Elsevier Editorial System(tm) for Expert Systems With Applications
Manuscript Draft

Manuscript Number: ESWA-D-11-01474

Title: ModeleR: an enviromental model repository as Knowledge base for Experts

Article Type: Full Length Article

Keywords: environmental model repository; knowledge; experts; information systems; scientific workflow

Corresponding Author: Dr. Ramon Pérez Pérez, Ph.D.

Corresponding Author's Institution: University of Granada

First Author: Ramon Pérez Pérez, Ph.D.

Order of Authors: Ramon Pérez Pérez, Ph.D.; B.M. Benito, Ph.D.; F.J. Bonet, Ph.D.

Centro Andaluz de Medio Ambiente - University of Granada
Av. del Mediterráneo S/N. Granada. Spain
Tel.: +34958241000 – Ext. 31202
Ramón Pérez Pérez
E-mail: ramon@ugr.es

Granada, 1st June 2011

Dear Dr. J. Liebowitz,

Please, find attached our manuscript, entitled "ModeleR: an enviromental model repository as Knowledge base for Experts", for consideration for publication as a full-length article in Expert Systems with Applications. In this paper, we describe the development of a repository of environmental models oriented to cover the gaps found in this context in the disciplines of ecosystems management and environmental conservation. This work has several strengths:

(1) We have created a metadata schema to document environmental models. This specification allows to document different parts of a model and the relationships with other models.
(2) We have integrated in a same system, a system to metadata process and a scientific workflow management.
(3) We have defined three different levels of integration in the repository for a given model. In the first level, the System generates automatically a prototype of the Kepler workflow using model metadata. When an expert edits this prototype and make it totally functional in local host, we can say that the model has reached the second level of integration. The third level implies that the workflow associated with a model is executed remotely on a server by our system.
(4) ModeleR also incorporates Web Services that enhances the integration of the system with other parts of an information system. Finally we have add Web 2.0 functionality (blog and RSS) to improve collaboration and sharing information between team members.

Our work has not been published or accepted for publication, and is not under consideration for publication in another journal or book; its submission for publication has been approved by all relevant authors; and all persons entitled to authorship have been so named.

If you have any queries regarding the manuscript, please do not hesitate to contact the correspondence author.

Look forward to hearing from you.
Sincerely,
Ramón Pérez Pérez

Highlights

We have developed a repository of environmental models. > We have analyzed and assessed other different approaches to the problem. > We have designed and implemented a metadata schema for environmental models. > The ModeleR allows a different degrees of integration for each model. > ModeleR is a knowledge base for modelers that want to document and execute workflows.

ModeleR: an enviromental model repository as Knowledge base for Experts

R. Pérez-Pérez[a], B.M. Benito[a], F.J. Bonet[a]

[a]*Av. del Mediterráneo, s/n. iEcoLab Laboratorio de Ecología - Centro Andaluz de Medio Ambiente - Universidad de Granada, 18006 Granada*

Abstract

In this paper, we present the development of **ModeleR**, a repository of models accessible from the web, which enables the user to design, document, manage, and execute environmental models. The technique and features offered can be applied to any scientific context. Based on the development of its ontology, a metadata system has been established to document the modeling process. The set of models managed from ModeleR reflects the knowledge base of the experts of the system, allowing other experts to reuse, replicate, and delve deeper into the existing models in the repository. Different levels of integration have been included, from the conceptual description of the model to the process needed to execute a model from a remote server, acting as an execution engine through the use of workflow managers. In this paper, we present the problems encountered as well as the solutions reached on developing a prototype of ModeleR set up for ecosystem research and an environmental monitoring lab.

Keywords:
environmental model repository, knowledge, experts, information systems, workflow, scientific workflow

1. Introduction

In each sphere of work, experts are key in transforming the data. To gather information and amass knowledge is a complex process that requires expert training.

The skills of experts concerning how to process data and information to gain knowledge is difficult to abstract and represent. If we focus on expert systems, we attempt to simulate the behaviour of an expert by following a set of rules represented in the knowledge base, which by inference engines seeks to solve specific problems (Ignizio, 1991).

However, the generation of rules is a biased case of generating models that enable the simulation of the real world in any context. In this sense, the modelling enables the construction of representations of the real world that simplify the problem, making it possible to predict or explain the behaviour of the system by making use of the model.

Other initiatives attempt to construct a Decision-Support System, in which the data models existing in the environmental sphere are masked within a system of objects, allowing the reuse of the data contained in the data models, and establishing a system to interconnect the different models at the level of higher abstraction (see Rizzoli et al. (1998)). However, this type of implementation implies unified and centralized access to the data models, but this is not always possible because of the diversity of schemes and suppliers of data involved.

To work with the plethora of complex and diverse models of any scientific domain is not easy. Different models are usually implemented in particular computer languages that may require specific libraries or operating systems to run. Input and output formats are also diverse and, finally, some user interfaces are not friendly enough for inexperienced users.

Faced with this complexity, we have developed a system that provides an easy way to design, document and manage models of the environmental management and ecological research domains. In this paper, we present our research on this topic and, mainly, the design of a functional prototype.

Section 2 details the background of the problem; Section 3 provides an analysis of the problem. Section 4, offers a descripcion of the main features of our prototype. Section 5, shows the API that allow to get the metadata for a model or execute it through the model repository. Finally, in Section 6, the conclusions are drawn.

2. Background

2.1. Introduction to the models

A model is a conceptual construct that describes a physical system existent in the real world and that aids in understanding its functioning by offering a simplified and manageable vision of that reality, adapted to our resources in term of data processing and analysis. This tool, the model, can be used to predict or explain the behaviour of the system subjected to different conditions.

According to Varcoe (1990), the modelling process has a number of advantages and drawbacks. The advantages are:

- The creation of a model always improves understanding of the system being modelled.

- In a matter of minutes, models allow the simulation the passage of long periods of time.

Email addresses: ramon.pperez@gmail.com (R. Pérez-Pérez), blasbenito@gmail.com (B.M. Benito), fjbonet@gmail.com (F.J. Bonet)

- Models can be updated to incorporate new knowledge or technological capacities.
- An established model can be projected over different scenarios to study possible alternatives.
- The creation of models, requiring the work of interdisciplinary groups, favours the information exchange and teamwork.
- The storage of the information is standardized, facilitating the re-analysis and consultation of the results.

Drawbacks:

- Models require an evaluation which has no standard methodology.
- Many models require a great quantity of data that are difficult to collect.
- While it is relatively easy to model any aspect of the environment disregarding critical information, it is difficult to adjust to reality with any guarantee of success.
- A model has applicability limits that should not be exceeded.
- While models generally fit the vision of reality that the researcher provides, this vision may differ from that of other researchers.

In our case, environmental models manage different types of variables (temporal, spatial, heterogeneous, etc.), making them a good example and representative of the generic concept of the model.

2.2. *Environmental Model*

Ecosystem research, environmental monitoring and environmental management, require complex models which require heterogeneous data on different environmental and ecological variables (Lawrie, 2007).

These data can have a geographical dimension when georeferenced, a temporal dimension when covering a time interval, or both at the same time, when the analysis has a spatio-temporal dimension. In addition, the data can also have different degrees of precision and exactitude, according to the sources that provided them, or different spatial or temporal scales. In short, data are inherently heterogeneous (Fegraus et al., 2005) and difficult to classify.

A data set alone provides no answers to any problems. Data need to be processed and analysed to gain information on which to articulate possible solutions to the problem posed. Normally, in a setting of territory management, this processing has a subjective component, linked to the experience of the expert in question. The work of this expert consists of consulting the data, organizing information gained from them, interpreting them, and providing an evaluation from which decisions are made. This mechanism presents several problems, one of which is directly associated to the subjectivity of the expert.

A simple example of an environmental model that requires no expert interpretation would be the calculation of the slope of the terrain (angle of the line of greatest pitch with respect to the reference horizontal) available in any GIS program. If the altitude values provided by the digital elevation model are taken as input data, an algorithm transforms the differences in elevation in adjacent cells in slope values offered in degrees. The output of this process is a digital map with the corresponding slope values assigned to each of the cells in which the territory is divided (Figure 1).

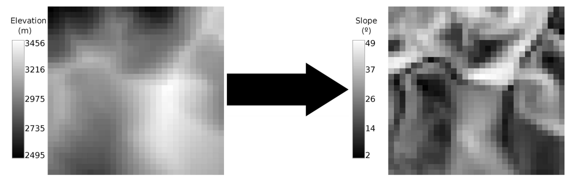

Figure 1: Graphic representation of a slope model of terrain, showing the origin data (elevation), the algorithm represented by the arrow, and the result (slope)

Taking this example as a basis, we can establish for our concept of the model a minimum formal structure composed of the following elements (Figure 2):

- Input data
- Processing cores
  - Input ports
  - Algorithm
  - Output ports
- Output information
- Channels
- Execution method

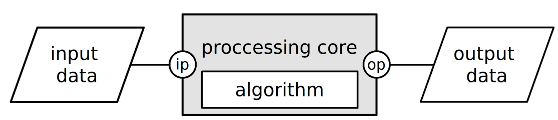

Figure 2: Formal structure of a model

However, in the field of environmental or ecological models, such simple structures are rare. In this type of model, it is very common to use different information sources (such as thematic maps, satellite images, or tables with alphanumeric data), which should be processed by different methods and algorithms (for example, automatic classification of soil uses, statistical analysis or fuzzy logic), and with varied output (new maps, graphs, or summary tables). This combination of elements gives rise to models having more complex structures (Figure 3), with several input data, processing cores with various input and output ports, branched channels, and different types of output information, thus complicating even more the underlying conceptual and physical structure (software). This in turn complicates the comprehensibility and reusability of the model. In research institutions, in which the researchers can use the same complex model with different aims, to cope with this complexity requires a system of storage, documentation, and execution of the models.

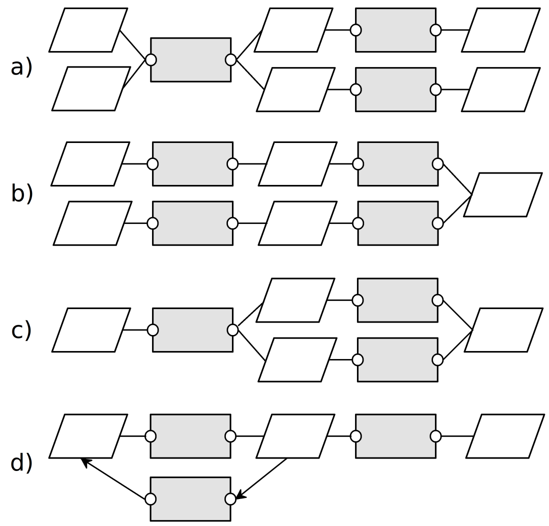

Figure 3: Various examples of complex models: a) branched model, with a sequential execution method; b) model of parallel segments, which can require a sequential or concurrent execution method; c) model with conditional branching, with a sequential execution method; d) model with feedback, which requires an execution method capable of iterating

3. System Analysis

In this section, we analyse the possible systems of documentation of environmental models, focusing on the metadata of the modelling process and the part of the execution of the models by the use of workflow.

3.1. Documentation and cataloguing of models

By a structure of metadata, we can reflect the knowledge concerning a model, making it accessible and reusable by other experts. However, due to the diversity of disciplines, there are many forces to establish a standard according to the field of work. Let us examine two types of metadata.

General-purpose metadata are those used to describe any type of information, the most important example being the Dublin Core (Weibel et al., 1998; Hanlon and Copeland, 2000; Allen, 2000) or Metadata Encoding and Transmission Standard (Cantara, 2005; Cundiff, 2004).

Special-purpose metadata are related to one type of digital information or a thematic domain, some examples in our context being:

- Ecological information: Ecological Metadata Language (Fegraus et al., 2005), Darwin Core, the Biological Data Profile, ...
- Formats of metainformation for geographic information: U.S. Federal Geographic Data Committee (FGDC) or the guideline ISO 19115-2003.
- Formats to describe archive information: Encoded Archival Description, EAD (Dow, 2009).

Following this classification, the general-purpose standards enable us to document a model and its sphere of action, while special-purpose metadata forms part of the specification of input and output data of a model. Nevertheless, no standards have been found that reflect the structure of environmental models.

The steps followed to gather information on starting data are complex tasks and directly reflect the quality of the information gathered. For this, these steps and mechanisms by which the expert achieves the result are process metadata, and should be reflected in the system.

3.2. Scientific workflows

In a setting of problem solving, the network of design processes to execute a model is called workflow. Designed to work on a heterogeneous dataset, workflow comprises a network of analytical processes which can be simple and linear or very complex and nonlinear. In this flow, the data as well as the components that process them are represented according to a specific formal language. That is, the workflow represents the components of the model according to the specific formal language. In addition, the workflow is described as a computer language, which has a visual representation that can be the same or very similar to the conceptual scheme of the model that it executes. The graphic representation of these elements facilitates the understanding of the functions of the workflow, a rapid location and correction of errors, and offers the option of modifying the physical structure of the workflow to gain new functionalities.

Workflow technology pursues the following aims:

- To normalize data-analysis processes within the information system of an organization.
- To make the data-analysis methods implemented in the workflow independent from the persons executing it.
- To facilitate the information-exchange process and ease decision making of an organization.

Different alternatives have been analysed to design and execute workflows:

Pipeline Pilot [1] is a potent client/server platform that enables workflows to be constructed by combining components to collect, filter, and analyse data. In this system, workflows are constructed by a visual language based on modules that fulfil specific functions, connected by flow lines.

[1] `http://accelrys.com/products/pipeline-pilot/`

Simulink [2] allows dynamic systems to be simulated and analysed in multiple domains, taking into account the time factor. Simulink offers an interactive visual setting and an extensive library of components. The system offers enough elasticity to be applied to different scientific and technical domains (Allen et al., 2001). It uses a control system of simulations based on "solvers" that analyse the model and compute their dynamics and execution. These solvers determine which way the data flow, when and how data should be passed from one block to another, and other parameters of the execution of the model.

SCIRun [3] is a design and execution system of workflow oriented to 3D simulation and image analysis, frequently used in the field of biomedical science, although with applications in engineering (Parker et al., 1997).

Taverna [4] is an open-source system for designing and executing workflows developed by the project myGrid (`http://www.mygrid.org.uk/`). The main objective of the project is to enable scientists with limited computer knowledge to construct complex analyses using their own as well as external resources. It specializes in the access to web services to make use of external resources. The workflows are constructed in Taverna by a user-friendly interface with selection and intuitive drag of the components. Taverna has a control element for executing the workflows similar to the solvers of Simulink, called "enactor" (Oinn et al., 2004).

Triana [5] an open and free system to solve problems, was developed by the University of Cardiff in the United Kingdom. The system combines an intuitive interface with a set of powerful analysis tools. It specializes in processing signals, texts, and images and facilitates the integration of its own tools to complete its already broad set of processing modules. It is a particularly good system in design and automation of repetitive tasks, such as data formatting and data collection from sensors (Majithia et al., 2004).

Kepler [6] is a collaborative project with an open code, promoted by the Supercomputation Center of San Diego (USA) `http://www.sdsc.edu/`, and the National Center for Ecological Analysis and Synthesis `http://www.nceas.ucsb.edu/`. It is a system tailored to designing and executing complex workflows, capable of making use of local as well as remote resources. Currently, it is successfully applied in areas such as ecology, molecular biology, genetics, and physics (Ludäscher et al., 2006; Wang et al., 2009; Altintas et al., 2004; Barseghian et al., 2010).

ModelBuilder [7] is a commercial application designed by the company ESRI, specializing in the development of geographic information systems and included as a package of the software ArcGis. It consists of a graphic display prepared for the easy design of geoprocessing flows, using all the tools available in ArcGis, and a visual language based on the shapes and colors of the figures that represent the flow elements. It allows access to Geodatabases, shapefiles, tables, covers, rasters, and CAD files. It permits its processing capabilities to be extended by Python (`http://www.python.org/`), and to be used by other applications.

Macro Modeller [8] is a module of the geographic analysis system Idrisi, specializing in raster-type information, designed by Clark Labs. It is a tool similar to Model Builder and allows easy construction of complex geographic workflows. Its interface, highly intuitive, is governed by the same principles as the other programs mentioned; input modules as well as processing and output data are dragged to a work area, where they are connected by flow lines. Only Idrisi modules can be used as components, implying a major limitation in comparison to other systems analysed. The system is fully directed to a desktop user.

3.3. Similar initiatives

Taking into account the modelling process, and the execution of a model, by flow managers, we have analysed similar initiatives that pursue similar aims than ours:

Ecobas [9] is an institutional network intended to place at the disposition of the scientific community the ecological data collected in research, storing the metadata of the data models used in the sphere of ecology (Benz et al., 2001). It seeks to compile and organize the greatest possible quantity of model metadata in order to make them publicly available. In addition, a cluster of remote desktops has been created with specific software already installed so that scientists can develop their models. For a given model, the following is offered:

- General information of the model.
- Technical information.
- Executables.
- Source code.
- Manuals.
- Data.
- Mathematical logix of the model in question.
- Mathematical operations.
- Default values.
- Inputs.
- Output.

[2] `http://www.mathworks.com/products/simulink/`
[3] `http://www.sci.utah.edu/cibc/software/106-scirun.html`
[4] `http://www.taverna.org.uk/`
[5] `http://www.trianacode.org/`
[6] `http://kepler-project.org`
[7] `http://www.esri.com`
[8] `http://www.clarklabs.org`
[9] `http://ecobas.org/index.html`

Table 1: Comparative table about model repositories

	Metadata	Workflow	Model of data
ECOBAS	•		•
DWMSS		•	
APROMORE			•

- References.
- Additional information on the web.

DWMSS Document-bases workflow modeling support system(Kim et al., 2002) using a reasoner based on cases to implement reuse mechanisms, with three notable main components: user interface, motor, and repository. It delves into the possible relationship between workflows for their reuse on creating new flows:

- The motor consists of three submodules that permit the navigation and edition of the workflows (Case Base Manager and Case Browser), a Case-Based Modeler to find similar cases and a vocabulary base manager.
- The repository is founded on the basic cases and on the vocabulary.

APROMORE is a repository of data models (Rosa et al., 2011) that applies discovery concepts, analysis, and the reuse of data models. It is a system that is accessible from the web with a graphic interface that enables the user to design chains of different types of operations (AND-join, AND-split, OR-join, OR-split) on the data models.

From the initiatives evaluated (Table 1), the most complete is ECOBAS, although it does not cover the aspect of workflow management. It provides a framework to construct environmental models in its own setting.

3.4. *Requisites for the construction of the model repository*

A set of necessary characteristics were established to cover the two main objectives, documentation and execution of a model in an open-code system.

1. Functional requisites
   - To reflect the concept of the model, and the relations between models.
   - To consistently store both the documentation as well as the workflows that enable a certain model to be executed.
   - To characterize and administer the possible data input using the existing standard principles (EML, Dublin Core, etc.).
   - To allow the reuse of the models existing in the repository.
   - To enable the local and remote execution of models making up the repository.
   - To facilitate the discovery by externals users, by consultation mechanisms.
2. Usability requisites
   - To manage users.
   - To generate an interface that enables the user-friendly input of metadata of the model.
   - To foment the interaction and cooperative development of the different experts, as well as the exchange of models with other external users.
   - Search system that enables discovery and reuse.

Regardless of the documentation, the execution of the existing models allows any expert to execute a model without the need to replicate the scenario in which this model was developed. For this, the system would need:

1. To reflect different levels of integration of a model.
2. To execute a model in a way independent from the expert.
3. To establish mechanisms of interaction by APIs that permit machine-machine interaction with the system.

4. System design

ModeleR was designed taking into account a client-server architecture, using a model-view-controller pattern (Figure 4) accessible from the web and permitting interaction by the user interface or accessing by a set of web services.

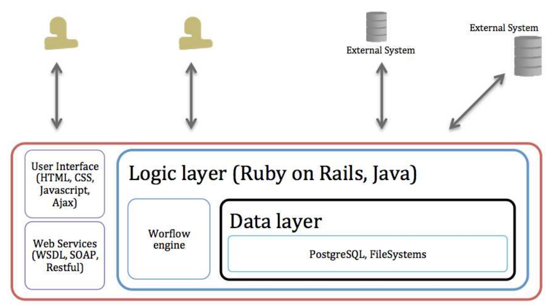

Figure 4: Client/Server architecture of the system.

4.1. *Metadata for environmental models*

Starting from initiatives to document models previously described, we define three main concepts that are key within the system:

Model represents the generic information of the model, covering from general data to the application logic of the model.

Implementation covers the needs to replicate the execution of the model. It indicates its limitations, as well as the access to the software required to execute the model.

Input/output data reflect the data that act as the input or output of the model, given that the output of the model, by a reuse process, can in turn act as an input for another.

For each of these main concepts, we have defined a set of properties necessary for documenting an environmental model:

- Model
  - Identification of the model
    - Acronym
    - Complete name, indicating the meaning of the acronym and its translation
  - Author: one or more authors
    - Name
    - Last name
    - Institution
    - Postal address
    - E-mail address
    - Role
  - Keywords
  - Bibliography: bibliographic relationship on which the model is based.
    - Basic: description of the logic of the model, evaluating it and comparing it with other similar models.
    - Complementary: detailing practical applications of the model, tutorials, and any other source of information not necessarily academic but sufficiently verified.
  - Areas of action: Set of properties that allow the documented model to be contextualized in different thematic and spatial spheres.
    - Sphere of management: application area of the model in the context of different environmental problems: water-resource management, erosion, management of protected areas, etc.
    - Target items: elements of the natural environment which the model affects: forests, threatened flora, target species, atmosphere, etc.
    - Territorial scope: geographic extension and/or specific areas of the territory in which the model will be functional.
    - Practical applications: the relationship which the model establishes between the management sphere, the target item, and the territorial scope. This is an open area in which the specific applications of the model are described.
  - Application Logic: attributes of the model that in general terms describe the internal functioning of the model.
    - Type of model: definition of the model; there are multiple ways of classifying the models according to the typology of the mathematical algorithms used. This field permits a category of this type to be freely assigned to each model.
    - Summary: a short account of the functioning of the model, without dealing with technical aspects.
    - Extended version: a detailed description of the internal logic of the model.
    - Conceptual map: diagram of the internal logic of the model using a conceptual or mental map. This is a link to a diagram that describes the entire process involved in the model.
    - Limitations: factors that limit the applicability of the results of the model, which are important to take into account when interpreting the results. They may refer to calculation limitations, data availability, algorithm fit, etc.
  - Evaluation: Set of attributes used to analyse to what degree the model documented can in reality be implemented in the repository, permitting its execution within that context or not.
    - Usability: subjective evaluation of the original module of software that implements the model, based on the type of interface, data input, formats of input and output, computation time, dependencies, etc.
    - Utility: degree of utility of the model based on the value of the information that it provides.
    - Implementability: brief report on the possibilities of the model to be completely integrated in the repository, based on usability, utility, software architecture, software license, etc..

- Implementation: technical details on the implementation of the model in a computer system.
  - Name: complete name of the program.
  - Download link.
  - Version: number of the version of the program.
  - Type: script, executable, GIS extension, etc..
  - Source code
    - Version
    - Language: knowledge of the language in which the model is programmed; this is necessary for implementing models.
    - Block diagram: description of blocks or modules that form the model as well as the functions and relations established among them.
  - Configuration parameters: enumeration of the parameters that should be introduced into the model and typical values according to the situation (if known). Each parameter has the following attributes:
    - Name
    - Default value
    - Unit
    - Description

– Platform: operating systems that support the program.

– Licence: Type of license applicable to the implementation (free software, proprietary, etc.).

– Software requirements: general description of the software features necessary to execute the model (dependencies, incompatibilities, etc.).

– Hardware requirements: minimum features needed for the computer to execute the model.

– Technical limitations: efficient memory use, maximum size of the input data, etc..

– Known bugs: software problems that could affect the quality or interpretation of the results.

– Tutorial of execution: document that details the steps to follow for the correct execution of the model.

- Input/Output: enumeration of data (layers, tables, etc.) needed to execute the model, together with data models and formats required in each case. Each input/output has several attributes that enable the system to recognize the characteristics of the information source and its specific location. This is key for the execution of the model within the repository.

  – Description: identification of the datum or data set.

  – Data model: Way of representing the information contained in the data source (raster, vectorial, table, matrix, etc.).

  – Format: the file format in which the data source is expressed (geotiff, ascii, etc.).

  – Availability: description of the data location (path, URL, etc.).

  – File: the possibility of annexing a file that contains the data source in question.

By the use of OWL 2, as a descriptive language (Golbreich and Wallace, 2009), an ontology has been generated to describe environmental models. This ontology reflects the relations between the different concepts as well as the restrictions between them (Figure 5). A model can have a set of implementations, which reflect different ways of fulfilling the aim of the model, beginning for example with different input data or simply the set of steps (exemplified by implementations) which are needed to reach the result expected. An implementation has a set of inputs and outputs, where the implementation outputs can act as the input of another implementation. Depending on the tests made with different types of models, this architecture has the versatility necessary to harbour the great diversity of ecological and environmental models that could be included in ModeleR.

The set of concepts established in the design phase is the product of the collaboration of different experts in generating environmental models, using varying techniques of software engineering to work out different concepts and relationships which document an environmental model.

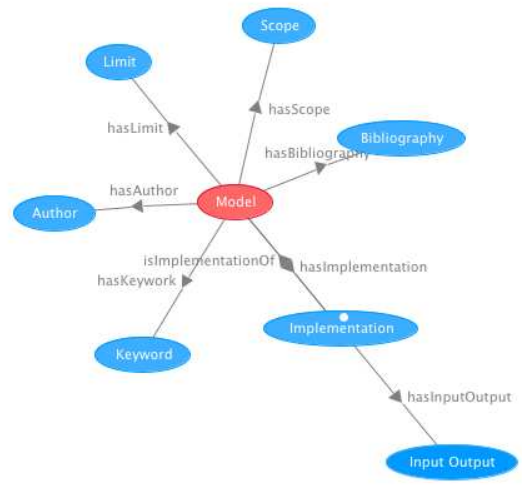

Figure 5: Ontology that represents the domain of the environmental model

For the management of the metadata, an Entity/Relationship model has in turn been designed to unify the access to the instances of the models as well as generate a set of procedures to search, reuse, and export the existing metadata (see Figure 6).

Figure 6 presents the relationship between the entity users, models and implementations. Also, the n-n relation between implementations and models, because a model can have different implementations and an implementation can use differents models. The concept of input is related n-n to the implementations, acting as input or as output of an implementation.

The concept of input, one of the key points of the documentation system, represents the raw or pre-processed data, which would form part of the environmental model. Therefore, apart from the different fields established on the metadata sheet, it is necessary to broaden the set of input objects that the user might specify. For this reason, we have delved into the different specific metadata and the associated protocols.

EML is based on the XML schema-based, which describes ecological data. It is used to describe scientific metadata in detail. EML originated as a work of the National Center for Ecological Analysis and Synthesis (NCEAS) (Michener et al., 1997; McCartney and Jones, 2002), where the software needed to construct a fedetared network of data was analysed and developed using Metacat software (Berkley et al., 2001).

Metacat, a schema-independent data-storage system for XML, acts as an indexer of EML schema, incorporating discovery and data-access mechanisms. The indexing of EML is performed by a harvester process. For this, a standard XML schema harvesting(Sheldon, 2005) was established, where an URL is indicated with the different XML files in the EML specification, its versions and metadata suppliers. By providing an URL with the harvest list document, a Metacat server can index your EML metadata.

The standard EML has been adopted by diverse major entities in compiling and managing ecological information:

- The Ecological Society of America, which promotes ecological science by improving communication among ecologists and raises public awareness of the importance of ecological science.

- The LTER Network (Long Term Ecological Research), which is a network formed to improve long-term knowledge of ecosystems on the global scale LTER (HOBBIE, 2003). From the LTER network, efforts join to achieve a representation standard for ecological data, which permits interoperability in Biology (Gil et al., 2010).

- Organization of biological field stations, which is a non-governmental organization that seeks to help biological stations comprising it to increase their effectiveness in critical scientific research, education, and respect for the environment.

For improved interoperability of ModeleR, a set of tables has been incorporated (external_sources, external_inputs, i_inputs; see Figure 6) so that, by a programmed task, the different lists of EML resources are queried, and the new data sources are added from the differents providers of data included in ModeleR (showed in the table external_sources). Each EML is incorporated as possible inputs/outputs of one implementation.

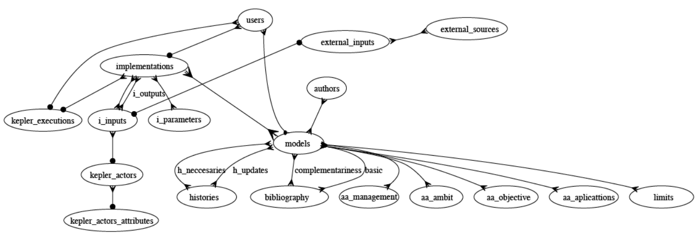

Figure 6: Entity-Relationship Diagram

The user gains access to the system by a process of authentication with the login and password. This access is complemented with the use of an API key that permits access to the web services implemented. In this way, user access is managed together with that of permitted operations so that:

- Users can have access to the content of any model in the system.

- Users can modify or delete only their own models.

- Users can search and link their own model with any other existing ones.

4.2. Models and workflows

After evaluating the different scientific workflow software, we have considered Kepler as the most suitable one for executing the models. It is an open source system, multiplatform system developed for executing scientific workflows, which permits distributed execution, having the support for a multitude of data sources

Taking into account the conceptual description of the model, its implementations and the input/output data, we designed a procedure that generates a non-functional prototype to reflect the interaction of the different models, implementations, and inputs/outputs documented. This prototype defines the first level of integration of the model within the repository based on the metadata of the model (Figure 7).

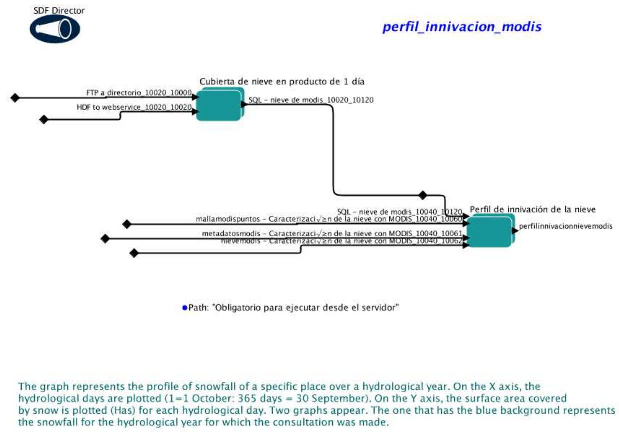

Figure 7: First level of integration: non-functional prototype

The prototype is written in MoML (Modeling Markup Language, Lee and Neuendorffer 2000) based on XML schema. We have established similarities between the different elements of our metadata system and Kepler elements:

- Implementation concept according to ModeleR and the actor composed according to Kepler (Ludäscher et al., 2006)

- Inputs defined in the metadata system of ModeleR and input ports according to Kepler.

- Parametrization (name and default values) of the implementation according to the metadata of the model in ModeleR and the parameters (ptolemy.data.expr.Parameter). The type ptolemy.vergil.kernel.attributes is used, with TextAttribute for the annotation in the form of a description.

Following the diagram in Figure 8, the system generate the prototype that greatly reflects the conceptual design of the model.

The prototype of the workflow can be completed manually by the user, who needs to have knowledge of Kepler. With the xml file, now functional, the user can upload this file in ModeleR, for later downloading and execution. It is also possible to upload to the repository a compressed file with auxiliary files that may be necessary for its execution. Any users of the repository could download the workflow and execute it from their own computer, improve it, and provide feedback for the system.

The execution implies that the user must install the software necessary to execute the workflow, e.g. (Neteler et al., 2008) or Matlab (Higham and Higham, 2005). This may force the adoption of a certain platform (Linux, Windows, etc.) or require the installation of proprietary software, with its respective cost. The generation of the corresponding scientific workflow

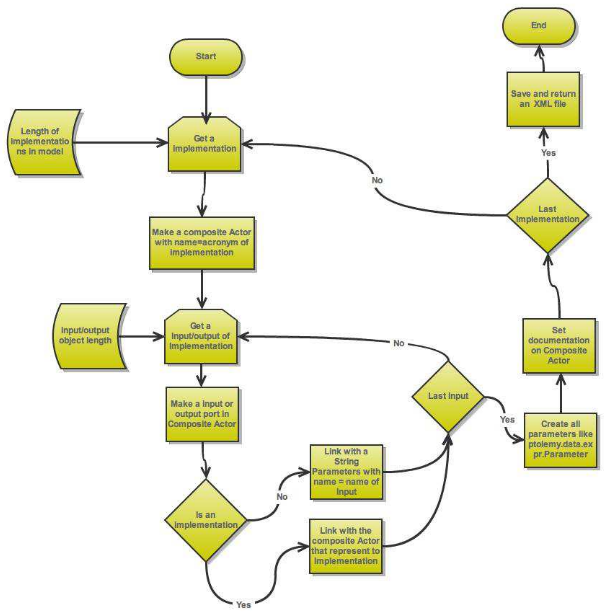

Figure 8: Flow diagram for generating a prototype according to its metadata

to work locally and upload it to the repository constitutes the second possible degree of integration of a model in the repository.

4.3. Remote execution

A third level of integration was designed for remote execution of the workflow.

The execution of a model in a centralized way implies:

- To adapt the workflow to the server's environment.
- To install and configure in the servers the software needed for the execution and management of the workflows that the model requires.

The advantage is notable, given that it avoids the problem of other users being required to adapt their computers, resulting in a monetary savings.

Remote execution is verified from the user interface, uploading to the server both the workflow as well as a compressed file that contains auxiliary files necessary to execute the flow from the server.

For the integration of the model and its corresponding workflow, it is necessary:

- To include the xml file of the Kepler workflow.
- To include a compressed file with the software and additional files which need the workflow to function.
- To document the different parameters of the model.

After the model is integrated into ModeleR, users need only access the system and select the desired model for its execution.

The system shows the user the parameters of the model for its adaptation (Figure 9). Once the user launches the execution of a workflow, the system enables a URI associated with the process, allowing the user to follow the execution process online. When the execution is finalized, the system sends an email with the URL to the process log and to the compressed file with the results.

Figure 9: Configuration of the parameters of a model before remote execution

For the correct execution of the server, the following requirements were established in the workflow:

- PATH parameter: the paths of the existing files in the flow must be related to these path parameters so that the system can change the physical storage of the files, without affecting the workflow.
- The writing of the log of the process in a file called output.log, located in the root directory of the workflow.
- The writing of the output flow files, in the Results file. In this way the results can be compressed automatically to be sent to the user.
- Elements that require the interaction of the user or that show the user visual information cannot be used because the flow is executed in an unattended away.

Following the guidelines designed in the formulation of the workflow, the system can automate its execution and facilitate the above-mentioned services.

4.4. *Reuse of past executions*

The executions made on the server are managed by the repository that stores a record of: the executed model, parameters, user, and the compressed file, with the results that this model generated (see Figure 10 and the entity Kepler_executions related to the entity implementations and users in Figure 6).

Figure 10: List of executions made in the repository by the user interface

The record of the executions enables users to download the results of these evaluations directly without needing to execute the model again. This function is part of the collaborative component of the repository of the models.

4.5. *Tools from web 2.0 and ModeleR*

ModeleR has a blog in which experts can exchange opinions and create inputs and/or commentaries. In this way, we make use of the advantages of web 2.0 (Oreilly, 2007) and its application in ecological research (Waldrop, 2008), thereby fomenting collaboration between different users (see Figure 11). Experts can create inputs with a title and a body. Commentaries can be added to the existing inputs, allowing interaction between experts (see Figure 12).

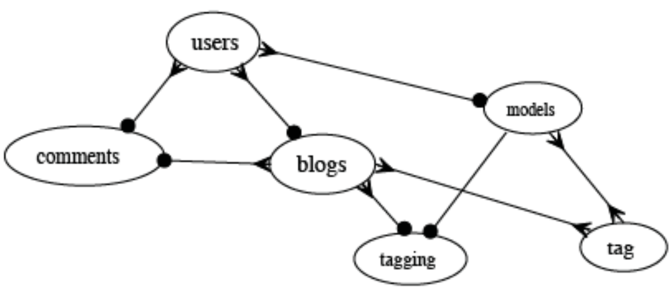

Figure 11: Set of tables for the model web 2.0 of ModeleR

The system complements the management of the inputs, with a search engine for keywords both in the inputs of the blog as well as in the metadata of the model. Finally, a cloud of tags are shown, these permitting a quick navigation through the key concepts of the system.

To improve the integration of the users with the system, a syndication system has been incorporated to allow the user to get

- The latest topics and commentaries.
- New models added to the system.
- The last executions made from the repository.

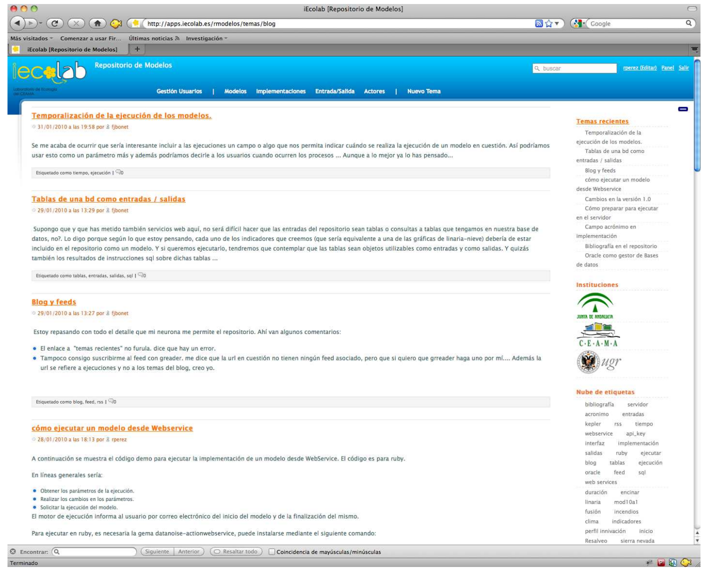

Figure 12: Example of an input in the repository of the models

5. External Services and case studies

ModeleR is positioned as a referent for search as well as execution of the different processes undertaken by the research group. The integration of ModeleR with other applications makes it notable and thus an array of web services have been designed to permit machine/machine interaction.

By a description in WSDL (Christensen et al., 2001) format, web services are offered, these covering the two central aspects of the repository: the management of metadata and the execution of the model.

5.1. *Discovery and access*

The discovery and access of the metadata of the models managed from the repository enable users to delve into not only the process of gathering data but also the process of the model used. Within the context of an information system, this enables the raw data to be linked with the result of an analytical process in which it is involved, through the repository of models.

The services designed are:

Get obtains the metadata for a model. The acronym of the model is used as the identification key. The web service returns the xml with the request of the model requested (Figure 13).

List provides a list of the existing models, providing the acronym and name (Figure 14).

ListImplementations offers a list of the implementations in the repository, providing the acronym, name, and model (Figure 15).

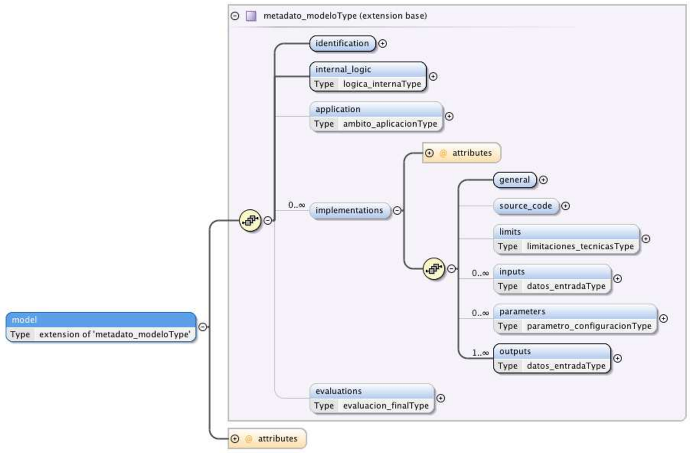

Figure 13: XML Schema for the Get operation

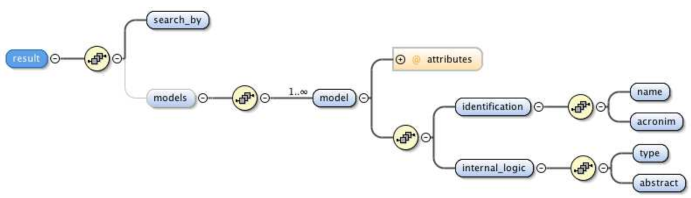

Figure 14: XML Schema for the operation List

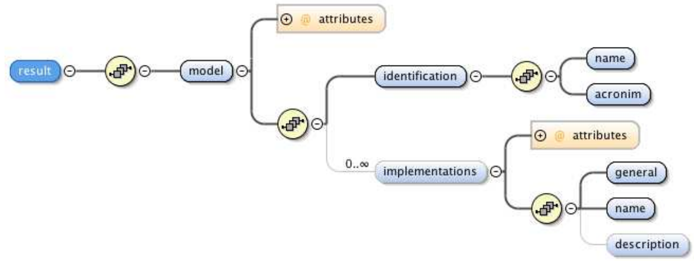

Figure 15: XML Schema for the operation ListImplementations.

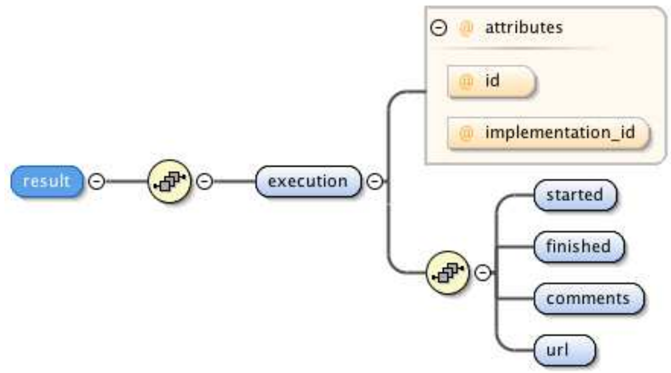

Figure 16: XML Schema for the operation ExecutionImplementation

5.2. Execution of the models

The web services from the repository for the execution of the models are:

ExecutionImplementation returns the result of executing the implementation of a certain model (Figure 16).

ParameterExecution provides a list of the parameters needed in order to execute the model. It receives as a parameter the acronym of the model and the implementation. It provides a list of the parameters with: name, default value, type of parameter, and description of the parameter (Figure 17).

Execute launches the execution of the model in an unattended way. The system begins the normal process of executing a model, sending the email for the start and finish of the execution to the user who launched the execution. In this case, on being an execution by web service, the parameters are passed through the call to the service, so that there is no interaction with the user (Figure 18).

5.3. Example of a model for calculating the snowfall profile

Within the context of the project called Sierra Nevada Global Change Observatory [10], an environmental model was developed to enable calculations of the snowfall profile using snow data of the sensor MODIS of NASA (Hall et al., 2002). The snowfall profile shows the time course of the surface area covered by snow on Sierra Nevada over a given hydrological year. The example that we show describes how this snowfall profile is formulated from the raw data using ModeleR.

Every 8 days, NASA supplies its users an image in HDF format showing the maximum area of the snow cover for the 8 previous days. This information is provided by the sensor MODIS (Moderate Resolution Imaging Spectroradiometer), lodged in the satellites Terra and Aqua. These satellites were placed in orbit by NASA in the year 2000. The product in question, called MOD10A2, represents an effective way of detailed monitoring of the snow cover any place on earth. The images that MODIS supplies for this product have a resolution of 500 m.

Each of the images, available from February 2000 to the present is processed and stored in a data base. Each pixel of each image is translated and recorded in a table (Figure 19). Each record contains information on the presence or absence of snow and on the location of the pixel in question. Given that our study zone has some 78,000 points, the data base contains approximately 42 million records.

After the images are processed, these pre-processed raw data are necessary to generate useful indicators for monitoring the effects of global change. By a set of consultations to the database, we can construct a snowfall profile for Sierra Nevada (see Figure 20). This indicator is composed of two components:

1. The average surface area covered by snow in Sierra Nevada for the entire time series available and grouped by day and hydrological year.
2. The surface area of the hydrological year in process grouped by hydrological day.

This model was added to the repository by dividing it into

[10] http://observatoriosierranevada.iecolab.es

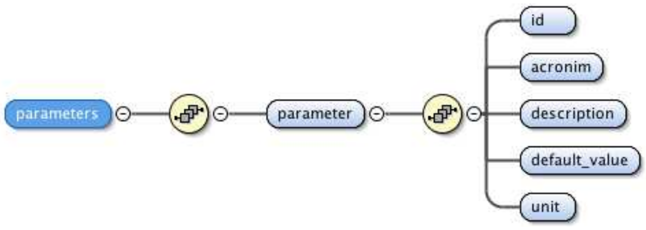

Figure 17: XML Schema for the operation ParameterExecution

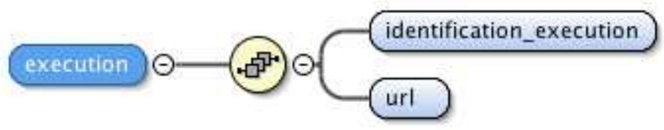

Figure 18: XML Schema for the operation Execute

two implementations:

- Image processing: the associated flow involves the downloading of the HDF files from the NASA servers, the removal of the pixels corresponding to the study zone, and inclusion in the database in question (PostgreSQL).

- Calculation of the snowfall profile: the result of the previous model (a series of records in a table) is used for this second implementation of the model in order to calculate the snowfall profile.

The metadata sheet prepared in order to document the model shows all the information necessary to understand the process from the original information to the final product. In this way, the underlying analytical process is completely documented and becomes part of the corpus of knowledge of the group of experts responsible for the model (Figure 21 and 22).

6. Conclusions and final notes

In this study, we have described our experience in developing ModeleR, a repository of models, accessible through Internet, which acts as a base of expert knowledge. The system is developed taking into account the needs of a group of terrestrial-ecology researchers and environmental monitoring, using standard open ontologies and a workflow manager to generate a dynamic and easily maintained system.

The main conclusions of this work are:

- The structure of metadata has been defined so that the documentation of the environmental models can be managed.

- The tools necessary for ModeleR to act as an execution engine have been designed using Kepler as a scientific workflow manager.

- The capacity to store and manage expert knowledge has been achieved in a flexible way and accessible through the web.

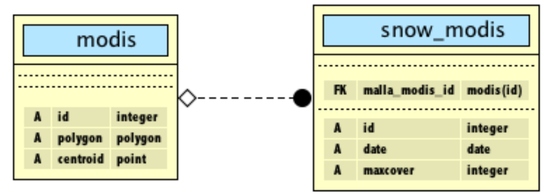

Figure 19: Relationship of tables that store the processing of the snow images

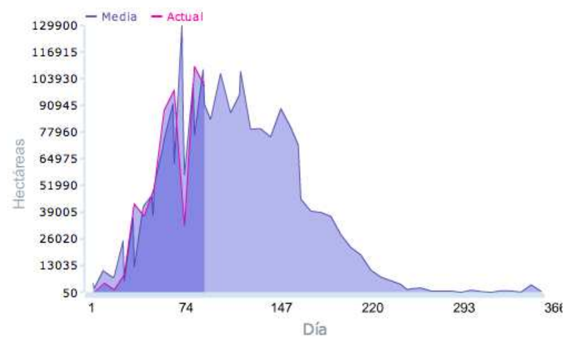

Figure 20: Time course for the quantity of snow accumulated on Sierra Nevada. The blue line shows the innovation profile for the entire time series considered (2000-2011) and the red line represents the profile for the hydrological year in course.

- A set of web services has been designed to allow integration with other systems.

- The resident models in the repository serve for the different phases of the data processing, improving the quality of the information and knowledge derived from them.

- Techniques of web 2.0 have been implemented to foment the collaboration between experts.

Currently, work is being performed to improve the input of data existing in the documentation of the model. For this, other data sources and standards are being evaluated (e.g. Darwin Core), promoted by initiatives at the global level, such as Global Biodiversity Information Facility [11]. In addition, the goal is to improve the integration of the workflow manager with other systems in the Cloud Computing, which enable better execution of the models, making use of the resources in Cloud.

7. Acknowledgements

ModeleR development has been supported primarily by Consejería de Medio Ambiente de la Junta de Andalucía in a project called "Desing and creation of a model repository to the network of ambiental information of Andalucia". We thank Regino Zamora and Raúl Casado for their comments and suggestions.

[11] http://www.gbif.org/

Figure 21: Metadata sheet of the environmental model for calculating the snowfall profile: model and bibliography

Figure 22: Metadata sheet of the environmental model for calculating the snowfall profile: implementation and data

References

Allen, D. Y., 2000. Using the Dublin Core with CORC to Catalog Digital Images of Maps. Journal of Library Metadata 4 (1), 163–176.
URL http://dx.doi.org/10.1300/J141v04n01_15

Allen, E., Lawhite, N., Yoon, Y., Chapman, J., Ilic, M., 2001. Interactive object-oriented simulation of interconnected power systems using SIMULINK. Education, IEEE Transactions on 44 (1), 87–94.
URL http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=912714

Altintas, I., Berkley, C., Jaeger, E., Jones, M., Ludascher, B., Mock, S., 2004. Kepler: an extensible system for design and execution of scientific workflows. In: Scientific and Statistical Database Management, 2004. Proceedings. 16th International Conference on. pp. 423 – 424.

Barseghian, D., Altintas, I., Jones, M. B., Crawl, D., Potter, N., Gallagher, J., Cornillon, P., Schildhauer, M., Borer, E. T., Seabloom, E. W., Hosseini, P. R., 2010. Workflows and extensions to the kepler scientific workflow system to support environmental sensor data access and analysis. Ecological Informatics 5 (1), 42 – 50, special Issue: Advances in environmental information management.
URL http://www.sciencedirect.com/science/article/B7W63-4XB19BP-1/2/55169633bb16008676c6b8350e555074

Benz, J., Hoch, R., Legovic, T., 2001. Ecobas – modelling and documentation. Ecological Modelling 138 (1-3), 3 – 15.
URL http://www.sciencedirect.com/science/article/B6VBS-433GP00-2/2/e866f5b535a36291b48d0ba4cebe528d

Berkley, C., Jones, M., Bojilova, J., Higgins, D., 2001. Metacat: a schema-independent xml database system. In: Scientific and Statistical Database Management, 2001. SSDBM 2001. Proceedings. Thirteenth International Conference on. pp. 171 –179.

Cantara, L., 2005. METS: The Metadata Encoding and Transmission Standard. Cataloging & Classification Quarterly 40 (3), 237–253.
URL http://dx.doi.org/10.1300/J104v40n03_11

Christensen, E., Curbera, F., Meredith, G., Weerawarana, S., 2001. Web services description language (wsdl) 1.1. Tech. rep., W3C Recommendation.

Cundiff, M. V., 2004. An introduction to the Metadata Encoding and Transmission Standard (METS). Library Hi Tech 22 (1), 52–64.
URL http://dx.doi.org/10.1108/07378830410524495

Dow, E. H., 2009. Encoded Archival Description as a Halfway Technology. Journal of Archival Organization 7 (3), 108–115.
URL http://dx.doi.org/10.1080/15332740903117701

Fegraus, E., Andelman, S. J., Jones, M. B., Schildhauer, M., 2005. Maximizing the value of ecological data with structured metadata: An introduction to ecological metadata language (eml) and principles for metadata creation. Bulletin of the Ecological Society of America 86 (3), –168.

Gil, I. S., Hutchison, V., Frame, M., Palanisamy, G., 2010. Metadata activities in biology. Journal of Library Metadata 10 (2), 99–118.
URL http://www.informaworld.com/10.1080/19386389.2010.506389

Golbreich, C., Wallace, E. K., October 2009. Owl 2 web ontology language: New features and rationale. Tech. rep., W3C Recommendation.

Hall, D. K., Riggs, G. A., Salomonson, V. V., DiGirolamo, N. E., Bayr, K. J., 2002. Modis snow-cover products. Remote Sensing of Environment 83 (1-2), 181 – 194.
URL http://www.sciencedirect.com/science/article/B6V6V-4700BJP-6/2/9722a9e37719752f0cae0f454df84746

Hanlon, A., Copeland, A., 2000. Using the Dublin Core to Document Digital Art. Journal of Library Metadata 4 (1), 149–161.
URL http://dx.doi.org/10.1300/J141v04n01_14

Higham, D. J., Higham, N. J., Apr. 2005. MATLAB Guide. SIAM: Society for Industrial and Applied Mathematics.
URL http://www.amazon.com/exec/obidos/redirect?tag=citeulike07-20\&path=ASIN/0898715784

HOBBIE, J. E., 2011/04/14 2003. Scientific accomplishments of the long term ecological research program: An introduction. BioScience 53 (1), 17–20.
URL http://dx.doi.org/10.1641/0006-3568(2003)053[0017:SAOTLT]2.0.CO;2

Ignizio, J. P., 1991. Introduction to Expert Systems: The Development and Implementation of Rule-Based Expert Systems. McGraw-Hill, New York, NY, USA.

Kim, J., Suh, W., Lee, H., 2002. Document-based workflow modeling: a case-based reasoning approach. Expert Systems with Applications 23 (2), 77 – 93.
URL http://www.sciencedirect.com/science/article/B6V03-45JGYB0-1/2/135b3bd748b2339c129e2ca092e62800

Lawrie, J., H. J., 2007. Reducing model complexity via output sensitivity. Ecological Modelling 207 (2-4), 137–144, cited By (since 1996) 11.
URL http://www.scopus.com/inward/record.url?eid=2-s2.0-34548422404&partnerID=40&md5=1b13ecb10e8da00e4660515f08400323

Lee, E. A., Neuendorffer, S., March 2000. MoML - A Modeling Markup Language in XML. Technical Memorandum ERL/UCB M 00/12, University of California, Berkeley, CA 94720.

Ludäscher, B., Altintas, I., Berkley, C., Higgins, D., Jaeger, E., Jones, M. B., Lee, E. A., Tao, J., Zhao, Y., 2006. Scientific workflow management and the kepler system. Concurrency and Computation: Practice and Experience 18 (10), 1039–1065.

Majithia, S., Taylor, I., Shields, M., Wang, I., 2004. Triana: A graphical web service composition and execution toolkit. IEEE Computer Society, pp. 514–524.

McCartney, P., Jones, M., 2002. 'using xml-encoded metadataas a basis for advancedinformationsystems for ecological research. In: Proc. 6th World Multiconference Systemics, Cybernetics and Informatics. Vol. 7. Int'l. Inst. Informatics and Systematics, Orlando, FL. USA, pp. 379–384.

Michener, W. K., Brunt, J. W., Helly, J. J., Kirchner, T. B., Stafford, S. G., 1997. Nongeospatial metadata for the ecological sciences. Ecological Applications 7 (1), 330–342.
URL http://www.esajournals.org/doi/abs/10.1890/1051-0761%281997%29007%5B0330%3ANMFTES%5D2.0.CO%3B2

Neteler, M., Beaudette, D., Cavallini, P., Lami, L., Cepicky, J., 2008. GRASS GIS. Vol. 2. Springer Berlin Heidelberg, Berlin, Heidelberg, Ch. Chapter 9, pp. 171–199.

URL http://dx.doi.org/10.1007/978-3-540-74831-1_9
Oinn, T., Addis, M., Ferris, J., Marvin, D., Carver, T., Pocock, M. R., Wipat, A., 2004. Taverna: A tool for the composition and enactment of bioinformatics workflows. Bioinformatics 20, 2004.
Oreilly, T., 2007. What is web 2.0: Design patterns and business models for the next generation of software. Communications & Strategies 1 (65), 17.
URL http://ssrn.com/abstract=1008839
Parker, S. G., Weinstein, D. W., Johnson, C. R., 1997. The SCIRun computational steering software system. Birkhauser Boston Inc., Cambridge, MA, USA, pp. 5–44.
URL http://portal.acm.org/citation.cfm?id=266469.266473
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 (2), 127 – 144.
URL http://www.sciencedirect.com/science/article/B6V8S-3VGCBRF-4/2/5b3a141d5083f34872a7927b86bd803c
Rosa, M. L., Reijers, H. A., van der Aalst, W. M., Dijkman, R. M., Mendling, J., Dumas, M., GarcÌa-BaÒuelos, L., 2011. Apromore: An advanced process model repository. Expert Systems with Applications 38 (6), 7029 – 7040.
URL http://www.sciencedirect.com/science/article/B6V03-51PRY5V-2/2/fef1d2b3281e34a895889aadf6cdb482
Sheldon, W., 2005. Eml harvesting ii: Preparing site metadata and harvest lists.
Varcoe, V., 1990. A note on the computer simulation of crop growth in agricultural land evaluation. Soil Use and Management 6 (3), 157–160.
URL http://dx.doi.org/10.1111/j.1475-2743.1990.tb00826.x
Waldrop, M., 2008. Science 2.0. Scientific American 298 (5), 68–73.
Wang, J., Crawl, D., Altintas, I., 2009. Kepler + hadoop: a general architecture facilitating data-intensive applications in scientific workflow systems. In: SC-WORKS.
Weibel, S., Kunze, J., Lagoze, C., Wolf, M., 1998. Dublin core metadata for resource discovery.