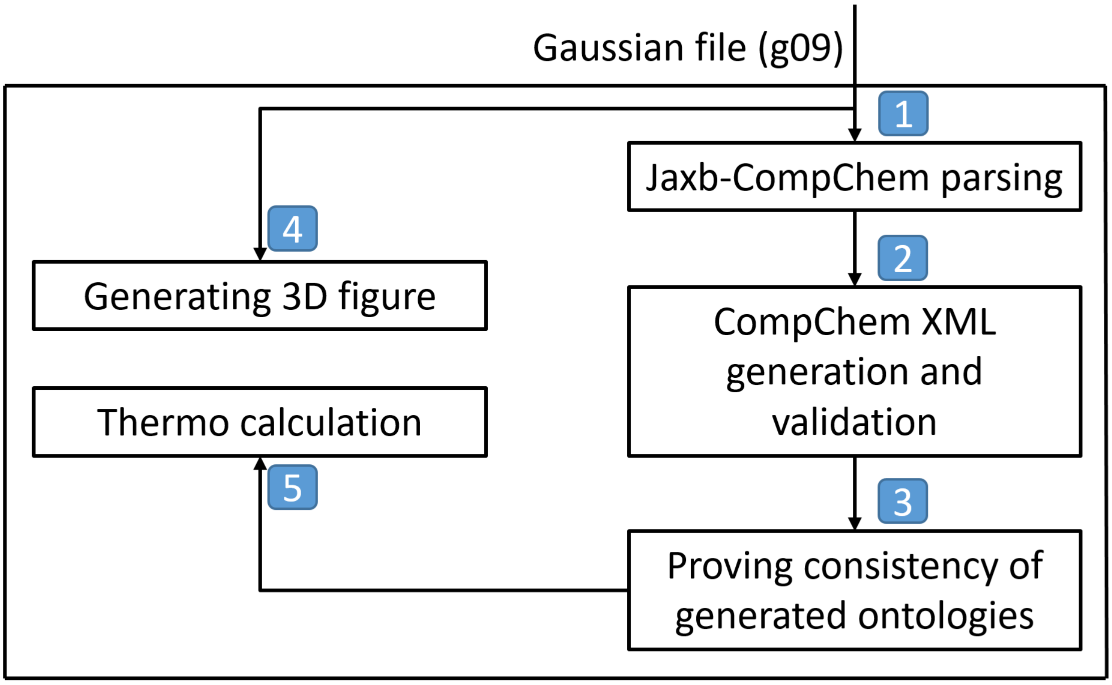


Figure 3: Component A: MolHub server components.

calculation action.

The maximum number of Gaussian files that can be uploaded to MolHub is limited by the number of RDF triples allowed in an RDF4J database. The latter can support 10^8 triples[47]. Based on a sample of about 2700 Gaussian files available to us, we estimate that the triple-store would therefore be able to support about 2×10^5 Gaussian files.

Implementation of Query Engine

To enable users to query quantum chemistry calculations stored in the JPS knowledge graph, the novel MolHub framework allows users to utilize propositional logic like formal language as a query language. A query expressed in that formal language is interpreted by the query engine. Three type of symbols are used to build queries[48]. Symbols from the countable set $\mathbf{P} = \{q_0, q_1, q_2, \ldots\}$ stand for atomic queries. Symbols from finite set $\mathbf{L_O} = \{not, and, or, implies, equals\} \cup \{\sim, \&, |, \Rightarrow, \Leftrightarrow\}$ denote logical operations, and elements of the finite set $\mathbf{I} = \{(,)\}$ are interpunction symbols. Set $\mathbf{L}$ is formed by the union of these three sets,

$$\mathbf{L} := \mathbf{P} \cup \mathbf{L_O} \cup \mathbf{I}. \tag{1}$$

Each member of set **P** must satisfy regular expression of the form (2)

$$[A - Z][a - z]\{0, 3\}[0 - 9] + . \tag{2}$$

It means that each member of this set must start with an atom name that has an exact match with corresponding name in the periodic table of chemical elements, following by the number of atoms that appears in the species name. Based on this regular expression (2), it is easy to check that *Cl2* is a well formed atomic query because term *Cl* is the member of the periodic table, and number *2* denotes the exact number of atoms that appears in the species name. As a counterexample, the atomic query of the form *Xy2* is not well formed because the term *Xy* is not the member of the periodic table.

A well formed query over the set **L** can be defined inductively by using elements from the sets **P**, $\mathbf{L_O}$, and **I** as given in the following definition.

Definition 1 *Let* **L** *be the union of the countable set* **P***, the finite sets* $\mathbf{L_O}$ *and* **I***. The set of well formed queries over set* **L** *is inductively defined as follows:*

- *All elements of the set* **P** *that satisfy a regular expression of the form (2) are atomic queries.*

- *If p and q are atomic queries then (not p), (p and q), (p or q), (p implies q), (p equals q), (∼ p), (p & q), (p | q), (p ⇒ q), and (p ⇔ q) are queries.*

- *The query (not p) can be rewritten as query (∼ p) and query (∼ p) can be rewritten as query (not p).*

- *The query (p and q) can be rewritten as query (p & q) and query (p & q) can be rewritten as query (p and q).*

- *The query (p or q) can be rewritten as query (p | q) and query (p | q) can be rewritten only as query (p or q).*

- *The query (p implies q) can be rewritten as query (p ⇒ q) and query (p ⇒ q) can be rewritten as query (p implies q).*

- *The query (p equals q) can be rewritten as query (p ⇔ q) and query (p ⇔ q) can be rewritten as query (p equals q).*

- *Any query can be created only by a finite number of applications of previous rules.*

For example, the query $((Cl1\, and\, Ti1) \mid (Cl1\, \&\, Ti1))$ is well formed with respect to definition (1). The implementation of the query engine has two layers. The first layer is responsible for checking satisfiability and whether a query is well formed according to rules given in definition (1). In the novel MolHub, query q is satisfiable if there is an interpretation I such that $I \models q$ (pp. 36, 37, 38;[48]). If users or software tools submit an unsatisfiable query or a query that is not well formed then the novel MolHub returns an error message. For example, a query of the form

$$(not \quad Cl2 \quad and \quad Cl2) \tag{3}$$

is well formed but it is not satisfiable. In this case the query engine does not generate a number of SPARQL queries and throws an exception.

Whereas, a query of the form

$$\sim \quad (Cl2 \quad \Rightarrow \quad \sim \quad Ti3) \tag{4}$$

is both, well formed and satisfiable.

The second layer of the query engine converts the input query into the conjunctive normal form (CNF) (see[48], pp. 53-54) that is equivalent to the starting query. The CNF is a conjunction of clauses, where each clause is disjunction of positive or negative literals.[48] The CNF of the query given in the example (4) is

$$Cl2 \quad \& \quad Ti3. \tag{5}$$

In the case of performing query (5), the query engine converts this query into intersection of two clause sets $\{Cl2\}$ and $\{Ti3\}$. For the first clause set, the query engine generates one SPARQL query that finds all species names that contain $Cl2$ in their names. For the second clause set, the query engine generates another SPARQL query that finds all species names that contain $Ti3$ in their names. The intersection of these two query results gives the set of species names that contains $Cl2$ and $Ti3$ in their names. The query engine iterates over this set and generates one SPARQL query for each species name that is the member of that set. Finally, the result of these queries should be the set of molecule properties presented to users through *View* model of the novel MolHub framework. The number of generated SPARQL queries depends on the number of elements in each clause set and the number of elements in the species set.

Implementation of Thermo Calculations

In many applications, not only species molecular data, but also how they manifest on a macro-scale in the form of thermodynamic data are required. The species molecular data are already easily accessible via the novel MolHub querying engine (see Implementation of Query Engine section). In order to provide associated thermodynamic data, a simple thermodynamic data calculator code (TDC) has been added to the novel MolHub framework.

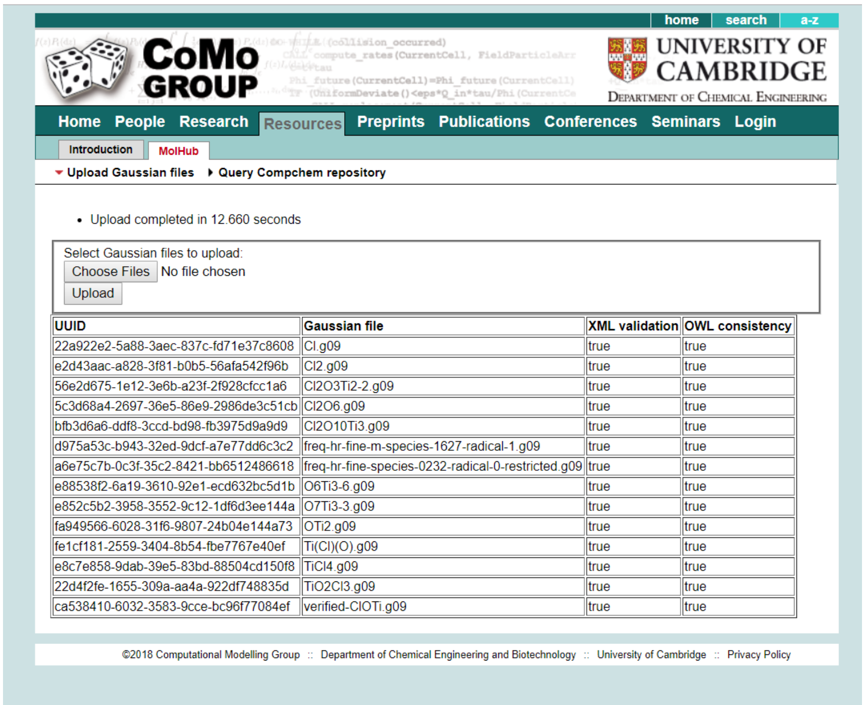
The thermodynamic data calculator, as the name suggests, calculates species thermodynamic properties data (heat capacities, entropy and enthalpy) using statistical thermodynamics. The molecular partition functions are obtained by applying the rigid rotor harmonic oscillator treatment (RRHO) and include the following energy contributions: translational, vibrational, rotational and electronic. Details of these computations may be found in various texts[49,50] so they will not be repeated here. Note that the calculated species enthalpies are not referenced to their standard states and must be corrected if absolute enthalpies are to be used. This is achieved by providing a standard enthalpy of formation of species of interest as an additional input to the TDC. All other species data are automatically taken from the JPS knowledge graph.

In order to facilitate passing data between TDC and other modelling software (e.g *k*inetics & SRM Engine Suite, Chemkin or Cantera) the calculated thermodynamic data are additionally fitted into a standard NASA polynomial form.

MolHub Framework: A Use Case

This section explains the NASA polynomial calculation use case scenario. It demonstrates the semantic interoperability between tools for quantum chemistry calculations and tools for thermochemistry calculations. By clicking on the *Choose Files* button, users have the option to select one or more Gaussian files to be uploaded on the server for further processing. After selecting one or more Gaussian files, users should press the *Upload* button. Then the novel MolHub generates the report shown in Figure 4. The first column in the report provides information about the UUID that denotes the name of the folder where the novel MolHub stores Gaussian files and names of XML, OWL, and PNG/JPG files respectively.

In the second column of the report, the novel MolHub informs users about Gaussian file names that are uploaded on the server. If users upload the same Gaussian file twice, then two different UUID are generated. The third column reports whether generated



- Upload completed in 12.660 seconds

Select Gaussian files to upload:
Choose Files No file chosen
Upload

UUID	Gaussian file	XML validation	OWL consistency
22a922e2-5a88-3aec-837c-fd71e37c8608	Cl.g09	true	true
e2d43aac-a828-3f81-b0b5-56afa542f96b	Cl2.g09	true	true
56e2d675-1e12-3e6b-a23f-2f928cfcc1a6	Cl2O3Ti2-2.g09	true	true
5c3d68a4-2697-36e5-86e9-2986de3c51cb	Cl2O6.g09	true	true
bfb3d6a6-ddf8-3ccd-bd98-fb3975d9a9d9	Cl2O10Ti3.g09	true	true
d975a53c-b943-32ed-9dcf-a7e77dd6c3c2	freq-hr-fine-m-species-1627-radical-1.g09	true	true
a6e75c7b-0c3f-35c2-8421-bb6512486618	freq-hr-fine-species-0232-radical-0-restricted.g09	true	true
e88538f2-6a19-3610-92e1-ecd632bc5d1b	O6Ti3-6.g09	true	true
e852c5b2-3958-3552-9c12-1df6d3ee144a	O7Ti3-3.g09	true	true
fa949566-6028-31f6-9807-24b04e144a73	OTi2.g09	true	true
fe1cf181-2559-3404-8b54-fbe7767e40ef	Ti(Cl)(O).g09	true	true
e8c7e858-9dab-39e5-83bd-88504cd150f8	TiCl4.g09	true	true
22d4f2fe-1655-309a-aa4a-922df748835d	TiO2Cl3.g09	true	true
ca538410-6032-3583-9cce-bc96f77084ef	verified-ClOTi.g09	true	true

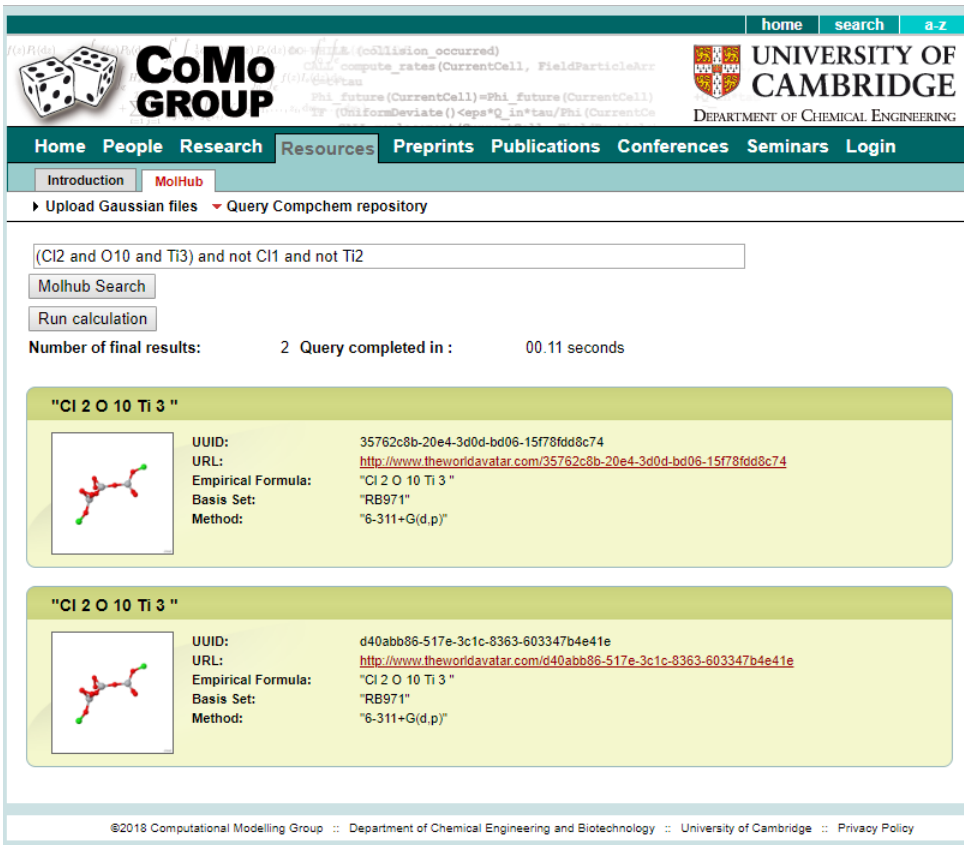
©2018 Computational Modelling Group :: Department of Chemical Engineering and Biotechnology :: University of Cambridge :: Privacy Policy

Figure 4: MolHub: Service for uploading Gaussian files, and validating CompChem XML and OntoCompChem OWL files.

XML files are valid against CompChem convention of CML, and the last column informs users whether generated OWL files are consistent. In the current implementation we chose not to display inconsistency explanation for generated OWL files, if such a case occurs. Including an inconsistency report will form part of future work.

Figure 5 shows the interface for querying the JPS knowledge graph stored in a remote repository and for running thermochemistry calculations. On the same page, when the novel MolHub delivers to users the results of queries, users receive information about the number of results and time needed to complete all generated queries. This use case shows the results for the query *((Cl2 and O10 and Ti3) and not Cl1 and not Ti2)*. This query is well formed and satisfiable.

By submitting this query, the novel MolHub requires from the query engine to find all data about species that contain in their names two atoms of *Cl*, ten atoms of *O*, three atoms of *Ti*, and at the same time do not have one atom of *Cl* and two atoms of *Ti*.



(Cl2 and O10 and Ti3) and not Cl1 and not Ti2

Number of final results: 2 Query completed in : 00.11 seconds

"Cl 2 O 10 Ti 3 "

UUID:	35762c8b-20e4-3d0d-bd06-15f78fdd8c74
URL:	http://www.theworldavatar.com/35762c8b-20e4-3d0d-bd06-15f78fdd8c74
Empirical Formula:	"Cl 2 O 10 Ti 3 "
Basis Set:	"RB971"
Method:	"6-311+G(d,p)"

"Cl 2 O 10 Ti 3 "

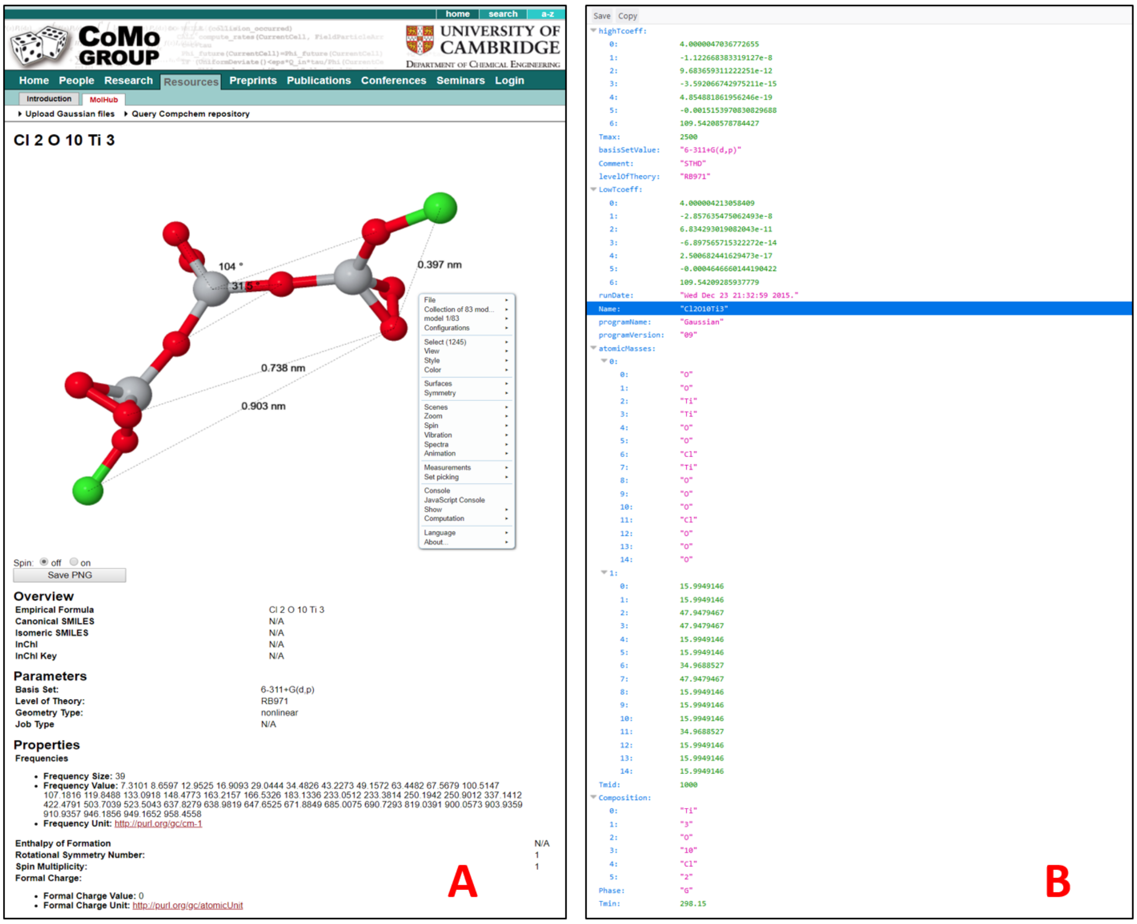
UUID:	d40abb86-517e-3c1c-8363-603347b4e41e
URL:	http://www.theworldavatar.com/d40abb86-517e-3c1c-8363-603347b4e41e
Empirical Formula:	"Cl 2 O 10 Ti 3 "
Basis Set:	"RB971"
Method:	"6-311+G(d,p)"

©2018 Computational Modelling Group :: Department of Chemical Engineering and Biotechnology :: University of Cambridge :: Privacy Policy

Figure 5: Services for thermochemistry calculations and querying JPS knowledge graph stored in remote repository.

On pressing *Molhub Search* button, users will observe the total number of results for the given query (see Figure 5). All of these results have the same species name, and different UUID that is used to create unique Uniform Resource Locator (URL). By clicking on the URL, the novel MolHub shows to users all information about the selected species that are available in the JPS knowledge graph as shown in Figure 6 (A).

Also, on this page (Figure 6, A) users can inspect molecule properties using visual events, as made available by Jmol[51]. By pressing the button *Run calculation*, as shown in Figure 5, users request from MolHub to run a NASA polynomial calculation for a species, in this example $Cl_2O_{10}Ti_3$. By using the query engine, MolHub delivers data to the Python script that calculates NASA polynomials. As a result of a NASA polynomial calculation, a JSON file is generated for each selected species. That file is stored in the



Cl 2 O 10 Ti 3

Overview

Empirical Formula	Cl 2 O 10 Ti 3
Canonical SMILES	N/A
Isomeric SMILES	N/A
InChI	N/A
InChI Key	N/A

Parameters

Basis Set:	6-311+G(d,p)
Level of Theory:	RB971
Geometry Type:	nonlinear
Job Type	N/A

Properties

Frequencies

- **Frequency Size:** 39
- **Frequency Value:** 7.3101 8.6597 12.9525 16.9093 29.0444 34.4826 43.2273 49.1572 63.4482 67.5679 100.5147 107.1816 119.8488 133.0918 148.4773 163.2157 166.5326 183.1336 233.0512 233.3814 250.1942 250.9012 337.1412 422.4791 503.7039 523.5043 637.8279 638.9819 647.6525 671.8849 685.0075 690.7293 819.0391 900.0573 903.9359 910.9357 946.1856 949.1652 958.4558
- **Frequency Unit:** http://purl.org/gc/cm-1

Enthalpy of Formation	N/A
Rotational Symmetry Number:	1
Spin Multiplicity:	1
Formal Charge:	

- **Formal Charge Value:** 0
- **Formal Charge Unit:** http://purl.org/gc/atomicUnit

```
▾ highTcoeff:
    0:              4.0000047036772655
    1:              -1.122668383319127e-8
    2:              9.683659311222251e-12
    3:              -3.592066742975211e-15
    4:              4.854881861956246e-19
    5:              -0.0015153970830829688
    6:              109.54208578784427
  Tmax:             2500
  basisSetValue:    "6-311+G(d,p)"
  Comment:          "STHD"
  levelOfTheory:    "RB971"
▾ LowTcoeff:
    0:              4.000004213058409
    1:              -2.857635475062493e-8
    2:              6.834293019082043e-11
    3:              -6.897565715322272e-14
    4:              2.500682441629473e-17
    5:              -0.0004646660144190422
    6:              109.54209285937779
  runDate:          "Wed Dec 23 21:32:59 2015."
  Name:             "Cl2O10Ti3"
  programName:      "Gaussian"
  programVersion:   "09"
▾ atomicMasses:
  ▾ 0:
      0:            "O"
      1:            "O"
      2:            "Ti"
      3:            "Ti"
      4:            "O"
      5:            "O"
      6:            "Cl"
      7:            "Ti"
      8:            "O"
      9:            "O"
      10:           "O"
      11:           "Cl"
      12:           "O"
      13:           "O"
      14:           "O"
  ▾ 1:
      0:            15.9949146
      1:            15.9949146
      2:            47.9479467
      3:            47.9479467
      4:            15.9949146
      5:            15.9949146
      6:            34.9688527
      7:            47.9479467
      8:            15.9949146
      9:            15.9949146
      10:           15.9949146
      11:           34.9688527
      12:           15.9949146
      13:           15.9949146
      14:           15.9949146
  Tmid:             1000
▾ Composition:
    0:              "Ti"
    1:              "3"
    2:              "O"
    3:              "10"
    4:              "Cl"
    5:              "2"
  Phase:            "G"
  Tmin:             298.15
```

Figure 6: (A) Service for viewing data as a result of querying JPS knowledge graph repository, 3D visualization of species, and downloading XML, OWL, and JSON files of a selected $Cl_2O_{10}Ti_3$ molecule. (B) Result of thermochemistry calculations expressed in JSON format for the selected molecule (see supplementary material).

same folder where the XML and the OWL file of the species are stored. Figure 6 (B) shows a JSON file as a result of a NASA polynomial calculation for a $Cl_2O_{10}Ti_3$ species.

Conclusions

A knowledge base for quantum chemistry calculations, named OntoCompChem, has been proposed as part of the JPS knowledge graph. The OntoCompChem knowledge base provides the implementation of sixteen features specified in the CompChem convention of CML. To describe the OntoCompChem knowledge base, we use DL syntax in this paper. This knowledge base is implemented in the OntoCompChem ontology. The OntoCompChem ontology improves web services for quantum chemistry calculations by allowing machines to interpret all implemented aspects of the OntoCompChem knowledge base as well as to keep consistency of quantum chemistry calculations. The main contributions of this paper are:

- The introduction of the OntoCompChem knowledge base that is implemented as OntoCompChem ontology that forms part of the JPS knowledge graph;
- The development of the novel MolHub as semantic web service for uploading and parsing Gaussian files, generating XML and OWL files, and validating these files, which are all available for download;
- The development of a web service for the visualization of quantum chemistry calculations;
- The development of a novel method and system to explore quantum chemistry calculations stored in the JPS knowledge graph by using propositional logic as a formal language;
- The generation of SPARQL queries based on formal language expressions.

All of these contributions allow semantic interoperability between tools for quantum chemistry calculations and thermochemistry calculations, and at the same time guarantee the consistency of data used in these calculations. The OntoCompChem ontology in near future will have implemented all features defined in the CompChem convention of CML. We aim to offer users software tools and services to query cross domain ontologies stored in the JPS knowledge graph by using a more expressive formal language than proposed in this paper. One research direction would be to implement propositional logic with binary metric operators as a query language that could be used in the semantic query of cross-domain repositories in the JPS and to support the semantic interoperability between different tools in computational chemistry. Further directions include establishing links to experimental as well as wider chemical semantic work, as part of larger projects to create integrated, cross-domain knowledge-graphs for Industry 4.0.

Acknowledgements

This project is supported by the National Research Foundation (NRF), Prime Minister's Office, Singapore under its Campus for Research Excellence and Technological Enterprise (CREATE) programme. Markus Kraft acknowledges the support of the Alexander von Humboldt foundation.

References

(1) Frisch, M. J.; Trucks, G. W.; Schlegel, H. B.; Scuseria, G. E.; Robb, M. A.; Cheeseman, J. R.; Scalmani, G.; Barone, V.; Mennucci, B.; Petersson, G. A.; Nakatsuji, H.; Caricato, M.; Li, X.; Hratchian, H. P.; Izmaylov, A. F.; Bloino, J.; Zheng, G.; Sonnenberg, J. L.; Hada, M.; Ehara, M.; Toyota, K.; Fukuda, R.; Hasegawa, J.; Ishida, M.; Nakajima, T.; Honda, Y.; Kitao, O.; Nakai, H.; Vreven, T.; Montgomery, J. A., Jr.; Peralta, J. E.; Ogliaro, F.; Bearpark, M.; Heyd, J. J.; Brothers, E.; Kudin, K. N.; Staroverov, V. N.; Kobayashi, R.; Normand, J.; Raghavachari, K.; Rendell, A.; Burant, J. C.; Iyengar, S. S.; Tomasi, J.; Cossi, M.; Rega, N.; Millam, J. M.; Klene, M.; Knox, J. E.; Cross, J. B.; Bakken, V.; Adamo, C.; Jaramillo, J.; Gomperts, R.; Stratmann, R. E.; Yazyev, O.; Austin, A. J.; Cammi, R.; Pomelli, C.; Ochterski, J. W.; Martin, R. L.; Morokuma, K.; Zakrzewski, V. G.; Voth, G. A.; Salvador, P.; Dannenberg, J. J.; Dap-

prich, S.; Daniels, A. D.; Farkas, O.; Foresman, J. B.; Ortiz, J. V.; Cioslowski, J.; Fox, D. J. Gaussian 09 Revision A.1. 2009; Gaussian Inc. Wallingford CT 2009.

(2) R. D. Johnson III (ed.), NIST Computational Chemistry Comparison and Benchmark Database, NIST Standard Reference Database Number 101, Release 19. 2018; `http://cccbdb.nist.gov/`, Accessed March 05th, 2019.

(3) Ghahremanpour, M. M.; van Maaren, P. J.; van der Spoel, D. The Alexandria library, a quantum-chemical database of molecular properties for force field development. *Nat. Sci. Data* **2018**, *5*, 180062.

(4) Smith, J.; Isayev, O.; E. Roitberg, A. ANI-1, A data set of 20 million calculated off-equilibrium conformations for organic molecules. *Nat. Sci. Data* **2017**, *4*, 170193.

(5) Rupp, M.; Ramakrishnan, R.; von Lilienfeld, O. A. Machine Learning for Quantum Mechanical Properties of Atoms in Molecules. *J. Phys. Chem. Lett.* **2015**, *6*, 3309–3313.

(6) Simmie, J. M. A Database of Formation Enthalpies of Nitrogen Species by Compound Methods (CBS-QB3, CBS-APNO, G3, G4). *J. Phys. Chem. A* **2015**, *119*, 10511–10526.

(7) Nakata, M.; Shimazaki, T. PubChemQC Project: A Large-Scale First-Principles Electronic Structure Database for Data-Driven Chemistry. *J. Chem. Inf. Model.* **2017**, *57*, 1300–1308.

(8) Hait, D.; Head-Gordon, M. How accurate are static polarizability predictions from density functional theory? An assessment over 132 species at equilibrium geometry. *Phys. Chem. Chem. Phys.* **2018**, *20*, 19800–19810.

(9) Berners-Lee, T.; Hendler, J.; Lassila, O. The Semantic Web. *Sci. Am.* **2001**, *284*, 34–43.

(10) Eibeck, A.; Lim, M. Q.; Kraft, M. J-Park Simulator: An ontology-based platform for cross-domain scenarios in process industry. 2019; Submitted for publication.

(11) Rigo-Mariani, R.; Zhang, C.; Romagnoli, A.; Kraft, M.; Ling, K. V.; Maciejowski, J. M. A Combined Cycle Gas Turbine Model for Heat and Power Dispatch Subject to Grid Constraints. *IEEE Trans. Sustainable Energy* **2019**,

(12) OWL 2 Web Ontology Language Structural Specification and Functional-Style Syntax (Second Edition). 2012; `https://www.w3.org/TR/owl2-syntax/`, Accessed February 15th, 2019.

(13) Phadungsukanan, W.; Adams, S.; Townsend, J.; Thomas, J. Chemical Markup Language - CompChem Convention. 2011; `http://www.xml-cml.org/convention/compchem`, Accessed October 22th, 2018.

(14) Rzepa, H.; Murray-Rust, P. Chemical Markup Language (CML) Schema, version 3. 2018; `http://www.xml-cml.org/schema/schema3/`, Accessed October 26th, 2018.

(15) Phadungsukanan, W.; Kraft, M.; Townsend, J. A.; Murray-Rust, P. The semantics of Chemical Markup Language (CML) for computational chemistry : CompChem. *J. Cheminf.* **2012**, *4*, 15.

(16) Ostlund, N. S.; Sopek, M. Applying the Semantic Web to Computational Chemistry. Proceedings of the 6th International Workshop on Semantic Web Applications and Tools for Life Sciences (SWAT4LS 2013). Edinburgh, United Kingdom, 2013; Accessed February 7th, 2019.

(17) Wang, B.; Dobosh, P. A.; Chalk, S.; Ito, K.; Sopek, M.; Ostlund, N. S. A Portal for Quantum Chemistry Data Based on the Semantic Web. Concepts, Methods and Applications of Quantum Systems in Chemistry and Physics. 2018; pp 3–27.

(18) Wang, B.; Dobosh, P. A.; Chalk, S.; Sopek, M.; Ostlund, N. S. Computational Chemistry Data Management Platform Based on the Semantic Web. *J. Phys. Chem.* **2017**, *121*, 298–307.

(19) Ostlund, N. S.; Sopek, M. GNVC: Gainesville Core Ontology - standard for publishing results of computational chemistry, ver. 0.7. 2015; `http://ontologies.makolab.com/gc/gc07.owl`, Accessed October 24th, 2018.

(20) OpenLink Virtuoso. 2019; `https://virtuoso.openlinksw.com/`, Accessed February 15th, 2019.

(21) Schmidt, M.; Baldridge, K.; Boatz, J.; Elbert, S.; Gordon, M.; Jensen, J.; Koseki, S.; Matsunaga, N.; Nguyen, K.; Su, S.; Windus, T.; Dupuis, M.; Montgomery, J. General Atomic and Molecular Electronic Structure System. *J. Comput. Chem.* **1993**, *14*, 1347–1363.

(22) Corney, D.; Miller, S. UK Collaborative Computational Projects. 2018; `http://www.ccp.ac.uk/`, Accessed February 15th, 2018.

(23) Couch, P. A.; Sherwood, P.; Sufi, S.; Todorov, I.; Allan, R. J.; Knowles, P. J.; Bruin, R. P.; Dove, M. T.; Murray-Rust, P. Towards data integration for computational chemistry. Proceedings of the UK e-Science All Hands Meeting 2005, EPSRC. 2005; Accessed February 2019.

(24) Hanwell, M. D.; de Jong, W. A.; Harris, C. J. Open chemistry: RESTful web APIs, JSON, NWChem and the modern web application. *J. Cheminf.* **2017**, *9*, 55.

(25) Kraft, M.; Mosbach, S. The future of computational modelling in reaction engineering. *Philos. Trans. R. Soc., A* **2010**, *368*, 3633–3644.

(26) Kleinelanghorst, M. J.; Zhou, L.; Sikorski, J.; Foo, E. Y. S.; Aditya, K.; Mosbach, S.; Karimi, I.; Lau, R.; Kraft, M. J-Park Simulator: Roadmap to Smart

Eco-Industrial Parks. Proceedings of the Second International Conference on Internet of Things, Data and Cloud Computing. New York, NY, USA, 2017; pp 107:1–107:10.

(27) Zhou, L.; Zhang, C.; Karimi, I. A.; Kraft, M. An ontology framework towards decentralized information management for eco-industrial parks. *Comput. Chem. Eng.* **2018**, *118*, 49–63.

(28) Gröger, G.; Plümer, L. CityGML - Interoperable semantic 3D city models. *ISPRS J. Photogramm. Remote Sens.* **2012**, *71*, 12 – 33.

(29) Farazi, F.; Akroyd, J.; Mosbach, S.; Buerger, P.; Nurkowski, D.; Kraft, M. OntoKin: An Ontology for Chemical Kinetic Reaction Mechanisms. 2019; Submitted for publication.

(30) Hlomani, H.; Stacey, D. Approaches, methods, metrics, measures, and subjectivity in ontology evaluation: A survey. *Sem. Web J.* **2014**, *1*, 1–11, Accessed February 7th, 2019.

(31) Musen, M. The Protégé project: A look back and a look forward. *AI Matters* **2015**, *1*.

(32) Horrocks, I.; Kutz, O.; Sattler, U. The Even More Irresistible SROIQ. Proceedings of the 10th International Conference on Principles of Knowledge Representation and Reasoning (KR 2006). 2006; pp 57–67, Accessed October 26th, 2018.

(33) Glimm, B.; Horrocks, I.; Motik, B.; Stoilos, G.; Wang, Z. HermiT: An OWL 2 Reasoner. *J. Autom. Reasoning* **2014**, *53*, 245–269.

(34) Motik, B.; Shearer, R.; Horrocks, I. Hypertableau Reasoning for Description Logics. *J. Artif. Intell. Res.* **2009**, *36*, 165–228.

(35) Polikoff, I.; Masters, J. E. Unit ontology. 2010; `http://data.qudt.org/qudt/owl/1.0.0/unit.owl`, Accessed October 24th, 2018.

(36) Baader, F.; Horrocks, I.; Lutz, C.; Sattler, U. *An Introduction to Description Logic*; Cambridge University Press, 2017.

(37) Smith, M.; Horrocks, I.; Krotzsch, M.; Glimm (eds.), B. OWL 2 Web Ontology Language Conformance (Second Edition). 2012; `https://www.w3.org/TR/owl2-overview/`, Accessed October 26th, 2018.

(38) Fabien, G.; Guus, S. RDF 1.1 XML Syntax, W3C Recommendation. 2014; `https://www.w3.org/TR/rdf-syntax-grammar/`, Accessed February 27th, 2019.

(39) Leff, A.; Rayfield, J. T. Web-Application Development Using the Model/View/Controller Design Pattern. Proceedings of the 5th IEEE International Conference on Enterprise Distributed Object Computing. 2001; p 118, Accessed February 15th, 2019.

(40) Brown, D.; Davis, C.; Stanlick, S. *Struts2 in Action*; Wiley India Pvt. Limited, 2008; Accessed February 15th, 2019.

(41) Harrison, J. *Handbook of Practical Logic and Automated Reasoning*; Cambridge University Press, 2009.

(42) Russell, S.; Norvig, P. *Artificial Intelligence: A Modern Approach*, 3rd ed.; Prentice Hall Press: Upper Saddle River, NJ, USA, 2009; Accessed October 24th, 2018.

(43) Ort, E.; Mehta, B. Java Architecture for XML Binding. 2003; `https://www.oracle.com/technetwork/articles/javase/index-140168.html`, Accessed October 24th, 2018.

(44) Buerger, P.; Akroyd, J.; Martin, J. W.; Kraft, M. A big data framework to validate thermodynamic data for chemical species. *Combust. Flame* **2017**, *176*, 584 – 591.

(45) Kay, M. XSL Transformations (XSLT) Version 2.0. 2007; `https://www.w3.org/TR/xslt20/`, Accessed October 24th, 2018.

(46) Motik, B.; Patel-Schneider, P. F.; Cuenca Grau, B.; Horrocks, I.; Parsia, B.; Sattler, U. OWL 2 Web Ontology Language Direct Semantics (Second Edition). 2012; `https://www.w3.org/TR/2012/REC-owl2-direct-semantics-20121211/`.

(47) Hanssens, B. RDF4J Databases. 2019; `http://rdf4j.org/rdf4j-databases/`, Accessed on the May 15th, 2019.

(48) Huth, M.; Ryan, M. *Logic in Computer Science: Modelling and Reasoning about Systems*, 2nd ed.; Cambridge University Press, 2004.

(49) McQuarrie, D.; Simon, J. *Molecular Thermodynamics*; University Science Books: Sausalito, CA, United States, 1999; Accessed February 7th, 2019.

(50) Seddon, J. M.; Gale, J. D. *Thermodynamics and Statistical Mechanics*; The Royal Society of Chemistry: Cambridge, United Kingdom, 2001; Vol. 10; pp 77–83.

(51) Jmol: an open-source browser-based HTML5 viewer and stand-alone Java viewer for chemical structures in 3D. 2019; `http://jmol.sourceforge.net/`, Accessed February 19th, 2019.

Graphical TOC Entry