Towards a Common Data and Command Representation for Quantum Chemistry

Rob Allan[1], **Philip Couch**[1], Peter Knowles[2] & Paul Sherwood[1]

1. CCLRC, Daresbury Laboratory, Daresbury, Warrington, Cheshire. WA4 4AD. UK.
2. School of Chemistry, Cardiff University, P.O. Box 912, Cardiff, CF10 3TB. UK.

Abstract

Computational chemistry increasingly requires the use of a variety of applications in close cooperation to solve complex problems, and the realisation of Grid technologies has provided a strong technological framework for such integration. However, communication between applications is still hindered by a lack of data format standards and agreed semantics. Consideration of a common data and command representation would alleviate some of this difficulty. In addition, the storage of data in a 'universal' format with suitable meta-data would simplify its analysis, interpretation and appropriate re-use. Another aim is to be able to log metadata conforming to a known data model such that scientific procedures can be more reliably repeatable. Simple conversion between existing application data formats is error prone and not scalable, and a solution is to find a common 'middle ground' for the data representation. The eCCP1 project has been established to formulate the requirements for enabling effective use of Grid resources by the quantum chemistry community. More specifically, the project investigates Grid middleware, compute resources, client tools and tracks and develops standards in data and command representation.

1. Introduction

Quantum chemistry (QC) applications often work with their own unique file formats that tend to be implicit, and have semantics that are only completely understood by those that have considerable experience of the codes themselves. The approach to the use of such computational codes is changing, with the frequent use of several techniques and codes to solve specific complex problems. This arises from a desire to model systems over ever increasing length and time scales. Examples include the study of bio-molecular processes that should ideally be treated with a full quantum mechanical approach, but where the size of the system requires some classical treatment. Another example is in the understanding of environmental processes, where atomistic processes are affecting large systems over an extended period [1]. The current UK e-science program is seeking to address issues associated with the construction of a computational Grid based framework for such code interoperability. However, the process is still stifled by the lack of agreed data, meta-data and semantic standards. In QC tools do exist to convert between a range of the more common file formats (e.g. Open Babel [2]), but this is often only a small subset of formats, that can change between code versions. A common data and command representation for quantum chemistry also has advantages outside those of code interoperability. If formatted data is well supported with meta-data and the semantics are clearly defined, then advantages include the promotion of appropriate data reuse. Conformance to an agreed standard also makes data available to a wider community through the use of tools adapted for this purpose. Such common tools could be used to analyse and visualise data, including rendering structures and the analysis of charge densities (such as a Bader analysis).

There are a number of existing ways that such data could be represented. These include ASCII representations, such as the Crystallographic Information File (CIF) [3], storage in relational tables, use of the eXtensible Mark up Language (XML) such as the Chemical Mark up Language (CML) [4] or as serialised objects with their associated methods and data members. The data models could then be defined in SQL, in XML Schema, using class interface definitions or using diagrammatic forms such as the Unified Modelling Language (UML). Formatted data should be supported by meta-data defined by a meta-data schema. Examples of existing schema include the CCLRC Scientific Meta-data Model [5]. If data is to be shared amongst different codes and stored in repositories for later use, it is important to be clear about the data semantics.

For performance reasons, many scientific application developers work with the Fortran language and careful consideration needs to be given to the implementation of APIs/GUI front and back ends that enable the use of formatted data from such codes.

2. The Data Model

2.1 QC data-types

The initial QC data-types that are to be considered in the data model are those that are commonly used in a range of different applications. These include: geometries, basis sets, commands, properties on grids (such as orbitals and densities), dynamics and accuracies and tolerances. XML has been chosen as the meta-language for the data format, principally due to its wide spread adoption and the availability of mature tools. This makes an XML format easily interpretable by disparate groups of collaborators. A document detailing QC data-type issues can be found on the eCCP1 TWiki site [6].

2.2 The data model design

The data model is currently being defined with UML, using Object Domain, with the aim of generating XML schema using the UML profile of David Carlson [7]. This profile allows for automatic generation of the XML schema using the Eclipse based Hypermodel tool [8]. A modular approach to the construction of a QC schema, similar to that specified in the UN/CEFACT CCTS [9] is under consideration. Here semantically free core components are defined in XML schema that will form the building blocks of quantum chemistry data-types. These core components are of either complex or simple type with simple XSD data-type content. The data-types are then used in further schema to construct information entities. Entities have well defined semantics and are used to construct the QC data model. This highly modular approach makes the data model easily extensible and components re-usable.

2.3 Links and relationships

Some data is not complete without other supplementary data. In a quantum chemistry context an example would be the dependence of molecular orbital coefficients on the atomic basis sets and molecular geometry. It is therefore important to clearly define links and relationships between data. Some relationships are implicit, through the structure of the document. For example, this could be through the ordering or nesting of elements. In addition, XML data-types can be modelled in schema as extensions or restrictions of other types, allowing sub-typing. However, often more complex relationships need to be declared. Of course, it is possible to declare implicit relationships through legislation, but a way of specifying a change of the rules in certain circumstances is required. An example would include the selection of a specific atomic basis set for certain atoms of a molecule, where the basis set is different from that mapped to these atoms through legislation. It is also important to be able to record such mappings and referrals for future references. It is likely that semantic web technologies, such as the resource description framework (RDF) will be used in its XML serialised form to declare such relationships. Another consideration is that QC data will reside in repositories, such as basis set and structure libraries, and the data model must permit data to be standalone in this respect. Therefore, it is difficult to rely on the data structures themselves to highlight such relationships.

It is common practice to separate data and meta-data, primarily to increase the efficiency of meta-data searches. Linking of the data and its meta-data is therefore a further concern. Data is often derived from other data. A simple QC example would be the relaxation of a structure, originally determined through a less expensive calculation; it is important to keep track of data provenance. These links could be modelled in the meta-data schema (as is the case in the CCLRC meta-data model, which supports the concepts of related references, directionality and the granularity of the referenced source). An alternative is to make use of XLink linkbases. These are databases of links between external resources. The XLink specification allows each link to have an ‘arcrole’, which is like an RDF ‘HAS A’ predicate.

2.4 Binary data

Some QC data is not suitable to be formatted in XML, due mainly to the very large documents that would result. It is likely that data structures scaling in size greater than with the square of the number of electrons should have a binary representation, examples would include two electron integrals. If this data is to be transferred between machines then a machine independent way of storing the binary data is required. NetCDF, HDF, BinX or DFDL [10] could be used for this purpose.

3. Architecture

3.1 Data handling

A quantum chemistry calculation requires data, commands, controls and parameters. Some of these are new and required from the user and others can be taken from elsewhere. This could be from a data repository such as an XML database, examples including Xindice or eXist. A meta-data repository could be mined for links to data that match specified criteria. In the current context, this could be for the structure of

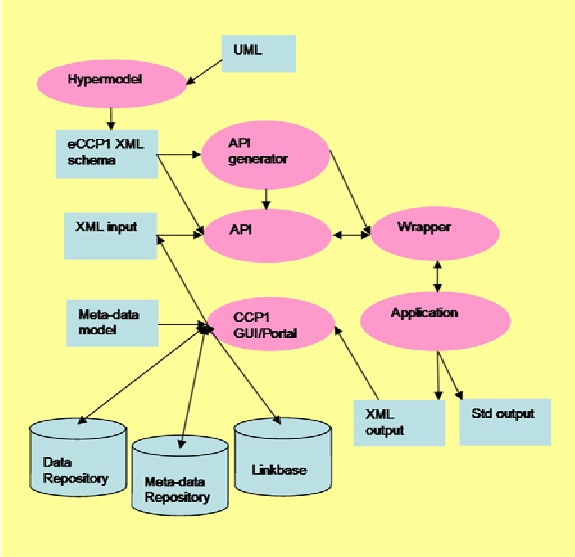

Figure 1. QC data management framework

a molecule or for particular atomic basis sets. A framework is required for both the data mining, input of data from the user and subsequent collation of the data for input into the quantum chemistry application. This collation would require consideration of the relationships between data, such as those between atomic basis sets and atoms. Default relationships could be assumed, with the user having the option to change these from within a data management framework. The development of common GUI/portal front and back ends to QC applications, such as the Python based CCP1 GUI [11], could provide such a framework. The QC code input is to be XML conforming to the eCCP1 data model. This could either be read by the application directly or transformed (for example using XSLT) to standard code input. Many QC applications depend on the Fortran language and traditionally XML support for Fortran has been poor. However, several Fortran APIs now exist. xmlf90 provides a SAX and DOM interface for reading and a 'well formed XML' module for writing [12]. It also includes an XPath inspired module for locating specific parts of a document (SAX-based). An alternative is to write Fortran wrappers that link to foreign APIs for XML parsing, such as the C based libXML or Xerces. This approach has a shortfall in that the parsing of data between codes written in different languages is compiler and platform dependent and any wrapper must consider the possibilities. The upside is that these libraries are often very advanced, heavily used and have a good support base.

A further consideration is that some libraries, such as libXML, have mixed SAX-DOM interfaces for parsing data. A pure SAX interface is useful for quickly reading data with very little overheads, but poor if the data has many links. A DOM interface is useful for heavily referenced data, but the memory overheads are potentially more significant for large XML documents. A mixed interface, such as libXML's xmlTextReader, will allow SAX to, and the construction of a DOM from, an XPath specified node in the XML document.

Data models evolve with time and this can lead to overheads keeping the parser APIs up to date. Automatic creation of APIs from the XML schema is possible and is implemented by the Chemical Markup Language (Java and C++ API generators) and CCPN (Python API generators) [13]. Some work still remains in mapping the wrapper data structures to the internal data structures of the QC application.

Completion of a calculation will result in a new set of data. This data should be formatted in a way that conforms to the eCCP1 data model. If this data is to be stored in repositories, meta-data should be harvested. Some meta-data can be automatically generated by the GUI/portal, knowing the application executed and the details of the job's data, commands, parameters and controls. The input data itself could form part of the meta-data. Other meta-data will be required from the user, being entered for each calculation or common data being read from a file. This would include information about the user, the project, the purpose of the study and so forth. The data and meta-data should then be transferred to the repository and any links updated. From the earlier discussion, these links would include those from the data to meta-data, those that provide pedigree and those to data dependencies. Figure 1. provides an outline of the above processes.

The eCCP1 project is planning a reference implementation of the above framework. This

will include data and meta-data models for key quantum chemistry data-types, APIs and wrappers for the XML parsing and the development of a data management framework, such as an extension of the CCP1 GUI for meta-data mining, data collation, job submission and data and meta-data harvesting and curation. The project is driven by the requirements of the international quantum chemistry community and information is being gathered via meetings such as those at the NeSC [14]. It is planned to start discussions on the eCCP1 TWiki site (http://grids.ac.uk/eccp) and email list (eccp-data@forge.nesc.ac.uk). Tools will be available at (http://forge.nesc.ac.uk/projects/eccp), while data and meta-data models will be available at the eCCP1 TWiki site. These models will appear as UML diagrams and XML schema. Detailed discussions will take place on the mail list and overview documents will be maintained at the TWiki site. These overviews will include details of the components of the models and software tools and issues associated with each.

3.2 Grid Processes

A framework for the handling of a common data and command format for QC applications is an important component of promoting the effective use of Grid resources by the QC community. However, there are a number of other processes that require careful attention. Client tools should be developed that expose Grid resource characteristics and match these to job requirements. Job requirement determination is a complex task, which could involve estimations by the user or passes through a job process. Careful thought has to be given to authorisation and authentication, ensuring that jobs are executed on Grid resources by genuine users that have the required permissions. Grid resources may not have the required applications and their deployment then becomes an issue, along with that of software licensing. The submission of jobs requires the handling of failures and a framework for the management of workflow. Finally, transparent access to the data by local and remote resources is required. This could utilise technologies such as the Storage Resource Broker (SRB).

4. Concluding summary

The development of a common data and command representation for quantum chemistry has clear advantages for the interoperability of applications developed by disparate groups of collaborators. Such a representation promotes the development of common tools that can be used to analyse and visualise data. Other advantages include the ability to intelligently search for data by mining meta-data repositories. Such data, with well defined semantics and meta-data can be appropriately used by a range of applications. Repositories could form convenient libraries of structures and basis sets. As the current trend towards multi-scale, multi-length science continues, a framework as described herein will become an important tool for computational science.

5. References

[1] Environment from the molecular level: an escience testbed project. Martin T Dove, Mark Calleja, Jon Wakelin et al., AHM 2003, ISBN 1-904425-11-9

[2] The Open Babel Project.
http://openbabel.sourceforge.net/

[3] The Crystallographic Information File.
http://www.iucr.org/iucr-top/cif/#docs

[4] The Chemical Markup Language.
http://www.xml-cml.org

[5] The CCLRC Scientific Meta-data Model.
http://www-dienst.rl.ac.uk/library/2002/tr/dltr-2002001.pdf

[6] The eCCP1 TWiki site.
http://grids.ac.uk/eccp

[7] Modelling XML applications with UML, David Carlson, Addison-Wesley 2001, ISBN 0-201-70915-5

[8] Hypermodel Overview.
http://www.xmlmodeling.com/hyperModel/index.html

[9] UN/CEFACT Core Components Technical Specification.
http://www.oasis-open.org/committees/download.php/6232/CEF[ACT-CCTS-Version-2pt01.zip

[10] GGF DFDL Primer.
http://www.ggf.org/Meetings/GGF11/Documents/DFDL_Primer_v2.pdf

[11] The CCP1 GUI.
http://www.cse.clrc.ac.uk/qcg/ccp1gui

[12] An XML parser in Fortran.
http://lcdx00.wm.lc.ehu.es/~wdpgaara/xml/

[13]The CCPN Data Model.
http://www.ccpn.ac.uk/datamodel/datamodel.html

[14] Past eSI Events.
http://www.nesc.ac.uk/esi/past.html