Data Frameworks for Environmental Modelling

R. M. Argent

C. Maul

T. Krämerkämper

Data Frameworks for Environmental Modelling

Robert M. Argent[a], C. Maul[b], T. Krämerkämper[c]

[a] *Centre for Environmental Applied Hydrology; Department of Civil and Environmental Engineering, The University of Melbourne, Australia (R.Argent@unimelb.edu.au)*

[b] *DNRE Tatura, Ferguson Rd, Tatura VIC 3616, Australia (Christian.Maul@nre.vic.gov.au)*

[c] *getit, Ges. fuer Technologie- und Informationstransfer mbH, Emil-Figge-Str 76-80, 44227 Dortmund, Germany (kraemerkaemper@getit.de)*

Abstract: Developers of environmental modelling software are adopting modern software engineering approaches that use a combination of model components to provide appropriate modelling services. One of the key issues in this development is easy and reliable storage, exchange and display of data, often using distributed sources and tools. This, in turn, requires standard methods for providing and describing data. Extensible Markup Language (XML) is a set of standards and protocols that has been adopted by commerce and industry for exchanging and displaying data, and it also provides a metadata standard that can be used to specify interfaces and data requirements for model components. Associated with the adoption of XML is the development of a broad range of open source software for XML parsing, processing, and search or query functions. Parsing involves analysing an input stream of XML data and determining structure and context of elements, tags and attributes, whereas processing is used to transform XML documents into something that can be displayed, such as HTML or XHTML. The paper describes a data framework for environmental modelling. The framework uses XML-based parsing and processing to form the input and output filters for data delivery to model components, or to on-screen display. In this system, data are stored in an SQL-compliant DBMS, an object-oriented database, or in a system that stores XML directly or possesses an interface to XML. Implementation of the data framework is made in a fashion similar to Java Enterprise Beans which use XML to connect a program (a Bean) and a container. The container provides services such as load distribution, transaction monitoring and connection to the remote interfaces. The additional services provided by the data framework include structure of records, comprehensiveness of data, origin and time. In conjunction with open source web servers, use of the XML standard and the open source software available provide an inexpensive, flexible, central data repository for component-based environmental modelling.

Keywords: Extensible Markup Language, XML, Data Framework, Component-Based Modelling

1. INTRODUCTION

Environmental modelling practice is slowly moving towards broader adoption of good software engineering practices as researchers, developers and model users realise the benefits that accrue from adoption of these. Component-based approaches are becoming more popular as components for standard routines, such as analysis, input and output, become increasingly available, allowing developers to concentrate on development of core components. In the area of integrated modelling for environmental management, this approach has the added benefit of allowing scientists from different disciplines to focus on how the component they use to represent one part of the system works with other components.

Despite this practical and conceptual shift towards component-based *modelling* of environmental sub-systems, there has not been a similar shift in the equally important area of management of the environmental data that supports our modelling. One of the key requirements in environmental modelling is easy and reliable storage, exchange and display of data. This, in turn, requires standard methods for providing and describing data.

Use of the Extensible Markup Language (XML) is becoming more widespread in various areas of environmental modelling and data management. Such uses include data analysis [Aloisio *et al.*, 1999], storage and description [Kokkonen *et al.*, 2001], and model-data connectivity and interfacing [Rana *et al.*, 2000; Rizzoli *et al.*, 2001].

Open source tools for XML-based data management are becoming widely available, arising primarily from business and commerce areas, but slowly finding positions in ecological, hydrological and other environmental system modelling.

This paper takes up some of these ideas and pursues further the possible uses of open source tools for data management and handling in environmental modelling and simulation. Some of the practical limitations to adoption of improved data management techniques, primarily related to institutional requirements, are also discussed.

2. DATA MANAGEMENT OPTIONS AND REQUIREMENTS

Accessing and collating data for use by models, or by model components, is often an exacting, difficult, and time consuming task. Much work has been done to advance modelling concepts and practice over the past few years [for example Bian, 2000; Villa and Costanza, 2000; Argent *et al.*, 2001; Villa, 2001], but leaps in data management have not taken place. Many simulation modelling exercises now take advantage of component-based modelling approaches, but an increasingly large amount of project time is spent on data collation and manipulation, particularly with advanced integrated models that extract data from many sources, each with their own unique data storage method.

In environmental modelling there have been a progression of data access methods used, and many are still in use. The following describes some of these:

- Data are stored in a text file in a specific format. The model specifically loads the data set into single or array variables. Use of the data by the model is then done by access to a specific variable. e.g. Rainfall(3,11)
- Data are stored in a text file that is registered with the model as a specific data type. Data is then accessed by the model by calling the data, e.g. RainfallSet.get(StationID,Month)
- A single data source with XML-based metadata is used to supply various data needs for various model components. Each data requirement is uniquely specified using XML via descriptive aids such as a Document Type Definition (DTD) or XML schema.

In this third system there are a number of advantages that suggest more flexible methods for data storage, searching, access and visualisation than with alternative methods. This is of particular interest in an environment where increasing amounts of data are becoming available from remote data stores, and where systems may need to not only access and utilise data, but also seek out which data are available to meet some currently defined need.

In this paper we examine the third approach listed above and examine the system requirements, architecture and data manager changes that would need to be made for this to work, as well as some of the advantages and disadvantages of such a system.

3. DATA ACCESS AND USE FOR ENVIRONMENTAL MODELLING

In a situation where data relevant to some modelling problem are available from various alternative sources, the functions required of a data access and use system are:

- Analyse and/or standardize data description, i.e. that “rain mm per day”, “daily rain” or “mm daily precipitation” probably describe the same data set.
- Identify what data are available, where they are, and how they are obtained.
- Filter or translate a data request to identify which data set is appropriate to the request.
- Retrieve the relevant data.
- Filter or interpret the retrieved data into a form that is appropriate to the model component that is the end user of the data
- Manipulate data (such as filling gaps) to suit specific model requirements.

With the right tools and appropriate use of XML all of these needs could be provided. The consequences of such an approach are improved data access, less time and effort spent on data collation, more efficient model use, more flexible combination of model components, and exchange of data between model users.

4. XML AND ITS BENEFITS

Extensible Markup Language (XML) is a set of standards and protocols that has been adopted by commerce and industry for exchanging and displaying data, and it also provides a metadata standard that can be used to specify interfaces and data requirements for model components.

The advantages of XML in environmental modelling and simulation exist in all aspects from data storage and access to model-data matching and interface definition [Aloisio *et al.*, 1999; Rana *et al.*, 2000; Kokkonen *et al.*, 2001; Rizzoli *et al.*, 2001]. One of the possibilities that is offered is the ability for data to be described in such a way that different models can query a data source for different data from the same source.

XML is a sub-language of the Standard Generalised Markup Language (SGML) that has been simplified (by eliminating various SGML options and features) so that it is easier to work with, and build tools for, than SGML. It is not a fixed format, like HTML, but rather is *eXtensible* by allowing authors to define language parts (elements) to suit particular needs [Flynn, 2001; W3C, 2001].

There are a number of tools and features associated with XML that lend themselves to providing the features listed in Section 3.

4.1 XML Tools

There are three common tools associated with use of XML – parsers, processors and query tools. Parsing involves analysing an input stream of XML data and determining structure and context of elements, tags and attributes. Processing is used to transform XML documents into something that can be displayed, such as HTML or XHTML. Querying is similar to querying in standard database management, using an approach such as SQL, but in the case of XML, the query is XML based.

4.2 Elements

Elements are one of the key features of XML, and provide considerable flexibility in enabling discipline groups to consistently describe data. A simple element familiar to users of HTML is a 'list' containing 'items'. For environmental modelling, a situation-specific element could be a 'StationList' containing any number of 'Stations'. Elements can be used hierarchically, with any parent element enclosing an indefinite number of child elements.

4.3 Document Type Definition (DTD)

One of the interesting aspects of XML in the realm of data management is the ability to define 'valid' document types. It is possible to use XML without a DTD, provided the XML document is 'well formed' – in that it follows the rules of XML, such as having start and end tags for every element. XML parsers can be used to check if a document is well formed. In order to ensure that an XML document is 'valid' the parser needs a DTD. A DTD serves three basic purposes:

- It documents the elements.
- It enforces compliance of the XML document with these elements.
- It enables the parser to validate the document.

An example of a DTD that is in common use is the British Weather Observation Format (OMF) [Strajnar, 2000; FNMOC, 2001].

4.4 Schemas

An alternative to the DTD is an XML Schema [Flynn, 2001; Rizzoli *et al.*, 2001]. DTDs have no mechanisms for defining data types, so a DTD cannot be used to specify numeric ranges, define limitations or to check text content A Schema provides a means of specifying element content in terms of data types, so that there are criteria for validating the content of elements, as well as the markup itself. An advantage of Schemas is that they are written as XML files.

An XML schema (IXRetail) for the retail sector [Golick and Mader, 2002] has been endorsed by the American National Retail Federation and the Association for Retail Technology Standards. This Schema helps retailers to interface point of sale data collection and back office order, logistics and billing applications. In the same fashion, an XML hydrology data schema could be generated that defines conventions for variable naming and units.

5. A DATA FRAMEWORK

Associated with the adoption of XML is the development of a broad range of open source software for XML parsing, processing, and search or query functions. These tools, along with data storage and distributed computing tools, can be used is the construction of a data framework for environmental modelling. The framework uses XML-based parsing and processing to form the input and output filters for data delivery to model components, or to on-screen display.

5.1 Storage

Data storage can be an SQL-compliant DBMS, an object-oriented database, or a system that stores XML directly or possesses an interface to XML. We do not consider direct XML-based storage as an elegant solution, as tags would be stored for each and every data value. However, large database application vendors, such as Oracle, have released or are releasing systems that do this.

Generally, any relational database can work with XML, providing methods to map between XML documents and stored objects. Sun is working to provide tools and API's that automate the mapping between XML documents and Java objects. JAXB is the API that is aimed at making XML easy to use by compiling an XML Schema into one or more Java classes. The generated classes handle all the details of XML parsing and formatting. Similarly, the generated classes ensure that the constraints expressed in the Schema are enforced in the methods and Java data types. Currently only DTDs are supported but the support of XML Schemas is planned. Alternatively, there are middle-ware tools available, such as XML-DBMS (http://www.rpbourret.com/xmldbms/) that can be used for specific mapping of XML documents to objects.

5.2 Distributed Computing

Data delivery from remote sources can be achieved with applications that use something simple such as the Simple Object Access Protocol (SOAP) (Figure 1). SOAP is a lightweight protocol for information exchange in a distributed environment. Lightweight means that the SOAP applications involved do not know and do not care about language, protocol host or operating system. It uses an XML based protocol that consists of three parts: i) an envelope that defines a framework for describing contents of a message and how to process it; ii) a set of encoding rules for expressing instances of application-defined datatypes, and iii) a convention for representing remote procedure calls and responses (http://xml.apache.org/soap/). SOAP clients and services can be written in a variety of languages such as Java, Perl, or C#. As a stateless protocol, SOAP does not have a memory of past transactions. This could limit use in environmental modelling where users might want to cascade models and use the output from one model as input for another, or start part of the modelling process again from a certain previous point.

Implementing state aware, persistent computing via the internet requires transaction monitors. The broadest range of tools, standards and API's is provided by Sun's Java 2 Enterprise Edition (J2EE) platform. Web servers such as JBoss (http://www.jboss.org/), Tomcat (http://jakarta.apache.org/tomcat), or JOnAS (http://www.evidian.com/jonas) provide an environment for server side programs such as servlets, and data or database connection. They are used for persistence management, security management, load distribution, messaging services and transaction management. In the simplest case it handles requests, running servlets or Java Server Pages, similar to use of a SOAP application. However, for truly distributed, persistent transactional computing, programming models in the J2EE fashion is required. This comprises a home interface that creates Enterprise Java Bean (EJB) objects, a remote Bean interface that defines methods the EJB object exports to remote clients, and the Bean implementation itself which would contain the model. This landscape would require Java as the implementation language for all models or at least a Java wrapper around the model. However, it seems preferable at the current time to use SOAP in the first instance, and to advise users to download required resources and operate models locally.

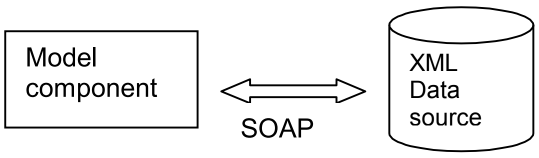

Figure 1. SOAP used for exchange between a model component and a remote data source

5.3 Parsing

Parsing involves analysing an input stream of XML data and determining structure and context of elements, tags and attributes (Figure 2). There are different approaches to parsing data effectively and different APIs are available. Tools and approaches are rapidly trending to simple document based APIs, such as JDOM, and the open source movement is providing a wide range of parsers and processors. Crimson or Xerces (http://xml.apache.org/) is one open-source tool that can be used to parse XML data. JAXP (Java API for XML Parsing) enables applications to parse and transform XML documents independent of a particular processing implementation for

XML. DOM supports processing of XML documents using DOM, SAX (Simple API for XML) and XSLT (Extensible Stylesheet Language Transformations). A similar, but independent, project is JDOM (http://www.jdom.org/) that aims to provide functionality using an approach that is simpler and easier than in SAX.

5.4 Processing

XML parsers and processors usually come in pairs. The corresponding XML processor to Xerces is Xalan (http://xml.apache.org/xalan-j/). The processor can be used to transform XML documents into something that can be displayed such as HTML or XHTML (Figure 2). Both could form the input and output filters that move data into model components or into a displayable and manipulable form on the screen. In this domain it may be advisable to resort to commercial applications as they are prevalent. Tidestone, for example, provides a spreadsheet component that can read and write Excel™ and generate whichever output format the user wants. Building on existing user knowledge, users could generate an XML data file and DTD, or XML Schema, and store it in a database [Maul, 2001].

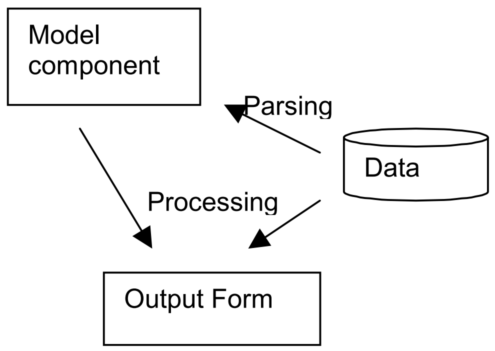

Figure 2. Use of parsers and processors to form input and output filters

5.5 Querying

Querying can be done in a number of ways. Because of the tree structure of XML documents the Xpath (XML) path language can be used to traverse the tree structure of an XML document and to extract particular portions of it.

There is also a more SQL-like approach. Currently XML Query specification, XQL, is in the very early stages but one early implementation is available from Gesellschaft für Mathematik und Datenverarbeitung, the GMD-IPSI XQL Engine (http://xml.darmstadt.gmd.de/xql/). It is thought that further tools will become available as use of XML in environmental modelling and data management increases.

5.6 Data Framework

Putting together the tools described above produces a data framework for environmental modelling. One working form of such as system could proceed as follows:

- A model is constructed using one or more model components. This model requires data.
- Data are stored in a range of data sources of different kinds, both locally and remotely. A metadata system is used to keep track of what data are available.
- A DTD or Schema exists, and is used to ensure compliance between the data required by a model component and the available data.
- A data query is generated by the model, and data are matched to the query.
- Data are retrieved and parsed for use by the model.
- The model uses the data and the output, and/or original data, are processed for external use.

6. IMPLEMENTATION, IMPLICATIONS AND THE FUTURE

The proposed data framework, or alternatives with or without various of the framework sub-components, appears to be straightforward, but considerable developments are required before it comes to fruition. First, the data request and data description must be matched. Ideally this would be done by adoption of standard descriptors or standard language for description of data types. In some areas, such as geographic data, such standards, and particularly metadata standards exist, but the standards for attaching metadata to data in such as way that a query from a model could find the right data, are less common.

This is some incentive for data suppliers to provide XML-based metadata or data in XML format, particularly for their own use for interactive querying either locally or remotely. Already, some web-based data services are proving XML formatted data. These data therefore only require a DTD or Schema to allow validation and mapping of available data to queries. This is good for modern data. However, there are still many data services where archival data extraction is performed using a non-XML system, and data are produced in flat text files with little or no attached metadata. The interim measure for this connectivity would be the use of data mapping approaches that link the data that are being requested with the data types and format that are supplied. Over time, and as standards develop, data suppliers should be encouraged to increase the quality and quantity of attached metadata.

For modelling, the framework suggested here also relies on developers building model components that use XML data queries. This is not necessarily as difficult to make happen as it appears, as modellers who are advanced enough in their programming to be working with component-based approaches will probably handle the integration of XML queries without difficulty. Support of XML in modelling environments, such as .NET, will also increase use of XML in modelling.

Adoption of some of the approaches suggested here can be seen as another step in the general improvement of modelling practices used by the environmental modelling community. A difficulty with implementation is that not only must modellers improve their practices, but so also must data service providers. Given the strong interdependence of the two groups, steps must be taken to find mutually acceptable processes for solving our model-data interaction problems.

7. ACKNOWLEDGEMENTS

The authors wish to acknowledge the support of the Cooperative Research Centre for Catchment Hydrology.

8. REFERENCES

Aloisio, G., G. Milillo and R. D. Williams, An XML architecture for high-preformance web-based analysis of remote-sesnsing archives. Future Generations Computer Systems, 16 (1), 91-100, 1999.

Argent, R. M., R. A. Vertessy, J. Rahman and S. Seaton, From components to decisions: The role of software engineering and frameworks in catchment decision support. In: Ghassemi, F., D. White, S. Cuddy and T. Nakanishi, (Eds.), MODSIM 2001 International Congress on Modelling and Simulation: Canberra, Modelling and Simulation Society of Australia and New Zealand Inc., p. 1589-1594, 2001.

Bian, L., Component modeling for the spatial representation of wildlife movements. Journal of Environmental Management, 59 (August), 235-245, 2000.

Flynn, P., The XML FAQ. http://www.ucc.ie/xml/, 2001.

FNMOC, Weather Observation Markup Format (OMF), Navy Fleet Numerical Meteorological and Oceanography Center. www.xml.org/xml/zapthink/std184.html, 2001.

Golick, P. and R. Mader, Good naming conventions extend beyond retail. http://www106.ibm.com/developerworks/library/x-retail.html, 2002.

Kokkonen, T., A. Jolma and H. Koivusalo, Interfacing environmental simulation models with databases using XML. In: Ghassemi, F., D. White, S. Cuddy and T. Nakanishi, (Eds.), MODSIM 2001 International Congress on Modelling and Simulation: Canberra, Modelling and Simulation Society of Australia and New Zealand Inc., p. 1643-1648, 2001.

Maul, C. R., Data presentation, structures, formats, and scalability of applications - How do they fit together? In: Ghassemi, F., D. White, S. Cuddy and T. Nakanishi, (Eds.), MODSIM 2001 International Congress on Modelling and Simulation: Canberra, Modelling and Simulation Society of Australia and New Zealand Inc., p. 1655-1658, 2001.

Rana, O. F., M. Z. Li, D. W. Walker and M. Shields, An XML based component model for generating scientific applications and performing large scale simulations in a meta-computing environment. Generative and Component-Based Software Engineering, Proceedings. Lecture Notes in Computer Science, 1799, 210-224, 2000.

Rizzoli, A. E., R. M. Argent, M. Manglaviti and M. Mutti, Encapsulating environmental models and data using Java and XML. In: Ghassemi, F., D. White, S. Cuddy and T. Nakanishi, (Eds.), MODSIM 2001 International Congress on Modelling and Simulation: Canberra, Modelling and Simulation Society of Australia and New Zealand Inc., p. 1649-1654, 2001.

Strajnar, U., XML based visualisation of meteorological data, Proceedings of the 11th EGOWS Meeting: Helsinki, Finland, p. 80-86, 2000.

Villa, F., Integrating modelling architecture: a declarative framework for multi-paradigm, multi-scale ecological modelling. Ecological Modelling, 137 (1), 23-42, 2001.

Villa, F. and R. Costanza, Design of multi-paradigm integrating modelling tools for ecological research. Environmental Modelling & Software, 15, 169-177, 2000.

W3C, Extensible Markup Language (XML) 1.0, World Wide Web Consortium. http://www.w3.org/TR/2000/REC-xml-20001006, 2001.