Available online at www.sciencedirect.com
SCIENCE DIRECT®
Environmental Modelling & Software 18 (2003) 463–471
Environmental Modelling & Software
www.elsevier.com/locate/envsoft

Interfacing environmental simulation models and databases using XML

T. Kokkonen *, A. Jolma, H. Koivusalo

Laboratory of Water Resources, Helsinki University of Technology, P.O. Box 5300, FIN-02015 HUT, Finland

Received 21 June 2002; received in revised form 18 October 2002; accepted 22 January 2003

Abstract

Typically in environmental management tasks one needs to examine and explore data from several sources, use simulation models, develop scenarios, assess impacts, and provide support for decision makers. Here we consider the eXtensible Markup Language (XML) standard in developing information transfer techniques between databases and simulation models. Suitability of XML as the agreed data transfer format is studied in a sample application, where two snow models of different complexity are linked with input data extracted from a relational database. The simple case study demonstrated that with free and easily accessible tools it was relatively straightforward to develop an XML interface between a meteorological data set and simulation models. Based on the case study, a structure for a more comprehensive system comprising model and data resources, and a broker application that acts as an intermediate between the user and those resources, is presented. We believe that such an XML-based structure is worth exploring on the track towards an open modelling framework. Such a framework would allow models developed by various expert groups to connect easily to a common information system.

Keywords: eXtensible markup language; Interoperability; Database; Information transfer; Modelling

1. Introduction

1.1. Motivation

Environmental problems are typically of a multidisciplinary nature, and may affect broad regions, requiring the analyst to examine data from several sources and to apply computation and assessment tools provided by various expert groups. Difficulty in linking data sets and analysis tools is one of the barriers to be overcome in developing integrated water resources management techniques (e.g. Huang and Xia, 2001). Consider assessing deterioration of a river habitat in response to changes occurring in the upstream part of the catchment. The analyst needs to take into account interactions among soil–vegetation–atmosphere processes and routing of water through the stream network, chemical processes affecting the water quality, and biological processes controlling the well-being of the habitat. Simulation tools and data may be available to address isolated parts of the above problem, but often they are scattered among numerous sources and exist in heterogeneous formats. Development of modelling tools and acquisition of data sets often deal with relatively specific problems and this leads to troublesome model and data transferability. Also, the physical boundary of a catchment rarely coincides with political regions, which easily leads to variable documentation formats, and creates the need for methods catering for communication between many administrative entities such as provinces and countries (Voinov and Costanza, 1999). Indeed, one-third of rivers pass through more than one country (Jones, 1997).

Parker et al. (2002) summarise deliberations of forty-five scientists dealing with integrated assessment and modelling in environmental problems, and they identified communication to be the critical factor in the success or failure of integrated studies. Clearly, agreement on techniques to transfer information between data sources and models is one of the key issues in integrated management of complex environmental problems.

* Corresponding author. Tel.: +358-9-451-3838; fax: +358-9-451-3827.
E-mail address: teemu.kokkonen@hut.fi (T. Kokkonen).

1364-8152/03/$ - see front matter © 2003 Elsevier Science Ltd. All rights reserved.
doi:10.1016/S1364-8152(03)00020-3

1.2. Integration in environmental modelling

To achieve integration, simulation models and engines for data retrieval need to be interoperable. This is a property frequently referred to in the literature but lacking a single, precise definition (Brandmeyer and Karimi, 2000).

Howie et al. (1996) address software interoperability and state that interoperability has usually been defined as "the capability with which two or more programs can share and process information irrespective of their implementation language and platform". Edwards et al. (2000) see interoperability—in the context of biodiversity information—to mean compatibility between databases allowing them to be simultaneously searchable. Goodchild et al. (1997), who discuss interoperating Geographical Information Systems, classify issues related to interoperability to three distinct levels. First, the focus is on technical issues—like format compatibility—which are required to enable data transfer from one system to another. Openness in the software industry, e.g. publication of internal data structures, allows users to build applications that integrate software components from different developers. Secondly, for semantic interoperability it is not sufficient that the data are transferred across systems intact but the data also need to be meaningful—and have the same meaning—for users of both systems. For example, a temperature value alone is not enough—the unit needs to be specified, too. Finally, the institutional level of interoperability, which can be the most problematic one, refers to the willingness to achieve interoperational goals. It is not clear that sharing data and methods is always desirable. Economic considerations may obstruct; the added cost in achieving interoperability may not be profitable, or the owner of the data wishes to charge for it, which complicates sharing of the data. Furthermore, legal and privacy issues can also hamper achievement of interoperational goals at the institutional level.

Integration of simulation models has been implemented in many environmental assessment studies. Aspinall and Pearson (2000) and Huang et al. (1999) developed integrated catchment assessment systems around specific GIS software packages. Booty et al. (2001) describe an environmental decision support system which runs on the Windows platform, integrates data from multiple sources, and provides a selection of decision aiding tools. The modular design of their system allows for flexibility in adding components to the system to meet the demands of a particular application. Models can be incorporated into the system as executable binary codes by supplying an appropriate interface for the model input and output, or through conversion of the model source code to a programming language supported by the system. Papers by Bian (2000) and Rizzoli et al. (1998) also discuss modular structure in the context of environmental simulation systems. In these studies—rather than finding ways to incorporate existing software into the system—the focus is on designing new systems in such a manner that subcomponents of the system would be reusable and thus adoptable to other similar modelling projects. Such ideas have been implemented in the Interactive Component Modelling System (ICMS)—earlier known as the Integrated Catchment Management System—that was developed 'to facilitate the rapid development and delivery of catchment science to catchment managers' (http://www.cbr.clw.csiro.au/icms/). ICMS is a Windows-based software product, within which data and models can be combined together and simulation results can be visualised and further analysed.

Computation frameworks particularly designed for handling different models—with minimal effort required for the integration—are sometimes called open modelling engines (Reed et al., 1999) or open modelling systems (Blind et al., 2001; Cate et al., 1998). Ideally, the user can select the best combination of available modelling components for a particular problem at hand. Reed et al. (1999) promote open modelling by introducing a programming environment for developing and storing models in such a manner that they can easily be reused and integrated with each other. In the open modelling system of Blind et al. (2001), on the other hand, the objective is to bring existing legacy models together. Using the term interoperability discussed above, Reed et al. (1999) aim at developing interoperable models from the outset, while the objective of Blind et al. (2001) is to establish interoperability between existing, incompatible models.

Communication between model and data resources and the model user is crucial in the concept of the open modelling framework. The Internet—being an efficient means of communication—provides an attractive platform also in environmental studies to link models and data sets with each other. In recent years the accelerating rate at which information is transferred via the Internet has challenged service providers to adopt compatible techniques for information transfer. Such techniques need to have a widely accepted, standardised form, they must be universally accessible, and preferably they should be vendor-independent. Compatibility and reliability of data transfer are vitally important for business applications servicing a large number of clients and requiring access to extensive data resources. Example applications include web-based shopping and banking services, flight reservation services, and browsing of library databases. The World Wide Web Consortium (W3C, http://www.w3.org) was established in 1994 to develop, accept and maintain interoperable technologies (specifications, guidelines, software, and tools) to service the Internet community.

1.3. Contribution of the present study

The aim of this study is to explore how one can move towards an open modelling framework by adopting an agreed method that enables a simulation model to document and communicate its input requirements, and receive the input from information systems over the Internet. Existing, standardised techniques adopted within the Internet for information transfer have substantial potential to facilitate communication between environmental simulation models and data sets.

The present paper applies the eXtensible Markup Language (XML), which is a widely used format for structured documents and data on the web, for specifying the semantics of the model input and output. Similar ideas have been recently put forward in the works of Kokkonen et al. (2001) and Rizzoli et al. (2001). This study is an extension of the former paper, which deals with linking databases to the model inputs. The latter paper addresses separation of model interfaces from their actual implementations.

A simple case study is presented to demonstrate that it is relatively straightforward to develop an XML interface between a meteorological data set and environmental models. Data stored in a relational database are transferred to an XML document, which is then fed to a simulation model. Linkage between the XML document containing the model input and the database schema is accomplished with the help of a map file, which binds elements and attributes of XML input to the tables and columns in a relational database. Coupling models—capable of interpreting XML formatted input—with data sources via the Internet constitutes one avenue to an open modelling framework.

The paper concludes with discussion on extending the developed methodology to implement more complex and realistic modelling set-ups than what the presented case study represents. The envisaged, extended framework builds on a broker module, which is a client to servers providing model and data resources. The broker application binds the selected model with the necessary input data and enables thereby a simulation to be run.

2. eXtensible markup language

Lack of adequate model documentation is one of the serious bottlenecks in utilisation and further development of simulation models (e.g. Hoch et al., 1998; Tiktak and Vangrinsven, 1995). Concise documentation of the model input requirements is particularly critical when generation of the model input file from data stored in a database is automated. As motivated in Section 1, XML is an approved format for representing structured data on the web and it has been applied in the current study for establishing the link between data and model resources.

Development of XML started in 1996 and it was adopted as a W3C standard in February 1998. Although XML as its own standard is fairly new, the methodology has a solid background in the standard generalised markup language (SGML, see http://xml.coverpages.org/sgml.html), which was developed in the early 1980s, became an ISO standard in 1986, and has been widely used for large documentation projects. The designers of XML adopted the best parts of SGML, guided by the experience gained with development of the hypertext markup language (HTML), and produced a markup language whose expressive power almost matches that of SGML, but is more regular and simpler to use.

XML is a text-based markup language for describing what the datum is, rather than just specifying how to display it, as is the case with HTML. Similar to HTML, markup is composed of a set of tags, i.e. identifiers enclosed in angle brackets. The tag, its matching end tag, and the parsed character data (PCDATA) between the tags form an element of the XML data. An element can contain child elements allowing XML to represent hierarchical data structures. Additional information can be provided in an attribute, which is a qualifier within a tag. A document type definition (DTD) can be used to define the structure of an XML document. A DTD lists the set of legal element names, specifies the hierarchy of the elements, and gives a listing of allowed attribute names. With a DTD different parties can agree on a particular XML application for data exchange. A validating parser can verify that received XML documents conform to the structure defined in the DTD. This saves much effort from application developers who can rely on the fact that certain elements, such as the units of hydrometeorological variables, appear in the document.

The following three features of XML have contributed to its increasing acceptance in web-based applications requiring standard data transfer or storage formats. First, the markup tags identify the information and break up the data into logical parts. Thus, the same XML data document can be fed to different applications, which then can extract the relevant parts of the data from the document for further processing. Secondly, as a plain text markup language XML is platform-independent, which means that application development is not tied to certain software vendors. And thirdly, as a W3C technology XML is license-free and well-supported. There are commercial and freeware tools readily available for processing XML documents, and an active Internet community of developers can provide assistance. The present study takes advantage of the XML-DBMS (Bourret, 2003) middleware developed for transferring data between XML documents and relational databases. It views an XML document as a hierarchical tree of data-

specific objects and links these objects to a relational database using an XML-based mapping language. XML-DBMS is freely available at http://www.rpbourret.com/xmldbms/ as Java and Perl versions.

3. Sample application

In this section an application, which links simulation models to data using XML, is described. The application involves two snowmelt models, which have different process descriptions and different data requirements. The simpler model is based on the degree-day concept and it needs only daily precipitation and air temperature as input data. The other snow model (Koivusalo et al., 2001) is based on the energy balance approach and is much more demanding in its input data requirements. The energy balance model is driven by precipitation, air temperature, relative humidity, wind speed, and downward short- and long-wave radiation. In addition to the data input, both models require parameter values to be prescribed and given as parameter input to the models prior to a simulation. The degree-day model yields water equivalent and liquid water content of snow as a simulation output, and the energy balance model produces both snow mass and energy state variables as output. In this application the focus is on data transfer methodology, and the models do not contain any computational procedures. The meteorological data are from a station in Siuntio located in southern Finland within 50 km from Helsinki. The meteorological time-series, and additional site and data-specific information (metadata), are stored in a PostgreSQL (http://www.postgresql.org) relational database.

Fig. 1 shows an illustration of the linkage between the database and the snow models as perceived in the sample application. All input to the models and output from the models are directed through an interface, which accepts only XML. To create a simulation run, the user chooses via a web browser which model is to be run and which data are fed to the selected model. Each model available in the system has an associated DTD file, which specifies the required data input and its structure. Fig. 2(a) shows a snapshot of the user interface for selecting the model, and Fig. 2(b) and (c) depicts DTDs describing data input requirements for the selected model. After the model selection has been accomplished, the user informs the system where the requested variables reside in the database. This information and the DTD file are then used to generate automatically an XML-DBMS map file. The map file describes linkage between structure of the XML document, which is required as an input to the model, and the database schema. A utility application included in the XML-DBMS package for creating map file templates from DTD files was modified to construct the map

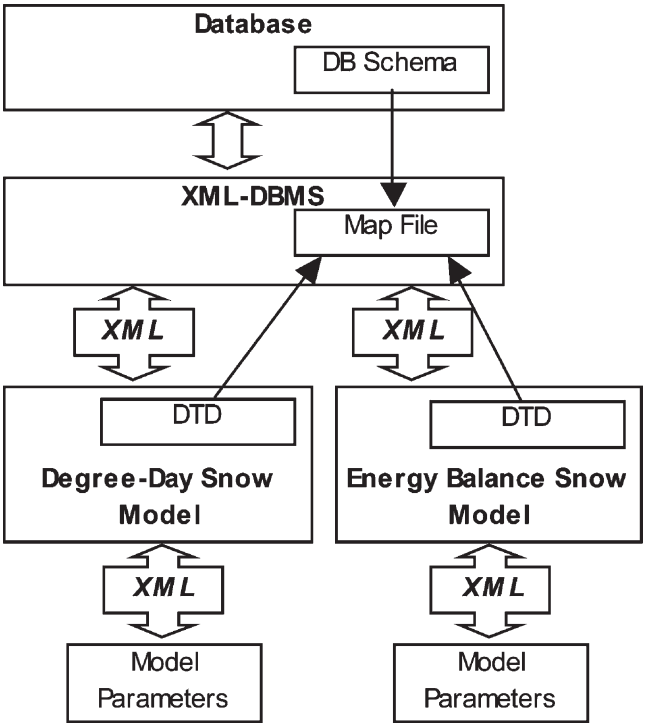


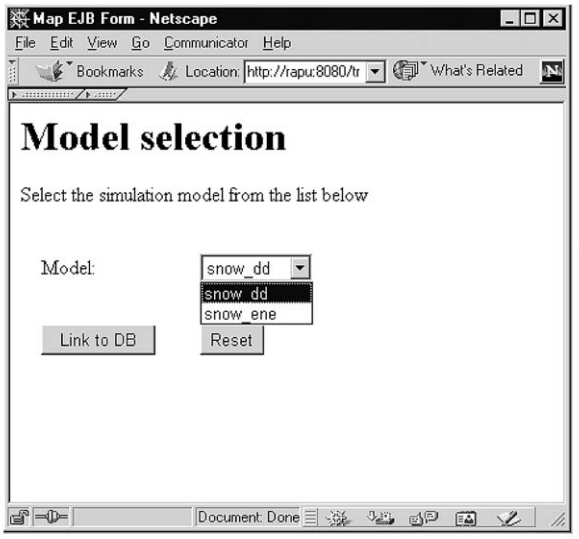
Fig. 1. Schematic representation of the linkage between database and snow models.

files required in the current sample application. Fig. 3(a) shows the map file for the degree-day model; the map file for the energy balance model (not shown) has a similar structure but is much larger due to the greater number of input variables. Fig. 3(b) depicts a snapshot of the user interface for specifying linkage between column and table names in the relational database and structure of the XML data input. Schematic describing how the data are stored in the relational database is shown in Fig. 3(c). Finally, the user selects via the web interface the site and the simulation period (see snapshot in Fig. 4). This completes the information required to retrieve the model input from the database.

The produced XML data file contains relevant time-series required to drive the selected model, and site-specific metadata such as site name and units of the input variables (Fig. 5). A separate XML input file could be created for user specified model parameters. An XML parser embedded in the simulation model extracts information required in the model execution from the input files. The output from a model is an XML document whose contents can be stored in the database through an XML-DBMS interface.

The application was developed using Xerces Java Parser, which is freely available from http://xml.apache.org. The Java version 1.0 of the XML-DBMS software was used. The PostgreSQL database was accessed using the Java database connection (JDBC) protocol over an Internet connection.

(a)



(b)

```
<!ELEMENT Site (Record+,AirTempUnit,
                PrecipUnit,SiteName)>
<!ELEMENT Record (AirTemp,Precip,Date)>
<!ELEMENT AirTemp (#PCDATA)>
<!ELEMENT Precip (#PCDATA)>
<!ELEMENT Date (#PCDATA)>
<!ELEMENT AirTempUnit (#PCDATA)>
<!ELEMENT PrecipUnit (#PCDATA)>
<!ELEMENT SiteName (#PCDATA)>
<!ATTLIST Site SiteCode CDATA #REQUIRED>
<!ATTLIST Record RecordNumber CDATA #REQUIRED>
```

(c)

```
<!ELEMENT Site (Record+,AirTempUnit,PrecipUnit,
                SWDownUnit,LWDownUnit,
                WindSpeedUnit,RHUnit,SiteName)>
<!ELEMENT Record (AirTemp,Precip,SWDown,
                  LWDown,WindSpeed,RH,Date)>
<!ELEMENT AirTemp (#PCDATA)>
<!ELEMENT Precip (#PCDATA)>
<!ELEMENT SWDown (#PCDATA)>
<!ELEMENT LWDown (#PCDATA)>
<!ELEMENT WindSpeed (#PCDATA)>
<!ELEMENT RH (#PCDATA)>
<!ELEMENT Date (#PCDATA)>
<!ELEMENT AirTempUnit (#PCDATA)>
<!ELEMENT PrecipUnit (#PCDATA)>
<!ELEMENT SWDownUnit (#PCDATA)>
<!ELEMENT LWDownUnit (#PCDATA)>
<!ELEMENT WindSpeedUnit (#PCDATA)>
<!ELEMENT RHUnit (#PCDATA)>
<!ELEMENT SiteName (#PCDATA)>
<!ATTLIST Site SiteCode CDATA #REQUIRED>
<!ATTLIST Record RecordNumber CDATA #REQUIRED>
```

Fig. 2. Snapshot of the user interface for selecting a model (a), and DTD files specifying data input requirements for degree-day (b), and energy balance snow models (c).

4. Discussion

4.1. Current results

The simple case study above demonstrated that with free and easily accessible tools it was relatively straightforward to develop an XML interface between a meteorological data set and two snow models of different complexity. XML-DBMS middleware (Bourret, 2003) was used to transfer data stored in a relational database to an XML document, which passes the input to a simulation model. A model capable of interpreting XML formatted input can thus be linked to a data source. From the data provider's point of view, there is no need to take into account differing model input data formats, but it suffices to receive a list of variables required for the modelling exercise. The modeller defines these requirements in a DTD file.

Clearly, standardised information transfer techniques become warranted when a large number of data sets and modellers are involved. The strength of XML as a data transfer format lies in applications comprising several parties and extensive data resources. As a platform-neutral and well-supported data documentation format it is readily adoptable and thus offers a promising basis for compatible communication between data and simulation tools in environmental information systems. As an example of XML support, there exist freeware tools developed for parsing and validating XML data. It should be noted, that the present sample application is limited and does not reveal problems that would only appear in more complex settings. For example, the current system does not allow a single model input file to be combined from data residing in several databases. Despite the limitations, the current study suggests that XML is worth exploring on the track towards an open modelling framework. Such a framework would allow resources provided by various expert groups to connect easily to a common information system.

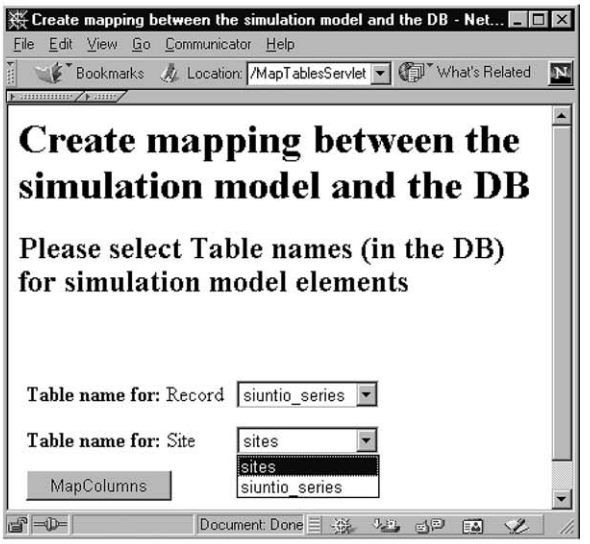
4.2. Vision

The open modelling framework envisioned in this paper comprises tools to connect to model and data resources. Fig. 6 shows an illustration of an open modelling framework where a broker application acts as an intermediary between the user and the data/model servers. The servers provide access to data and model resources in such a manner that the user need not be aware of the internal structure of database/model implementations. In the presented sample application access to model resources was in line with the above definition; input data requirements of a model were documented in a DTD file associated with the model, without specifying names of the variables in the model code. On the contrary, to access a data source the user

(a)

```
<?xml version="1.0" ?>
<!DOCTYPE XMLToDBMS SYSTEM "../xmldbms.dtd">
<XMLToDBMS Version="1.0">
<Maps>
 <ClassMap>
   <ElementType Name="Site"/>
   <ToRootTable>
    <Table Name="sites"/>
   </ToRootTable>
   <PropertyMap>
    <Attribute Name="SiteCode"/>
    <ToColumn>
      <Column Name="site_id"/>
    </ToColumn>
   </PropertyMap>
   <PropertyMap>
    <ElementType Name="SiteName"/>
    <ToColumn>
      <Column Name="site_name"/>
    </ToColumn>
   </PropertyMap>
   <PropertyMap>
    <ElementType Name="PrecipUnit"/>
    <ToColumn>
      <Column Name="precip_unit"/>
    </ToColumn>
   </PropertyMap>
    ...
   <RelatedClass KeyInParentTable="Candidate">
    <ElementType Name="Record"/>
    <CandidateKey Generate="No">
      <Column Name="site_id"/>
    </CandidateKey>
    <ForeignKey>
      <Column Name="site_id"/>
    </ForeignKey>
    <OrderColumn Name="rec_order" Generate="No"/>
   </RelatedClass>
 </ClassMap>
 <ClassMap>
   <ElementType Name="Record"/>
   <ToClassTable>
    <Table Name="siuntio_series"/>
   </ToClassTable>
   <PropertyMap>
    <Attribute Name="RecordNumber"/>
    <ToColumn>
      <Column Name="rec_order"/>
    </ToColumn>
   </PropertyMap>
   <PropertyMap>
    <ElementType Name="Date"/>
    <ToColumn>
      <Column Name="date"/>
    </ToColumn>
   </PropertyMap>
   <PropertyMap>
    <ElementType Name="Precip"/>
    <ToColumn>
      <Column Name="precip"/>
    </ToColumn>
   </PropertyMap>
     ...
  </ClassMap>
</Maps>
</XMLToDBMS>
```

(b)



(c)

Table "sites"

Column	Type	Modifier
site_id	integer	
site_name	character varying(100)	
precip_unit	character varying(10)	
airtemp_unit	character varying(10)	
swdown_unit	character varying(10)	
lwdown_unit	character varying(10)	
windspeed_unit	character varying(10)	
rh_unit	character varying(10)	

Table "siuntio_series"

Column	Type	Modifier
site_id	integer	
rec_order	integer	
date	date	
precip	double precision	
airtemp	double precision	
swdown	double precision	
lwdown	double precision	
windspeed	double precision	
rh	double precision	

Fig. 3. Linking database structure with XML data input file structure (degree-day snow model) with a map file (a), snapshot of the user interface for establishing the linkage (b), and schematic representation of how the data are stored in the relational database (c).

had to explicitly prescribe table and column names, i.e. the internal structure of the relational database. According to the concept of a data server, statements specific to the database architecture should be confined to the server, and more general data access statements would be used in communication between the data server and other framework components.

The envisioned data server describes its resources by providing a metadata XML document (Fig. 7). The structure of the metadata can and should be kept simple. In order to issue a meaningful data request at the technical level, the metadata only need to identify sites with data, stored variables, and periods of data availability. Additional metadata are required to describe semantics of the data to the user. Informations relevant to environmental simulation runs, which should be conveyed in

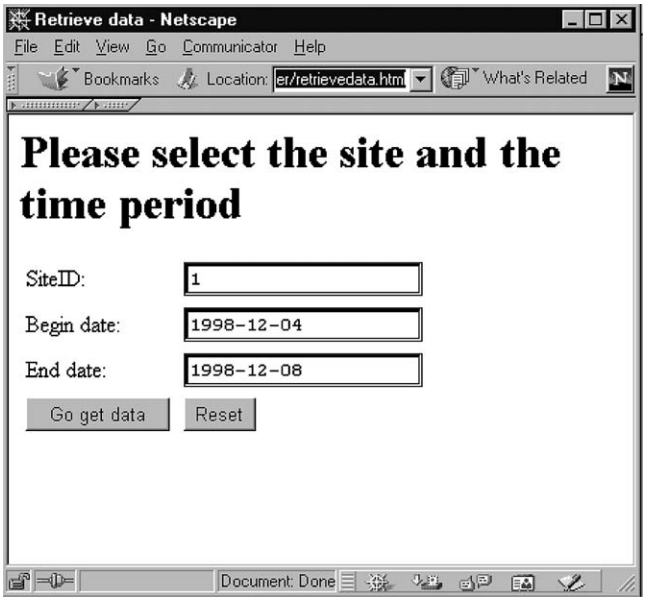


Fig. 4. Snapshot of the user interface for selecting the site and the simulation period.

(a)

```
<?xml version="1.0" encoding="UTF-8"?>
    <Site SiteCode="1">
        <Record RecordNumber="1">
            <AirTemp>0.39</AirTemp>
            <Precip>2.5</Precip>
            <Date>Jan 1, 1997</Date>
        </Record>
        <Record RecordNumber="2">
         ...
        </Record>
        <AirTempUnit>C</AirTempUnit>
        <PrecipUnit>mm/d</PrecipUnit>
        <SiteName>Siuntio</SiteName>
    </Site>
```

(b)

```
<?xml version="1.0" encoding="UTF-8"?>
    <Site SiteCode="1">
        <Record RecordNumber="1">
            <AirTemp>0.39</AirTemp>
            <Precip>2.5</Precip>
            <SWDown>0.2</SWDown>
            <LWDown>25.0 </LWDown>
            <WindSpeed>0.3</WindSpeed>
            <RH>95</RH>
            <Date>Jan 1, 1997</Date>
        </Record>
        <Record RecordNumber="2">
          ...
        </Record>
         <AirTempUnit>C</AirTempUnit>
        <PrecipUnit>mm/d</PrecipUnit>
        <SWDownUnit>MJ/m2/d</SWDownUnit>
        <LWDownUnit>MJ/m2/d</LWDownUnit>
        <WindSpeedUnit>m/s</WindSpeedUnit>
        <RHUnit>%</RHUnit>
        <SiteName>Siuntio</SiteName>
    </Site>
```

Fig. 5. Data input files in XML format for degree-day (a) and energy balance (b) snow models.

metadata, include geographical location and dimensions, units and observation times of the stored variables, measurement methods, river basin characteristics, and freeform comments on data reliability, etc. This division of metadata into technical and semantic parts is simple to implement in XML and allows the data requests to be kept simple while not restricting more intelligence to be developed into components of the open modelling framework. The data server can also offer other services in addition to simple data access. Data services often needed include aggregation of densely observed data into longer time intervals, interpolation over periods of no data, picking maximum/minimum values of data, etc. The data server should describe its capabilities regarding these services in the metadata.

The broker application reports to the user what resources are available and conveys requests issued by the user to the appropriate servers (Fig. 6). Requests must be producible by the broker application based on the resource metadata and the user input. Requests sent by the broker to data and model resources are in XML. The broker can be a stand-alone program or a website. In the latter case the broker is embedded into a program which also creates web pages and maintains user accounts and connection state information.

The recently defined standard for client–server communication, namely the simple object access protocol (SOAP) can be utilised to implement the communication between the client and the servers (http://www.w3.org/TR/SOAP/). The SOAP interfaces that need to be defined can be very simple. Each data/model server provides one method for a broker to request the resource metadata, and one method for a broker to request services, i.e. model/tool runs or data. In this case SOAP is needed only as a carrier of an XML document.

5. Conclusions

One of the real benefits of an open modelling framework as envisioned here is the potential to create integrated tools by combining parts developed and published by different parties. This requires linking of models to other models, and to tools like calibrators, decision support aids, etc. Considering real tasks encountered by the modeller or systems analyst, the problems increasingly lie in integrated analyses, which per se require linking various models together. The structure of the linkage would be described within the broker application, which then would request necessary models and data sets in an appropriate order.

Transparency with regard to interfaces of both data and model resources is vital for efficient integration. As noted by Goodchild et al. (1997), transparency means that the users need not know the physical locations of

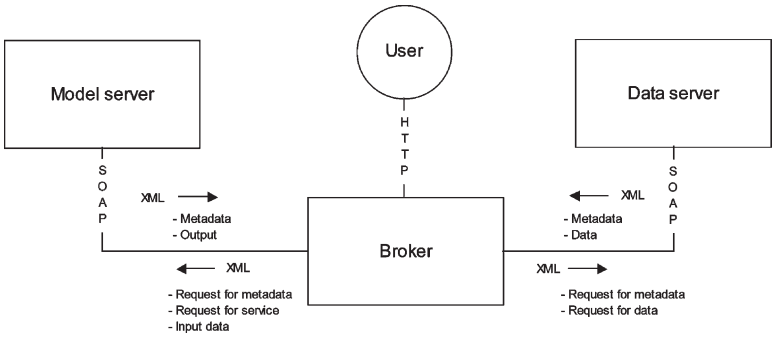


Fig. 6. Schematic representation of an open modelling framework.

(a)

```
<data_resource>
    <site database="Hiidenvesi" dbformat="VYH format" name="Hiidenvesi">
        <variable name="Precip" timestep="1d" unit="mm">
            <period from="7.10.1970" to="31.12.2001"/>
        </variable>
        <variable name="AverTemp" timestep="1d" unit="dC">
            <period from="1.1.1985" to="31.12.2001"/>
        </variable>
    </site>
</data_resource>
```

(b)

```
<model_resource>
    <model name="Snow_dd" type="simulation model">
        <parameters>
            <parameter name="TMIN" default="0" unit="dC"/>
            <parameter name="TMAX" default="1" unit="dC"/>
                      :
                      .
            <parameter name="CRAIN" default="1" unit=""/>
        </parameters>
        <input_variables>
            <input_variable name="TEMP" unit="dC"/>
            <input_variable name="PRECIP" unit="mm"/>
        </input_variables>
        <output_variables>
            <output_variable name="RAINMELT" unit="mm"/>
        </output_variables>
    </model>
</model_resource>
```

Fig. 7. A resource metadata document for a data resource (a) and a model resource (b).

data and software and can work at a more conceptual level. In the open modelling framework discussed here, responsibility for being aware of the physical locations of data and the actual model implementations is confined to the data and model servers, respectively.

XML is a platform-neutral and well-supported data documentation format and as such provides a worthwhile option for transparent interfacing among users, simulation models, and data resources. Efficient sharing of data and methods is vital on the track towards more comprehensive and reliable assessments in environmental decision problems.

Acknowledgements

This study was funded by the Academy of Finland project 'Predicting impacts of land use changes on catchment hydrological processes' headed by Prof. Pertti Vakkilainen (Helsinki University of Technology), whose support is greatly appreciated. Additional funding was received from the Land and Water Technology Foundation, and the Ministry of Agriculture and Forestry of Finland. Constructive comments given by anonymous reviewers are acknowledged.

References

Aspinall, R., Pearson, D., 2000. Integrated geographical assessment of environmental condition in water catchments: linking landscape ecology, environmental modelling and GIS. Journal of Environmental Management 59, 299–319.

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

Blind, M.W., van Adrichem, B., Groenendijk, P., 2001. Generic framework water: an open modelling system for efficient model linking in integrated water management—current status. In: Fourth International Eurosim 2001 Congress 'Shaping Future with Simulation',

Delft, the Netherlands. http://www.genericframework.org/download/pdf/EUROSIM—GF.pdf, accessed January 20, 2003.

Booty, W.G., Lam, D.C.L., Wong, I.W.S., Siconolfi, P., 2001. Design and implementation of an environmental decision support system. Environmental Modelling and Software 16, 453–458.

Bourret, R., 2003. XML-DBMS: Middleware for transferring data between XML documents and Relational Databases. http://www.rpbourret.com/xmldbms/, accessed January 20, 2003.

Brandmeyer, J.E., Karimi, H.A., 2000. Coupling methodologies for environmental models. Environmental Modelling and Software 15, 479–488.

Cate, H.H.ten., Lin, H.X., Mynett, A.E., 1998. A case study on integrating software packages. In: Babovic, V., Larsen, L.C. (Eds.), HYDROINFORMATICS '98, vol. 1. A.A. Balkema, Copenhagen, pp. 457–464.

Edwards, J.L., Lane, M.A., Nielsen, E.S., 2000. Interoperability of biodiversity databases: biodiversity information on every desktop. Science 289, 2312–2314.

Goodchild, M.F., Egenhofer, M.J., Fegeas, R., 1997. Interoperating GISs—Report of a Specialist Meeting held under the auspices of the Varenius Project Panel on Computational Implementations of Geographic Concepts. http://www.ncgia.ucsb.edu/conf/interop97/report.html, accessed June 7, 2002.

Hoch, R., Gabele, T., Benz, J., 1998. Towards a standard for documentation of mathematical models in ecology. Ecological Modelling 113, 3–12.

Howie, C.T., Kunz, J.C., Law, K.H., 1996. Software Interoperability. Stanford University, USA http://www.dacs.dtic.mil/techs/interop/title.shtml, accessed January 20, 2003.

Huang, G.H., Xia, J., 2001. Barriers to sustainable water-quality management. Journal of Environmental Management 61, 1–23.

Huang, G.H., Liu, L., Chakma, A., Wang, X.H., Yin, Y.Y., 1999. A hybrid GIS-supported watershed modeling system. Hydrological Science Journal 44, 597–610.

Jones, J.A.A., 1997. In: Global Hydrology: Processes, Resources and Environmental Management. Longman, Singapore, 399.

Koivusalo, H., Heikinheimo, M., Karvonen, T., 2001. Test of a simple two-layer parameterisation to simulate energy balance and temperature of a snowpack. Theoretical and Applied Climatology 70, 65–79.

Kokkonen, T., Jolma, A., Koivusalo, H., 2001. Interfacing environmental simulation models with databases using XML. In: Ghassemi, F., White, D., Cuddy, S., Nakanishi, T. (Eds.), MODSIM, 2001, vol. 4. The Modelling and Simulation Society of Australia and New Zealand, Canberra, Australia, pp. 1643–1648.

Parker, P., Letcher, R., Jakeman, A., Beck, M.B., Harris, G., Argent, R.M., et al. 2002. Progress in integrated assessment and modelling. Environmental Modelling and Software 17, 209–217.

Reed, M., Cuddy, S.M., Rizzoli, A.E., 1999. A framework for modelling multiple resource management issues—an open modelling approach. Environmental Modelling and Software 14, 503–509.

Rizzoli, A.E., Argent, R.M., Manglaviti, M., Mutti, M., 2001. Encapsulating environmental models and data using Java and XML. In: Ghassemi, F., White, D., Cuddy, S., Nakanishi, T. (Eds.), MODSIM, 2001, vol. 4. The Modelling and Simulation Society of Australia and New Zealand, Canberra, Australia, pp. 1649–1654.

Rizzoli, A.E., Davis, J.R., Abel, D.J., 1998. Model and data integration and re-use in environmental decision support systems. Decision Support Systems 24, 127–144.

Tiktak, A., Vangrinsven, H.J.M., 1995. Review of sixteen forest–soil–atmosphere models. Ecological Modelling 83, 35–53.

Voinov, A., Costanza, R., 1999. Watershed management and the web. Journal of Environmental Management 56, 231–245.