Lessons in scientific data interoperability: XML and the eMinerals project

BY T. O. H. WHITE 1, R. P. BRUIN 1, G.-T. CHIANG 1, M. T. DOVE 1,*, R. P. TYER 2 AND A. M. WALKER 1

1 Department of Earth Sciences, University of Cambridge, Downing Street, Cambridge CB2 3EQ, UK
2 Science and Technology Facilities Council, Daresbury Laboratory, Daresbury Science and Innovation Campus, Warrington WA4 4AD, UK

A collaborative environmental eScience project produces a broad range of data, notable as much for its diversity, in source and format, as its quantity. We find that extensible markup language (XML) and associated technologies are invaluable in managing this deluge of data. We describe FoX, a toolkit for allowing Fortran codes to read and write XML, thus allowing existing scientific tools to be easily re-used in an XML-centric workflow.

Keywords: eScience; eMinerals; Fortran; extensible markup language; chemical markup language; keyhole markup language

1. Introduction

eScience projects can use large amounts of data, and problems often arise as much from the diversity of the data as from their quantity. This is especially the case within environmental sciences, and more so in a cross-disciplinary project. The breadth of diversity characterizing the data can arise in two ways: where fundamentally different quantities are involved or where similar quantities are represented differently.

For example, in a molecular-scale simulation, a classical molecular dynamics code and a chemical-accuracy quantum mechanics code generate data in different formats. These data cannot be trivially reconciled, yet they must be somehow comparable, or we would not be interested in using both. (We might want to compare structures calculated via different methods.) In environmental science, we often need to compare widely different sources of information, e.g. the output of a hydrological model and census data. Obviously, these two data sources have almost nothing in common; yet they are commensurate in their geospatiality.

But data that are, in principle, shareable are rarely provided in a form that encourages this sharing. Collaborative eScience projects are particularly vulnerable—the form and content of the data to be shared across an interdisciplinary interface may be unique to that interface.

* Author for correspondence (mtd10@cam.ac.uk).

One contribution of 24 to a Discussion Meeting Issue ‘The environmental eScience revolution’.
We have found that encoding our data in extensible markup language (the XML), and building XML-based software, has been enormously beneficial in overcoming this problem.

— Many XML technologies are concerned with extracting subsets of data from XML files. Rather than needing to deal with the whole of a given XML dataset, we can work with only the portion of the data which we need.

— The obverse of this is that XML languages lend themselves to extensibility; an XML format designed for representing one type of data can easily be extended to represent another, or even mixed with other XML languages within the same file. Common representations can be chosen for shareable data, while more specialized data can be preserved in an idiomatic form.

— On a purely practical level, XML offers the advantage of a huge range of existing library implementations and toolkits, all of which may be brought to bear when writing tools.

These advantages make it far easier to manage heterogeneous data. Extensible formats, centred about common data, mean that we can encode and extract common portions of data transparently, while faithfully preserving the fullness of more specific, less transferable data.
We have deliberately not mentioned the use of any particular XML language here. Great advantages do arise from the network effect of building on top of an established, community-supported language. Nevertheless, even without that, the advantages listed above render XML a very powerful tool; as an extensible language, it lends itself well to non-standard, domain-specific extensions.

One frequently touted advantage of XML is the availability of schemata for data validation. In practice, though, we have found this to be of negligible use—indeed, frequently, we find the use of schemata more of a hindrance than a help, mitigating directly against the levels of extensibility needed for wholesale data interoperability.

The ability to use ad hoc extensions—in fact, the ability to design and implement XML dialects and languages on the fly, with varying levels of formality—is a great part of the attraction of XML to the eScience worker. Interoperability is enhanced rather than hindered by the ability to embed application-specific data, and to tailor XML dialects to their domain of use.

Regardless of formal schemata, languages conform to implied rules if tools can meaningfully process the data. There are questions as to how these rules might be expressed, and how the expression of these rules should be governed, but there is no expectation that the answers should be the same for every community.

Of course, there can be an overhead to encoding data as XML, in terms of both size and processing time. Direct usage should be restricted to small- or medium-sized datasets—we have successfully used XML processing on the

Figure 2. Visualizing a Greenland ice sheet in three dimensions.
gigabyte scale. However, even at the terabyte scale or beyond, XML can still be of use, given layers of indirection; for example, XML might be applied to stand-off metadata, while leaving the main body of the data as more space-efficient binary formats.

2. FoX

Although a multitude of XML tools exist, one crucial part of the tool chain did not. Many simulation programs are written in Fortran. However, there were no easy ways to interface Fortran programs with XML data. A major outcome of the eMinerals project has been the FoX toolkit (White et al. 2006a). This is a library allowing XML data to be both read and written from within a Fortran program. FoX is written to require as little XML expertise as possible from its users.

The initial target of FoX was the creation of CML documents, and here it has excelled. CML is the chemical markup language, designed for chemically interesting data. We have adapted several major community simulation codes to output CML data, and this approach has gained traction beyond the project; other popular atomistic codes have also adopted this approach. These codes span much of the breadth of atomistic science, from classical molecular dynamics, through solid-state density functional theory, to high-precision quantum chemistry.

Our success here has been, in large part, attributable to the way in which we have been able to play fast and loose with CML, working beyond formal schemata. Echoing our sentiments above, for example, we can encode three-dimensional atomic configurations—universally understood data structures—in a common fashion, while details of code-specific parameters can be stored in code-specific dialects within the CML.

We have subsequently formally adapted and subsetted the CML language to more closely focus on atomistic-level simulation data, and this new dialect, CMLComp, is now implemented in several leading community codes. The support of FoX has been invaluable in this effort, allowing us to supply communities with a prepared library of routines, and requiring little to no buy-in from existing developers.

Since all of these codes can now output data in closely related formats, we have been able to write browsers for CML data (figure 1), allowing users to use the same viewing tool for their atomic-scale data, regardless of its origin (White et al. 2006b).

FoX has also acquired facilities for XML input, as well as output, providing the only fully compliant and fully featured SAX and DOM interfaces available to the Fortran programmer.

Although FoX was initially built within the eMinerals project, and focused primarily on use with CML, it has since gained numerous users outside this area, and has been used within a number of different scientific domains, from glaciology to fire science. Its ability to allow reading of XML files has been particularly valuable in this regard; that it has been so widely adopted is testament to the generality and usability of the APIs it offers.
Keyhole markup language (KML) is a format for geospatial coordinates, as understood by Google Earth (GE), Google Maps and an increasing number of similar tools. The free availability of client tools to visualize KML data has made it a particularly popular target for many users of FoX.

Although specified as a visualization language, it turns out to be very good for storing any data of geospatial interest, precisely because of XML’s extensibility—data in arbitrary XML formats can be stored within a KML document. Using KML as a wrapper can thus add a geospatial dimension to any data format. Although possible earlier, this is explicitly documented and supported as of KML v. 2.2. (It should be noted that KML does not offer support for alternative coordinate reference systems; for high-precision geospatial work, KML will not be suitable.)

In environmental sciences, a common task is to compare geographical aspects of data. Extracting coordinates from different formats and placing them on a map can be difficult. However, where data have been produced directly in KML, the ubiquity of tools such as GE allows immediate viewing of the geospatial aspects of any data. We have adapted several environmentally relevant codes in this way (Chiang et al. 2007); an example is in figure 2.

FoX’s support of KML now makes visualizing arbitrary data on GE trivial for the Fortran-literate computational scientist. This has proved a boon to a very large number of users. As an added advantage, once users’ data have been placed into even an informally specified KML dialect, they are far easier to work with than in any previous incarnation—both for the creators of the data and any third parties.

This ease of manipulation has enabled closer integration of simulation packages with the general job submission process. We have built workflow and metadata-harvesting tools which extract and store data of any sort, via simple XML parsing of output files (Tyer et al. 2007). These give the scientific researcher the ability to meaningfully extract data from hundreds or thousands of individual data files.

3. Conclusion

Cross-disciplinary eScience projects place particular demands on their data-handling infrastructure; we believe XML to be peculiarly well suited to handling this task precisely because of its extensibility.

Fortran programs have been hitherto excluded from this domain owing to the inability to tightly integrate Fortran and XML workflows; however, we have developed a tool, FoX, which permits this. Indeed, it has been sufficiently successful that it is now used well beyond its initial application in the eMinerals project.

We acknowledge support from NERC under the eScience thematic programme.

References


Phil. Trans. R. Soc. A (2009)
