Earth Sci Inform (2008) 1:131–149
DOI 10.1007/s12145-008-0016-1

SOFTWARE ARTICLE

Earth system curator: metadata infrastructure for climate modeling

Rocky Dunlap · Leo Mark · Spencer Rugaber · V. Balaji · Julien Chastang · Luca Cinquini · Cecelia DeLuca · Don Middleton · Sylvia Murphy

Received: 3 March 2008 / Accepted: 9 October 2008 / Published online: 7 November 2008

Abstract The Earth System Curator is a National Science Foundation sponsored project developing a metadata formalism for describing the digital resources used in climate simulations. The primary motivating observation of the project is that a simulation/model's source code plus the configuration parameters required for a model run are a compact representation of the dataset generated when the model is executed. The end goal of the project is a convergence of models and data where both resources are accessed uniformly from a single registry. In this paper we review the current metadata landscape of the climate modeling community, present our work on developing a metadata formalism for describing climate models, and reflect on technical challenges we have faced that require new research in the area of Earth Science Informatics.

Keywords Climate data portal · Climate modeling · Common data model · Metadata management

Climate modeling and the earth system curator

Research into understanding and predicting Earth's climate has recently been given urgency due to the social, economic, and political implications of a changing climate. Climate modeling centers around the world are running large scale numerical simulations that model the geophysical, chemical, and biochemical processes that drive Earth's climate. Analysis performed on resulting datasets gives us valuable insight into the natural and anthropogenic processes affecting Earth's climate and helps us to better prepare for the future.

Exchange of model output datasets among climate scientists, modelers, and analysts is essential for improving models and reducing uncertainty of climate predictions. For example, Model Intercomparison Projects (MIPs) such as the Intergovernmental Panel on Climate Change Assessments (IPCC 2007) aggregate and analyze datasets generated from modeling centers around the

Communicated by: H. A. Babaie

R. Dunlap (✉) · L. Mark · S. Rugaber
Georgia Institute of Technology,
Atlanta, GA, USA
e-mail: rocky@cc.gatech.edu

L. Mark
e-mail: leomark@cc.gatech.edu

S. Rugaber
e-mail: spencer@cc.gatech.edu

V. Balaji
Princeton University,
Princeton, NJ, USA
e-mail: balaji@princeton.edu

J. Chastang · L. Cinquini · C. DeLuca · D. Middleton · S. Murphy
National Center for Atmospheric Research,
Boulder, CO, USA

J. Chastang
e-mail: chastang@ucar.edu

L. Cinquini
e-mail: luca@ucar.edu

C. DeLuca
e-mail: cdeluca@ucar.edu

D. Middleton
e-mail: don@ucar.edu

S. Murphy
e-mail: murphys@ucar.edu

world. To be successful, these projects must be able to access datasets generated at different modeling centers and moreover, the datasets themselves must exhibit a level of uniformity that enables integration and intercomparison. Such uniformity can be achieved by making datasets available in common data formats and adhering to formal metadata conventions.

To that end, one of the goals of the NSF-sponsored Earth System Curator project (DeLuca et al. 2005) is to develop a metadata formalism for describing the digital resources used in climate simulations. The primary motivating observation of the project is that a model's source code plus the configuration parameters required for a model run are a compact representation of the dataset generated when the model is executed. This observation implies that a comprehensive description of a climate model run is a complete and accurate description of the dataset produced by the model. Despite this connection between models and datasets, the climate community is currently treating the two resources as distinct entities. Those providing data infrastructure (e.g., data portals, search interfaces, retrieval mechanisms) typically begin addressing datasets only after they have been generated, ignoring the steps of the modeling process that led up to the generated dataset. Meanwhile, those providing modeling infrastructure are building tools to facilitate the enormously complex processes of development, assembly, and execution of model codes. But, despite the complexity involved in configuring a model run, the typical output dataset used for analysis is annotated with only a limited amount of metadata (e.g., standard field names, units, horizontal and vertical dimensions). This is an enormous disadvantage for those trying to discover or analyze a dataset based on properties of the original model, and this is the key issue that Curator attempts to address.

The Curator project seeks to blur the line between models and datasets on the premise that a detailed description of a model run is the basis for generating the dataset. The end goal is a *convergence of models and data* where both kinds of resources can be accessed uniformly from a common registry. Providing complete uniformity of access to models and datasets is a high ideal and is currently beyond the state of the art. Nonetheless, we seek to bridge the gap between the two resources as much as possible while laying a foundation for future work in this area.

From Curator's perspective, the primary mechanism for bridging the model—dataset gap is the development of common metadata schemata describing modeling components, model runs, and output datasets. Our goal is to work with existing efforts in the climate community to extend and develop standardized descriptors that can be applied to a wide range of climate models, thereby facilitating interoperability of model codes and moving us toward an environment where users can manipulate models and datasets seamlessly. With these goals in mind, development of climate model metadata has been and is currently the overarching task behind the Curator efforts.

In the rest of this section we present a brief background on climate modeling and describe several use-cases driving the Curator metadata model. In the following section we look at a wide range of existing initiatives of the climate modeling community that have influenced the design of Curator metadata. The next section entitled "Curator Metadata" describes the Curator metadata model at a high level. We then describe an implementation of Curator metadata within the Earth System Grid (ESG) data portal, an existing registry for discovering climate modeling artifacts such as numerical model output datasets (Ananthakrishnan et al. 2007). Finally, we discuss some of the issues and insights encountered along the way and point to the need for new research directions in the area of metadata management for scientific applications.

Project participants

The Curator project comprises funded team members from the National Center for Atmospheric Research (NCAR), the National Oceanic and Atmospheric Administration (NOAA) Geophysical Fluid Dynamics Laboratory (GFDL), Massachusetts Institute of Technology (MIT), and Georgia Tech. Our international collaborators include the developers of the Numerical Model Metadata schemata, which served as a starting point for our development efforts, and PRISM, a large EU model infrastructure project. International collaboration has been essential for ensuring that the collaboratively developed schemata are applicable to a broad community of modelers. Finally, we are indebted to the Earth System Grid Center for Enabling Technologies (ESG-CET) team for the opportunity to collaborate and extend the next generation ESG prototype data portal with Curator metadata.

Problem statement

The underlying problem that Curator addresses is the lack of metadata available to assist climate modelers and scientists in the tasks of building and running climate models and analyzing the resulting model-generated datasets. This lack of metadata creates a "gap" between a given dataset and the model configuration that was used to produce it. Insufficient metadata ultimately impedes scientific progress because scientists and tools do not have access to adequate descriptions of the very datasets undergoing analysis. For example, without an adequate

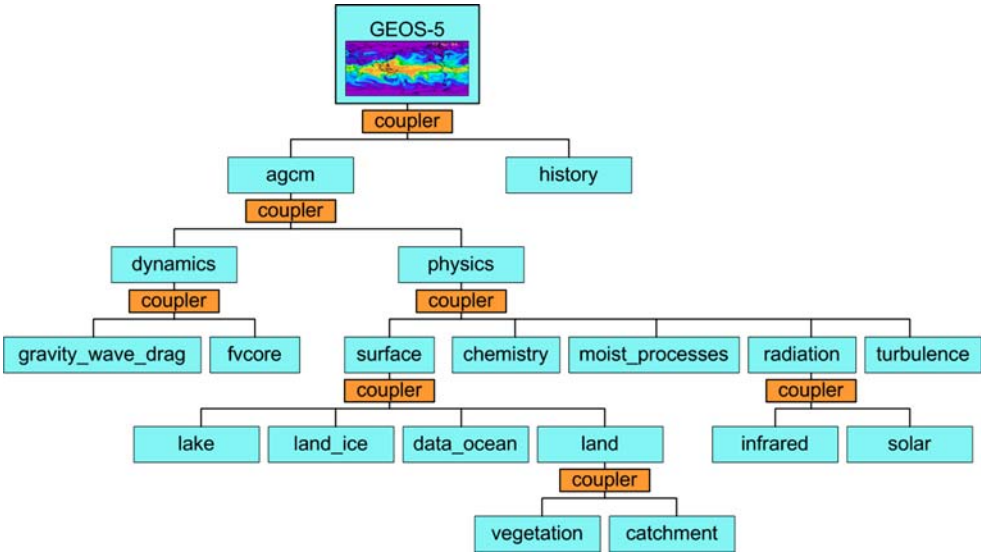

Fig. 1 Architecture of NASA GEOS-5 Atmospheric General Circulation Model

description of a dataset's provenance, it is difficult to determine what post-processing steps have occurred. Or, because most of the original configuration information has been lost, it is nearly impossible to reproduce a previous run. Finally, a lack of metadata impedes model intercomparison tasks because it is not easy to determine the differences between two models.

Curator's task is to develop a set of metadata schema that are sufficient for describing numerical climate models and output datasets. We use the term *model* to refer to numerical climate simulation software and *dataset* to refer to the output file(s) of such a simulation. As laid out by the original proposal, the primary goals of Curator metadata are the following.

- The metadata should serve as descriptions for climate models and datasets such that both can reside in a single registry. A user queries the metadata registry for either datasets or models. If the search is for a dataset, and no suitable dataset is found, the user's results are climate models that could be run to generate the needed data. Since climate models typically take months to validate, the expectation is that the user will get a validated model for routine configurations. For novel configurations, the issue of validation must be addressed by the user.
- The metadata should provide sufficient detail for determining the degree to which two climate modeling software components are technically compatible—that is, to what degree could they be coupled together to form a working simulation.
- Finally, the metadata should provide a description of climate modeling software interfaces that serves as input to a tool for automatically generating source code for coupling components.

It is important to note that currently the Curator project only addresses datasets generated by numerical models and does not consider observational data.

Background on climate modeling

Today's Earth System Models (ESMs) are highly complex systems. While earlier models developed in the 1960s and 1970s were monolithic in nature, today's models are modularized into components, typically partitioned by the geophysical domain or process that they model (e.g., atmosphere, ocean, land, solar radiation). The components are often developed by different groups, at different sites, and must be combined to form an application. Modularization allows modelers to develop components outside of the primary model. Later, when such a component reaches maturity, it is integrated into the primary model as a first class component. Although modularization of components provides this level of flexibility, it may also complicate the process of model assembly and execution.

Figure 1 shows the hierarchical architecture of the NASA Goddard Earth Observing System model, version 5 (GEOS-5)[1]. Each box is a separate component and lines indicate a parent/child relationship between components. For example, the "agcm" component is the parent of the "dynamics" and "physics" components. Processing is abstracted hierarchically such that parent components transfer control to child components to handle certain parts of the computation. Components at the same level communicate via a special component called a coupler. The *coupler* contains the "glue code" necessary for two or

[1] Figure courtesy Atanas Trayanov/NASA GMAO

more components to interoperate. Coupler components range in complexity. In the most basic case, the coupler simply maps the output fields from one component onto the input fields of another. More complex couplers may provide additional processing (such as re-gridding or changing units) to prepare output fields for ingestion by another component.

In response to increasingly complex models, the climate community has developed tools and methodologies to facilitate the modeling process. From the perspective of software development, modeling frameworks have emerged that provide infrastructure and tools to structure climate model codes and facilitate many common tasks (e.g., calendar management, grid generation, I/O). Such frameworks come with a number of advantages, including decreased model development time and increased compatibility of interfaces.

From the perspective of model execution and post-processing, so-called run-time environments have emerged that assist users in configuring, compiling, and running models, as well as post-processing output data files. Many current models are configured manually using text files and executed from the command line. The lack of a user interface makes configuration and execution tedious and error-prone. Run-time environments are intended to reduce the time to get the model up and running and at the same time reduce errors caused by manual manipulation of configuration files.

Motivating use-cases for curator metadata

At a high level, Curator metadata is meant to describe climate modeling software and output datasets in a consistent way in order to promote interoperability and intercomparison of numerical models and their output. In particular, the development of the Curator metadata model can be motivated by the following specific use-cases.

Metadata registry linking numerical climate modeling components and output datasets

A scientist logs into a portal containing both climate modeling software and model-generated datasets. The scientist executes a search for datasets based on properties of the numerical model, such as the grid resolution, the size of the time step, the model's initial conditions, or a particular input file. A list of datasets is returned and the scientists is able to download the dataset in order to perform analysis and visualization tasks locally.

King, et al. describes a *registry* as a "collection point for metadata about resources" (King et al. 2008). A primary goal of Curator metadata is to unify access to climate modeling components and output datasets by allowing descriptions of both to exist in a central metadata registry. This is a significant improvement from typical *repositories* which contain little or no provenance metadata about the numerical models responsible for generating the hosted data. The advantage of bringing numerical model metadata together with output dataset metadata is the ability to explicitly link a dataset with the fully configured numerical model that was run to produce the data. Additionally, if a search returns no results, the same registry can be used to find modeling software that the user could run to produce the data he or she seeks.

The advantages of maintaining a metadata registry for scientific datasets have been well-documented in the literature, and apply to Curator metadata. For example, Bose and Frew point to both *data quality* (e.g., communication of dataset suitability, enhanced interpretation of datasets, reducing risk of false assumptions, ensuring that data archives will be useful for future generations) and *scientific processing* benefits (e.g., audit trails, consistent documentation, reproducibility of processing sequences) (Bose and Frew 2005). Similarly, Simmhan, et al. present a taxonomy of applications for data provenance including data quality, audit trails, replication recipes, attribution, and informational (Simmhan et al. 2005). In addition to these advantages, Curator metadata can drive the design of a sophisticated query interface that supports queries for datasets based on properties of the underlying numerical model.

Automatic assessment of numerical model component compatibility

Consider two separate climate modeling centers, A and B. Modelers at site A would like to test the results of integrating (*coupling*) their atmosphere component with an ocean component written at site B. A number of checks must be performed to determine how suited the components are to interact with each other. The modelers log into the compatibility checker application and submit Curator-compatible descriptions of the atmosphere and ocean components. The compatibility checker performs two types of checks. *Technical compatibility* refers to the computational compatibility aspects of the two components. For example, can the two components run on a common platform? Will the software interfaces allow data exchange? Are the required libraries compatible? *Scientific compatibility* refers to the purely scientific requirements for two components to work together. At this level, we are concerned with whether it would be scientifically feasible to couple the two components. For example, does the atmosphere component expose all the fields that are required by a potentially coupled ocean component? To what degree are the scientific parameterizations compatible? The compatibility checker returns a

report showing the technical and scientific incompatibilities that must be worked out before coupling can take place.

Automatic generation of coupler components

A scientist wishes to couple two modeling components A and B. Based on descriptions of the scientific interfaces of the two modeling components, a coupler generation tool automatically generates the source code required for integrating the two components.

Although couplers can be quite complex, it should be possible to automatically generate at least some of the code required to couple two components based on metadata describing their interfaces. For example, the code that performs simple field mappings between components could be generated automatically. While generating a fully functional coupler from metadata is currently infeasible, Curator metadata allows us to take some first steps in the direction of software generation. In addition to coupling two dynamic (executing) components, the generation tool could also be used to generate *wrapper* code for coupling a component with a static dataset. Such technology would allow modelers to swap dynamic components for *history* or *data-only* versions of components without having to write a large amount of custom source code to achieve the coupling.

Scientific workflows for climate modeling

Assume a scientist A has already configured and run a specific simulation. Later, scientist B would like to rerun the simulation with one of the original parameters slightly modified. Scientist B acquires the experiment description created by scientist A, makes the needed parameter changes, and submits the experiment to a workflow engine for execution. Scientist B is guaranteed that only the intended parameter changes were made.

Numerical modeling in the climate domain is becoming increasingly complex and typically involves a long chain of data-intensive processes. A key application of Curator metadata infrastructure is to serve as a mechanism for both *recording* and *directing* the processes involved in dataset generation, post-processing, and analysis. Workflow-enabled run-time environments will help to automate modeling tasks, ensure repeatability of model runs, and reduce uncertainty about the way datasets are produced.

Previous work

Curator has built on existing metadata infrastructure. An initial task of the project was to assess the current state of the art with regard to schemata in use for describing numerical climate models and output datasets. The following sections describe the community efforts that have influenced our schema development tasks.

Metadata and standards initiatives

Numerical model metadata (NMM)

The Numerical Model Metadata[2] initiative was developed at the University of Reading, U.K. and seeks to add value to climate model output datasets by providing a standard description of the numerical model used to generate the dataset. The idea is that scientists will be able to better use a dataset by having a comprehensive description of the numerical model run that produced it, including all parameter settings used for the model run. NMM recognizes two levels of metadata for describing models: the unconfigured source code level, which describes modeling components without any specific configuration parameters set, and the running model level, which describes a fully configured, executable model. The need to describe various degrees of configuration is important to our metadata development efforts and is currently one of the primary research challenges we are addressing. This includes, for example, the need to describe both partially configured models and fully configured, executable models.

Climate and forecast (CF) conventions

The Climate and Forecast metadata convention facilitates dataset processing and sharing by providing standardized descriptions of fields appearing in climate model datasets and a set of conventions for structuring output data files. A widely used part of the convention is the standard names table[3]—an exhaustive list of variable names than can be used to describe fields in a climate model output dataset. As of June 2008, the table has over one thousand entries. The list is currently manually maintained, with new entries added after an informal approval process. The CF conventions are widely accepted and have been used to standardize field names in dataset submissions for the Intergovernmental Panel on Climate Change (IPCC) assessment reports. Currently, the standard names promulgated by CF are primarily used to describe datasets. Curator, however, leverages the standard names as descriptors for input and output fields of model components. Table 1 shows some sample entries from the CF standard names table.

[2] http://ncas-cms.nerc.ac.uk/NMM/

[3] http://cf-pcmdi.llnl.gov/documents/cf-standard-names/

Table 1 Sample entries from the CF standard names table

Standard name	Canonical units
Atmosphere_net_rate_of_absorption_of_longwave_energy	W m-2
Atmosphere_net_rate_of_absorption_of_shortwave_energy	W m-2
Downwelling_longwave_flux_in_air	W m-2

Modeling frameworks

Earth system modeling framework (ESMF)

The Earth System Modeling Framework[4] is a widely used infrastructure for developing climate, weather, hydrology, and related models. It facilitates the model development process by providing a component-based structure for encapsulating scientific codes and a set of commonly used modeling tools (e.g., calendar management, regridding capabilities, etc.). Curator has relied heavily on ESMF's internal data structures for informing the top level elements of the Curator metadata model. For example, ESMF defines programming concepts such as gridded components, coupler components, import and export states, fields, bundles, clocks, grids, and virtual machines. Because these constructs are generic and intended to work for a wide range of Earth system models, they can be mapped to metadata elements for general model descriptions.

Program for integrated earth system modelling (PRISM)

The Program for Integrated Earth System Modelling[5] is another framework providing component-based modeling infrastructure for climate models. PRISM differs from ESMF in that each model component is compiled as a separate executable (an ESMF application is usually a single executable), and interprocess communication happens via a common, generic coupler called OASIS (ESMF couplers are custom). Nonetheless, the Curator metadata model generalizes such concepts as coupler and field in order to describe both ESMF- and PRISM-based models.

The generic OASIS coupler has influenced the Curator schema heavily since we seek generic descriptors for couplers. In particular, the set of XML files for configuring the OASIS coupler serves as the initial template for a coupling specification that could apply to any coupler used in a climate model, even those models that do not use OASIS to accomplish field exchange among components.

[4] http://www.esmf.ucar.edu/

[5] http://www.prism.enes.org/

Run-time environments

Flexible modeling system runtime environment (FRE) and GFDL curator

NOAA/GFDL's model runs for the IPCC's latest Assessment Report (AR4) were managed by the Flexible Modeling System Runtime Environment (FRE). FRE enabled complete model configurations (source code, compilation, model run sequences of many-month compute duration, post-processing, and analysis) to be maintained in a single XML file. Clearly much of this information is also required as metadata to be used to interpret model output data.

One goal of FRE is to achieve reproducibility of a model run, a central element of the scientific method. The aim of this research is to enable a scientific query to be formulated whose answer points either to a model output dataset, or a model itself, which can be run to provide the result, perhaps even as a web service. This assumes that the model is configured in a manner that has been previously determined to provide valid results. We believe this technology will be transformative in how we go about building and running ESMs.

FMS experiments are described in a comprehensive XML configuration file. This file is processed by a set of Perl metascripts that generate the appropriate scripts for executing the model. Scripts are automatically scheduled for execution by the batch system.

The fields in the XML configuration file serve as a complete description of a model run from source code checkout to post-processing of output data. Because FRE is intended to run at a single lab, many of the workflow details are hard-coded into the metascripts instead of supplied as part of the XML configuration. For example, the logic for acquiring initialization datasets is hard coded into the metascripts. From Curator's perspective, however, we would like to expose the details of the datasets used for initialization because they offer a more complete explanation of the model run—especially if the data will be exchanged with others outside the lab. Part of Curator's task, therefore, is determining what hard-coded pieces of the FRE metascripts should be expressed declaratively as part of our own schemata.

Another activity undertaken at GFDL, complementary to FRE, is the development of the GFDL Curator database—an independent, early prototype database containing metadata describing climate models and datasets. While FRE focuses on the workflow from the producer point of view, the GFDL Curator database provides services from the consumer point of view, offering visitors to the GFDL Data Portal a view into a subset of the metadata associated with the models that produced any dataset. The GFDL Curator currently dynamically harvests the metadata from the model

output data; thus only the subset that has been encoded into the output is available for harvesting. However, the machinery associated with placing the metadata in a relational database has been achieved.

Data frameworks

Earth system grid (ESG)

The Earth System Grid project is sponsored by the U.S. Department of Energy's SciDAC-2 (Scientific Discovery through Advanced Computing) program. ESG is focused on using computational Grid technology to create "a virtual collaborative environment that links distributed centers, users, models, and data." ESG anticipates a time when climate models will produce upwards of tens of petabytes of output data. Making this output data accessible to global-change-impacts researchers and other analysts is essential for making the output data useful, and next generation systems are under development to address the petascale data volumes that the community will face in just a few years. Across the entire ESG enterprise, nearly 10,000 registered users have access to over 230 terabytes of scientific data, and approximately 400 terabytes have been downloaded by a global audience.

ESG has developed an ontology describing a hierarchy of various data products (e.g., dataset, ensemble, campaign, etc.). The ontology is shared with the Community Data Portal (CDP). Working in partnership with ESG/CDP developers, the Curator project team is supplementing the ontology with basic model metadata information. The enhanced ontology and corresponding extensions to the ESG interface will enable users to obtain more thorough descriptions of the models used to generate datasets and to download a wider variety of software (models, components, frameworks) through the ESG or CDP portal.

Summary of influences

Table 2 summarizes the various initiatives that have influenced Curator and lists specifically how each initiative has affected the Curator metadata design and/or the ESG data portal implementation of Curator metadata.

Curator Metadata

Curator has recognized the need for metadata in six key areas. These six areas are depicted as a UML package diagram in Fig. 2. Curator is an evolving metadata model and as such some of these packages remain incomplete. In this paper we describe only the components of the Curator metadata model where we have made significant progress—namely, the Core, Modeling, and Grids packages. Other packages are still in development and will only be discussed at a high level. Also, based on the "Previous Work" section, we have attached notes to each package in the diagram indicating specifically which initiatives have provided metadata structures that Curator has borrowed and/or adapted. These UML packages are theoretical constructs that are realized by several different implementation ontologies developed in parallel throughout the project. While the UML constructs represent general categories of metadata deemed necessary, different ontologies were necessary in order to facilitate interaction with overarching technical system employing the metadata.

Table 2 Climate modeling community initiatives and their influence on curator metadata

Initiative	Influence on curator metadata
Numerical Model Metadata (NMM)	Metadata can be modularized into separate component descriptions
	Model descriptions must describe *unconfigured* model components, including a list of configuration options
	Model descriptions must also describe *configured* numerical models that have been executed (or could be executed)
Climate and Forecast Conventions (CF)	Standard names for output datasets can also be used to describe field import and export states on model components
Earth System Modeling Framework (ESMF)	The modularization of metadata can follow hierarchical architecture of ESMF models
	Parent component descriptions point to child component descriptions
	Internal ESMF structures contain field level attributes that may be written out for harvesting
PRISM/OASIS generic coupler	OASIS coupler configurations are a starting point for generic specifications of how modeling components can be coupled
FMS Runtime Environment (FRE)	An XML configuration serves as a complete workflow description of a numerical model run.
	Configuration details can be grouped by modeling component
	The GFDL Curator database prototype harvests metadata from output datasets
Earth System Grid (ESG)	The existing ESG ontology describing output datasets can be extended to include Curator metadata

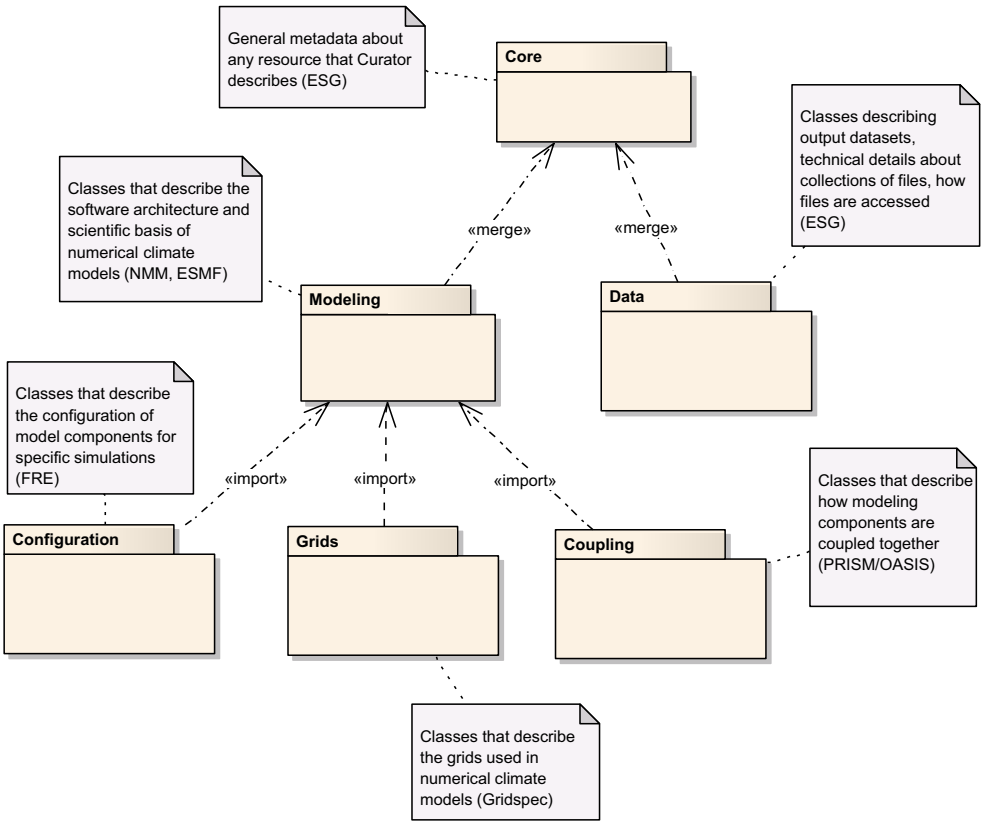

Fig. 2 Curator top level packages

- The Core package contains a small set of generic classes useful for describing essentially any digital resource that we would like to make available in a Curator-enabled registry. The Core classes are straightforward and rather uninteresting, but nonetheless include important information such as related references, institutional ownership, and author contact information. See Fig. 3 for a UML class diagram of the Core classes, including the top level Resource class.
- The Modeling package extends the Core package by supplying technical, numerical, and scientific details specific to numerical climate models. This package deals only with static, un-configured model component descriptions and is largely based on schemata from the Numerical Model Metadata (NMM) initiative.
- The Data package extends the Core package by supplying details for describing output datasets. We do not discuss this package in detail as it is primarily managed by the Earth System Grid.
- The Configuration package contains additional details for describing configured models. Whereas the metadata contained in the Modeling class would be useful for

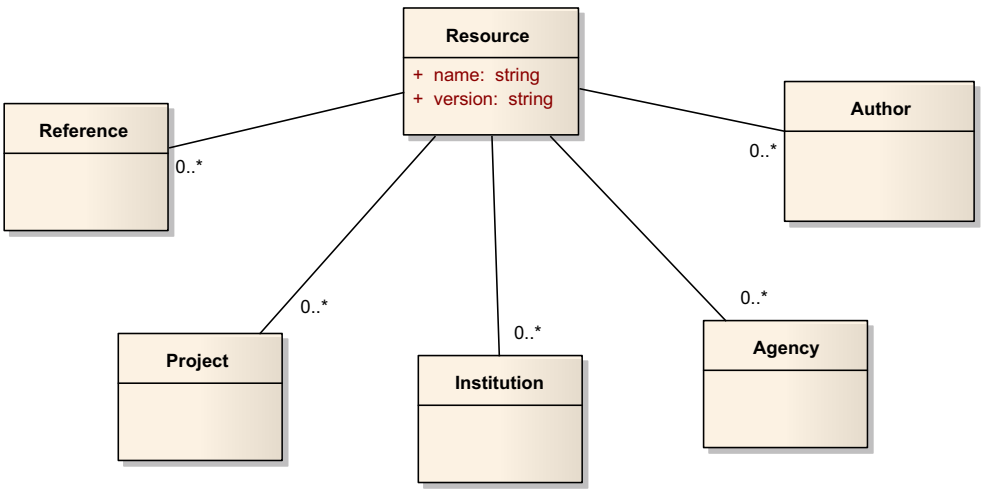

Fig. 3 Classes in Core package

finding and downloading components, the metadata in the Configuration package describes the additional details needed for a specific model run or simulation. The configuration metadata addresses a key part of the scientific method—namely, reproducibility of previous model runs.
- The Grids package includes classes for describing the spatial discretizations supported by modeling components.
- The Coupling package includes classes for describing the technical aspects of how model components are coupled together.

Core package

The Core package contains high level metadata for describing generic resources. Figure 3 shows the classes in the Core package. Both the NMM and the ESG ontologies incorporate facets that represent classes in the Core package.

Modeling package

The Modeling package contains those classes that describe static numerical model components. The contents of this package have been heavily influenced by NMM as well as the component architecture of ESMF.

- The NMM component schema describes individual modeling components.
- The NMM model schema describes sets of components that interact to perform a simulation.
- The ESG ontology combines model and component facets into one Model Component class with the view that all models are also components.

NMM component schema

The NMM Component schema is the most comprehensive schema and is focused on the development of community-wide standards of nomenclature and use. The NMM schema describes the details of a single software component used in a climate simulation and identifies the component's subordinate components. Initially, the metadata for a single component was embedded inside another schema. Curator's contribution was to break out the component metadata into its own schema so that component descriptions are standalone. The component metadata is divided into four "buckets" as defined by NMM.

- The technical properties bucket includes information about the computer architectures ("platforms") that the component supports, programming language used, supported compilers, and even names of local machines that the component can run on.
- The numerical properties bucket references the spatial and temporal discretizations supported by the component. Because grids used in ESMs vary widely (e.g., lat-lon, Gaussian, cubed sphere, unstructured, etc.), work on defining a standardized description of grids has been given special emphasis by Curator and has resulted in its own schema (see grid specification schema below).
- The scientific properties bucket deals with the parameterizations used by the component to model geophysical processes. This is accomplished by providing a list of all parameters understood by the component and optionally the actual settings used during a model run. Standardized parameter names (e.g., those provided by the CF Metadata Convention) are provided when they exist to facilitate intercomparison of components.
- The interface properties bucket deals with component inputs and outputs. An example of an input would be a static initialization dataset (e.g., bathymetry) or a restart file which prepares the internal data structures to continue a simulation that has stopped. Output deals with the names and formats of files that are written by the component. This includes a reference to the description of the interfaces used for coupling—i.e., those fields that another component may set or ingest when performing a coupled simulation.

In addition to these four buckets, the component schema defines general attributes such as local name, description, version, and a unique identifier.

NMM model schema

The NMM model schema acts as a container for a set of components that work together to form a working climate simulation. Attributes of the model are those attributes that are common to the set of components as a whole (e.g., model name, description, version, and a unique identifier). This means that many of the details remain at the component level. For example, the metadata describing the horizontal grid does not appear at the model level, but at the level of each individual component since each component may support different grid representations.

ESG model component schema

The ESG Model Component schema can model an entire componentized modeling system utilizing the concept of child components. Each child component, in addition to the parent, can be fully described using basic, technical, and scientific properties. Basic, unconfigured model components are distinguished from simulations through the attachment of input datasets, configuration files, and history files. Unconfigured models are allowed to have a range of values for various properties (e.g., possible grid resolutions) while simulations are expected to have just one value (e.g., the

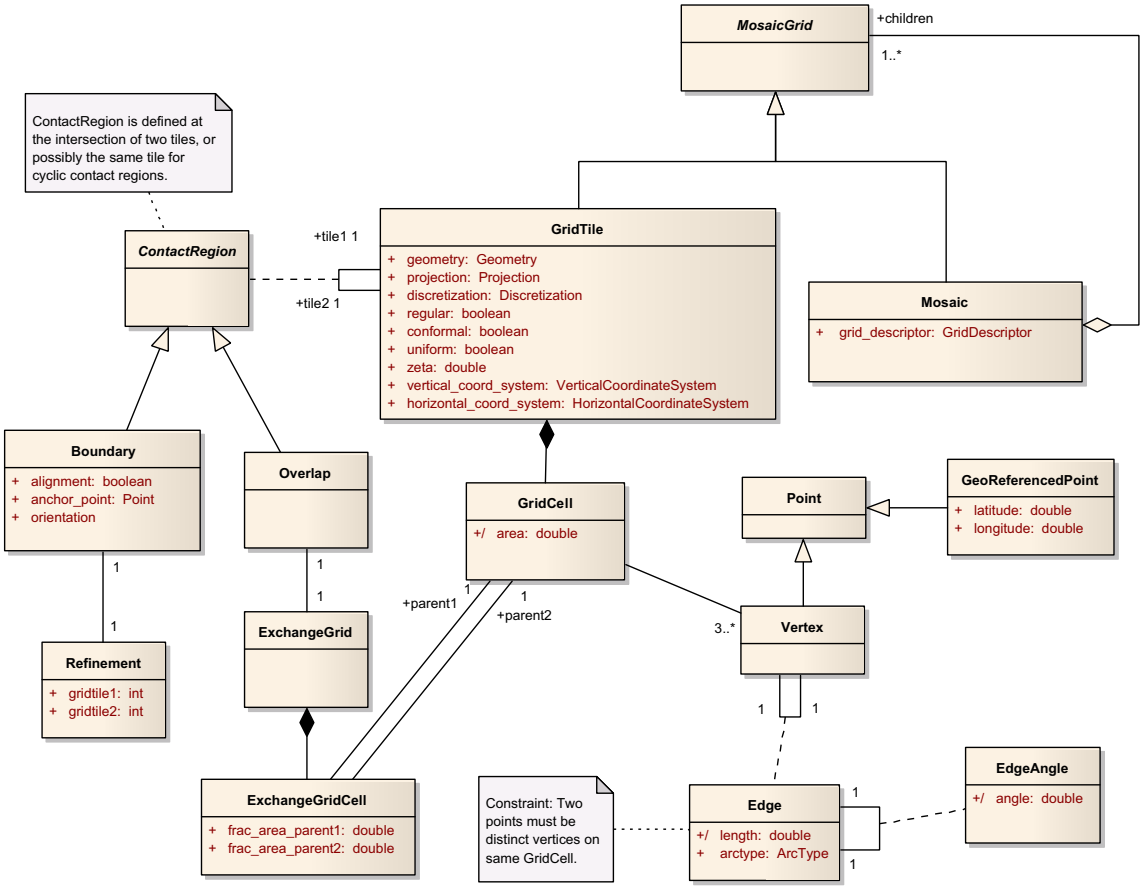

Fig. 4 UML class diagram of the Gridspec

actual grid resolution used during the model run). This ontology is currently in use within the ESG's data portal (see Design and Implementation section), which links datasets to the configured model that generated it.

Grids package and gridspec ontology

The representation of gridded data is a critical element in describing the contents of model output. The grid specification proposal was drafted specifically for inclusion within the Climate and Forecasting (CF) metadata conventions.

The grid specification (or Gridspec[6] for short) focuses on a key element of the metadata under development: the *grids* on which model data are discretized. Experience from recent international modeling campaigns indicates that there is a wide diversity in the model grids used; and further, this diversity is increasing. However, in the absence of a standard representation of grids, it has been rather difficult to perform comparative analyses of data from disparate model grids.

The Gridspec becomes even more necessary in considering other sorts of uses, such as in model *chains* where gridded data from one model becomes input to another. And, last but not least, multiple model grids and data transformations between them are intrinsic to modern ESMs and are the basis for coupled model development for components developed across the entire community (Figs. 4 and 5).

The Gridspec has the following general features and purposes.

- The Gridspec describes the grids commonly used in ESMs from global scale to fine scale, looking toward emerging discrete representations and allied research domains such as space weather and geosciences.
- The Gridspec contains all the information required to enable commonly performed scientific analysis and visualization of data.
- The Gridspec contains all the information required to perform transformations from one model grid to another, satisfying constraints of conservation and preservation of essential features, as science demands.
- The Gridspec makes possible the development of shared regridding software, varying from tools deployable as web services to perform on-the-fly regridding

[6] http://www.gfdl.noaa.gov/~vb/gridstd/gridstd.html

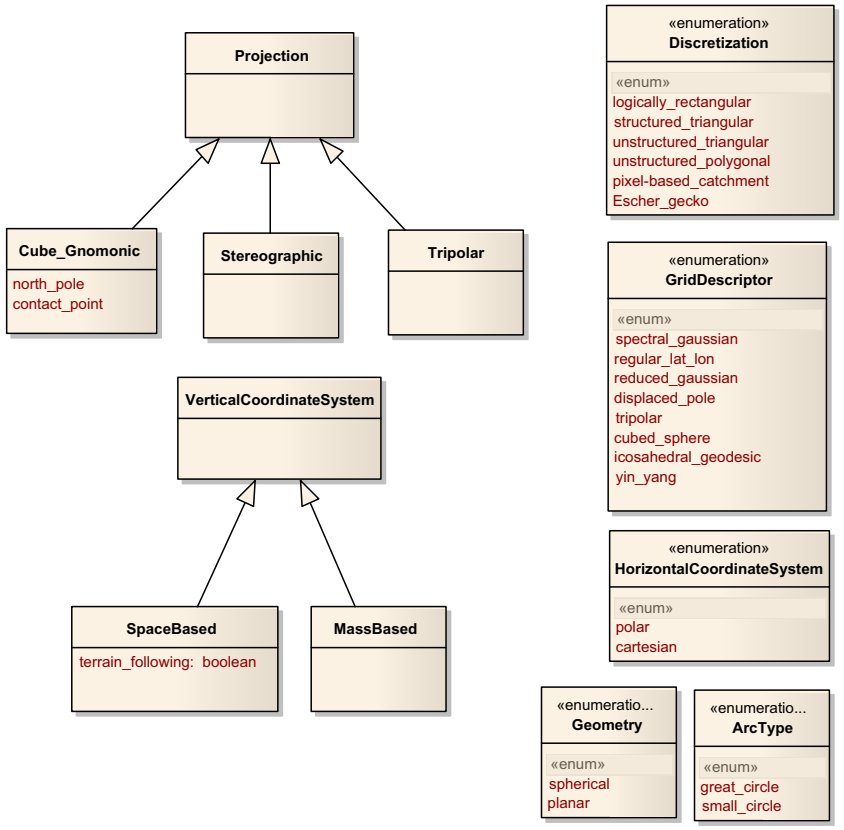

Fig. 5 Additional classes in the Gridspec

from data archives, to routines to be used within coupled models. It will enable, but not mandate, the use of these standard techniques.

Coupling package

The coupling package describes how data is exchanged between two components during a numerical simulation. The latest version of the coupling specification is based on the Potential Model Input and Output Description (PMIOD) XML schema developed for configuring the generic OASIS coupler (Valcke and Redler 2006). Each component references a single coupling specification. The heart of this specification is a description of the fields that the component is able to import and/or export. For example, an atmosphere component may expose fields such as "surface temperature" or "wind speed at 10 m." At this level, we specify all of the fields exposed by a component realizing that some of the fields may be ignored during a particular model run. When a component is configured to be used for a model run, the actual fields exchanged during coupling must be specified.

A single field is described by a number of properties: physical units, valid minimum and maximum values, whether the field is a scalar or vector quantity, the underlying data type, whether the field is intended for input, output, or both, and any field dependencies. A field dependency occurs, for example, when one field is used in the calculation of another. Each field also has a local name and a standard name. The local name is the name of the field from the perspective of the component source code. The standard name is a string from a controlled vocabulary such as the CF Standard Name table.

The Curator project is committed to supplementing existing and forthcoming standards before inventing entirely new schemata. As such, Curator owes a debt to the Numerical Model Metadata (NMM) and PRISM initiatives for allowing us to use their set of schemata as the basis for Curator metadata.

Design and implementation

In this section, we describe a particular application of Curator metadata. Because we do not have the resources to build and maintain our own registry, we have teamed with members of the Earth System Grid who currently maintain a state-of-the-art data portal for locating and downloading climate datasets from a set of distributed repositories. A particular ontology was developed to facilitate this collaboration. Because ESG already had a partially developed ontology, Curator took aspects of NMM and integrated this into ESG. The collaboration with ESG has been mutually

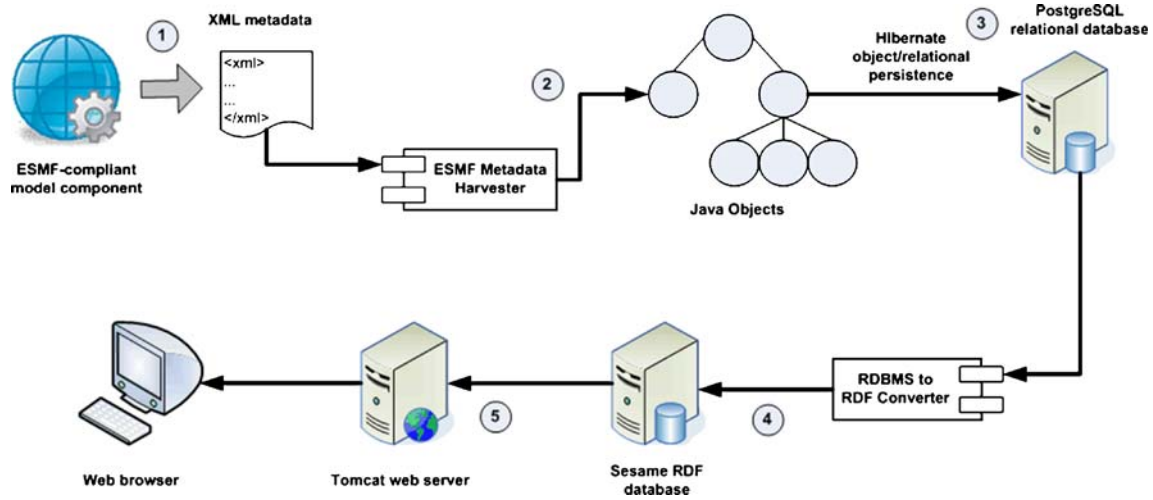

Fig. 6 Metadata harvesting architecture for ESMF-based model components

beneficial: Curator metadata adds value to ESG dataset holdings by providing rich descriptions of the numerical model that ran behind the scenes. Meanwhile, since the ESG data portal itself is already well-established and has robust infrastructure for hosting large datasets and harvesting metadata, it provides a home for Curator metadata and will ensure that our efforts have a path forward in a production data environment after the Curator project finishes. Please note that in this section we only cover aspects of ESG that are relevant to Curator. Readers interested in other aspects of ESG are encouraged to explore additional references (Williams et al. 2007), (Ananthakrishnan et al. 2007), and (Bernholdt et al. 2005).

Metadata harvesting

As discussed in King, et al., the best process for gathering or "harvesting" metadata about distributed resources into a central registry is a subject of debate. A centralized approach where the registry itself generates the needed metadata is appealing because it does not require the resource providers to adhere to any particular metadata standard. However, the centralized approach does not scale well as more providers come online that would like to register resources in the metadata registry. Moreover, it is not always possible for the registry itself to harvest the needed metadata directly from the resources. This is currently the case for Curator since most of the detailed properties of the numerical model are not contained in the datasets themselves. We therefore depend on each modeling center/data provider to submit metadata about their own numerical models.

King, et al. also notes that there is a "natural autonomy" that must be respected when dealing with multiple distributed providers. The climate modeling community is no different and we therefore seek to distribute the metadata generation process back to the providers themselves. This gives data providers the autonomy to run their own models according to their own processes and keeps metadata generation close to where the data itself is written. A disadvantage of this approach is that all providers must submit metadata in a standardized format. This presents difficulties because the standard itself must be developed and maintained. Additionally, it must be sufficiently generic to apply to a wide range of numerical climate models while still containing enough detail to be useful for interoperability and intercomparison of modeling components.

In this section we describe the architecture for harvesting metadata from climate modeling components. Initially, we are only harvesting from ESMF-based components (see the "Influences on Curator Metadata" section for more information of ESMF), although we hope to extend support to modeling components from other frameworks. Figure 6 shows a diagram of the metadata harvesting architecture.

1. A numerical model component writes out its own metadata in XML format. A component may provide very simple metadata, such as the author's contact information, or more detailed metadata, such as a description of the scientific interface to the component (in the form of supported input and output variables), or information about numerical methods used in the component.
2. An ESMF metadata harvesting service validates the XML and instantiates Java objects. The Java object model mimics the metadata in the XML, but allows programmatic manipulation of the objects. Currently, the ESMF harvesting service is invoked manually on a given set of XML instance documents. When the

system goes live, this process will be automated so that XML instances are ingested in conjunction with the datasets they describe.
3. The Java objects are persisted to a PostgreSQL relational database using Hibernate[7]. Hibernate is a persistence and query service that automatically persists Java objects to a relational database without requiring the programmer to write SQL.
4. A subset of the metadata stored in the relational database is extracted into a Sesame RDF database to support queries from the search interface. Currently, we only extract into RDF those portions of the metadata that are common search terms. The extraction tool is a Java application that explicitly maps Hibernate-backed Java objects into RDF triples. The extraction tool runs in the background periodically (e.g., nightly) and for efficiency, only extracts those objects that are new or changed into RDF. This is accomplished by tagging each database object with a lastUpdated property.
5. The Sesame RDF database is accessed by the ESG faceted search interface. Users access the interface with a standard web browser.

Faceted search for discovering resources

Once the metadata is harvested, it is exposed to the search interface to allow users to locate and download modeling components and datasets. In collaboration with Curator, the ESG team has extended a prototype faceted search interface that allows users to find resources based on Curator metadata. A *faceted* search interface enables users to narrow search results by selecting values within a set of orthogonal search categories (e.g., physical domain, numerical method, type of grid, etc.). The facets themselves can be switched on or off and reordered depending on which categories are most essential to the user's search. The user also selects which kinds of resources the search should retrieve: datasets, complete models, model components, or simulations (i.e., models configured for a specific run). Keeping in line with the original Curator vision, the search facets are designed to cut across resource boundaries—that is, the same search criteria can return datasets and models. Therefore, searching for a physical domain of "atmosphere" could potentially return atmospheric modeling components *and* datasets generated by atmospheric components.

Figure 7 shows a screenshot of the Earth System Grid prototype faceted search interface containing Curator metadata. The interface shows seven possible search categories (facets) for finding model components: discipline, physical domain, numerical method, vertical coordinate, equations of motion, grid, and conservation type. The user selects one, several, or all of the search categories. The categories themselves may be reordered as desired. For example, the numerical method category shown in Fig. 7 could be shifted completely to the left and would be the first category from which the user would choose an option. The user works left to right selecting options for each category. As an option is selected (e.g., *discipline* is set to *Atmospheric dynamical core*), the facets to the right are automatically updated to show only those options that overlap with the previously selected options. In the screenshot, the only physical domain associated with the Atmospheric dynamical core discipline is *Atmosphere*. Once the user selects the desired search options, the query can be submitted and the matching resources are returned.

To validate the metadata content and prototype portal, the Curator and ESG teams volunteered to host the data and metadata resulting from a recent NCAR Advanced Study Program colloquium entitled Numerical Techniques for Global Atmospheric Models held during July of 2008. The workshop focused on the comparison of thirteen atmospheric dynamical cores, a key element of next-generation climate models. At the completion of the workshop, the portal contained metadata for the thirteen model instances (dynamical cores), 365 simulation runs and 593 data files.

Issues and insights

Projects building scientific metadata infrastructure such as the Earth System Curator must deal with a number of technical issues. In this section we discuss some of the technical challenges we have encountered along the way.

Issue: developing a conceptual domain model

Prior to defining our formal metadata model, the Curator team spent considerable time identifying domain level concepts and the relationships among them. We have largely used Unified Modeling Language (UML) class diagrams for conceptual modeling. While a class diagram illustrates important concepts and relationships, because terminologies differ greatly among scientists even within the same domain, we have also found the need to develop a glossary of terms with the goal of providing concise yet detailed and accurate definitions of domain level concepts. The glossary is intended to augment the class model diagrams with human-readable descriptions of concepts and encourage community members to formulate concise natural language expressions of domain concepts.

Our glossary entries are represented using the Simple Knowledge Organisation System (SKOS), a data model for

[7] http://www.hibernate.org/

Fig. 7 Screenshot of ESG prototype faceted search interface showing Curator metadata

representing the structure and content of glossaries, thesauri, taxonomies, and similar concept schemes (Miles and Brickley 2005). The SKOS specification provides a Resource Description Framework (RDF) (Klyne and Carroll 2004) vocabulary for representing basic information about concepts such as preferred and alternate labels, definition, example usage, and simple relationships among concepts (e.g., broader terms, narrower terms, and related terms). Figure 8 shows a sample entry from the glossary using RDF/XML syntax.

There are several motivations for using the SKOS vocabulary for the Curator glossary. First, the SKOS vocabulary is simple with a shallow learning curve. Those new to SKOS can write glossary entries after seeing only a small number of examples. Using a simple standard representation is important because we would like to encourage widespread participation in development of the glossary entries. It should be easy for glossary contributors to add and modify entries. Secondarily, we wanted to keep the glossary entries close to the XML schemata containing the concepts described by the glossary. The SKOS RDF/XML syntax allows us to do this since the glossary RDF can be embedded directly within an XML schema. Embedding of RDF/XML inside XML schemas is possible by including the RDF/XML inside the XML Schema *documentation* tag. Any content under the documentation tag is ignored by XML Schema validators, but can still be read by third party tools, such as a glossary generator. The advantage to this approach is that a schema is self-describing in that it contains its own glossary entries. Keeping the glossary entries close the schema also helps to keep the glossary up to date: when a developer changes the schema itself, the glossary can be updated at the same time because it is part of the schema document.

We have developed a web application that extracts glossary entries from XML schemata and displays them with a simple search interface (see Fig. 9). This allows us, for example, to compare glossary entries from several schemata to see if there are conflicts that need to be resolved.

Issue: evolving the conceptual model based on community feedback

There is often impetus to begin prototyping software before the conceptual metadata model has been formalized—for example, to build a proof of concept for review by community members. Unless those people creating the

Fig. 8 Example glossary entry in RDF/XML syntax

```
<skos:Concept rdf:about="http://www.purl.oclc.org/ESC/1.0/#ChildComponent">
  <skos:prefLabel>child component</skos:prefLabel>
  <skos:definition>
        A component that is created/invoked/destroyed by its parent and is required by the parent for execution.
  </skos:definition>
</skos:Concept>
```

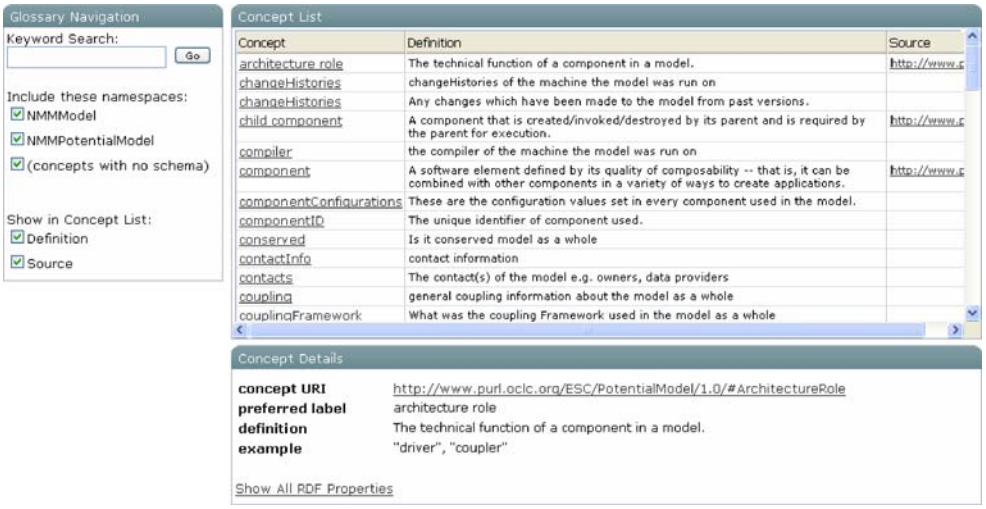

Fig. 9 Glossary screenshot

conceptual model have unusual foresight, the underlying conceptual model will evolve based on feedback from early system prototypes. Curator has been no exception as we have constantly revisited our metadata model to align it with recent community feedback. In fact, it is likely naïve to think that we will ever come to a "finished" metadata model considering that the climate modeling community itself is constantly evolving.

In light of a constantly evolving conceptual model coupled with the need to build rapid prototypes for community review, there is motivation to formally link the conceptual model with the implementation itself. A source of discontinuity in our data modeling process is that there is no formal link between our conceptual models written in UML and their corresponding implementations in XML Schema and RDF/OWL. A well-defined (and possibly automated) process for moving from the conceptual model to the implementation level objects would ensure that the implementation is an accurate and complete representation of the underlying conceptual model. Additionally, without a well-established relationship between the two, data model evolution becomes cumbersome since changes at one level (either to the conceptual model or the implementation) must be manually propagated to the other level. One result of this is the tendency to make changes directly to the implementation without updating the conceptual model. Before long, the conceptual representations are significantly different than the implementation used.

This points to the bigger issue of how to evolve and maintain a standard once it has been published. During the development and prototyping phases, there are relatively few community members (if any) who are dependent on a stable standard. However, once the standard reaches widespread use, schema evolution can become cumbersome, particularly when there is a need to maintain backwards compatibility.

Issue: meeting application requirements through multiple representations of the conceptual model

While we strive for unity at the conceptual level, for technical reasons software systems often rely on multiple metadata representations to meet application requirements. In other words, the same conceptual information is often implemented in several different representations. Again, the Curator project is no exception as we have dealt with a range of metadata representation languages while integrating the Curator metadata model with the ESG data portal. The following table briefly describes the languages involved in the implementation of the Curator metadata with ESG (Table 3).

The semantic web community has been busy developing web-friendly languages for exchanging data and supporting automated reasoning. Our experience with Curator indicates that much of the climate modeling community is comfortable using XML and, to a lesser degree, W3C XML Schema. In fact, at GFDL, dozens of scientists manually manipulate FRE XML configuration files to configure model runs. The XML instances are archived with the output datasets as a description of the model run. A few community members have developed prototype applications for locating datasets based on an underlying RDF/OWL ontology (Klyne and Carroll 2004, Patel-Sc hneider et al. 2004). For example, Blumenthal et al., are designing an OWL ontology for describing datasets (Blumenthal et al. 2006).

RDF was selected for the Curator data portal prototype for consistency with the ESG implementation. One advantage of using RDF/OWL is the ability of query engines to infer new knowledge from existing knowledge. Currently, in collaboration with ESG developers, we are using basic inheritance inferencing in the RDF property and class hierarchy. Curator is exploring what other ways we can use inferencing capabilities to accomplish the application

Table 3 Metadata representations in ESG implementation of Curator metadata

Metadata representation	Primary purpose	Technical motivations
UML class model	Conceptual modeling	Widely used standard for software engineering and data modeling Facilitates dissemination of conceptual model to groups outside of Curator
XML, XML Schema	Standard representation for passing metadata among heterogeneous systems	Widely accepted standard for data interchange on the Web Machine and human readable/writeable Significant API and tool support Many community members already comfortable with it
RDF/OWL	Internal representation	Conceptual nature good for representing high level classes and relationships Flexible, graph-based query language well-suited for faceted search "Web friendly," URIs for identifiers, XML representation Potential to leverage emerging Semantic Web tools and applications
Relational DBMS	Permanent storage of detailed metadata	Trusted, mature technology for reliable long-term storage Powerful query language support Robust backup and replication Automated persistence with Java via Hibernate

scenarios mentioned earlier. Additionally, new knowledge can be added easily without adversely affecting the existing RDF knowledge base. This is a welcome advantage since the proposed additions to the ESG data model are evolving rapidly during this early stage.

A related issue we are facing is the need to do translations from one data format to another. For example, much of our metadata is currently stored in XML instance documents. However, recognizing the need to combine descriptions of a wide range of resources into a common knowledge base (e.g., source code, model configurations, datasets, etc.), we are exploring the use of RDF/OWL. This means we have to translate the hierarchical structure of XML into RDF graphs. Our current thinking is to only translate a minimal amount of information into RDF—that is, the information that we want to search on or reason about while leaving a bulk of the details in XML.

Issue: harvesting metadata from existing, heterogeneous sources

The Curator use-cases are based on the existence of detailed metadata descriptions of climate modeling resources: models, components, datasets, etc. Once the metadata model has been agreed upon, a very real issue is the need to harvest metadata from existing, heterogeneous sources. As discussed in the implementation section, climate model metadata must be in XML format to be ingested into the ESG implementation of Curator. We have considered at least four possible scenarios for populating instance documents with the metadata we require:

- Initially, metadata can be written by hand. As was already mentioned, some climate scientists are already comfortable manipulating XML by hand. For prototype purposes, we have manually generated several XML instance documents that describe components of the NASA GEOS-5 climate model. However, writing metadata by hand is an error-prone task and is not scalable.
- Metadata could be generated by a Graphical User Interface (GUI) or form that serves as a template for the XML. This is an improvement over hand-written XML because it hides the details of writing the tags themselves. Writing a custom GUI by hand would require significant development effort, although a GUI could be generated automatically from the XML schema. However, even if a GUI did exist, the sheer amount of metadata that must be supplied presents a significant data entry burden.
- Another viable option is for model components to generate their own metadata. This is a natural extension to modeling frameworks (such as ESMF) that already have embedded metadata elements (e.g., field descriptors, calendar settings, horizontal and vertical grid descriptors, etc.). ESMF is currently in the process of designing and implementing this capability.
- A fourth option is to use static analysis techniques on model source code to extract metadata elements from the code. Here again we can leverage the common API calls provided by modeling frameworks. For example, a static analysis tool could locate calls to the ESMF function ESMF_StateAddField within a component to generate a list of potential coupling fields. A disadvantage of this method is that all required metadata may not be available statically (e.g., grid resolution may be set at run-time).

Assimilation of metadata into a centralized registry is complicated by the fact that the sources of metadata are heterogeneous. For example, metadata describing the

capabilities of a modeling component must be extracted from source code or at least from descriptions of source code (e.g., code comments, user manuals, etc). Configuration settings are often located in ASCII files, Fortran namelists, shell scripts, or as command line options. The heterogeneous sources of metadata require us to develop automated methods for extracting the needed data elements from each of the underlying sources.

Issue: expressing complex constraints on numerical model configurations

Constraints can be used to reduce user error during the tedious task of model configuration. Providing simple range checks on a model's input parameters allows a runtime environment to signal the user when a parameter value is in error before the run is submitted for execution. For example, the Legend tool can automatically build a Graphical User Interface for configuring ocean prediction models based on an explicit constraint specification written in LCML, a custom markup language (Evangelinos et al. 2006).

In addition to preventing errors, constraints may be used as rules of inference—for example, to infer possible coupling relationships among a set of modeling components based on descriptions of required and available inputs and outputs for each component. Or constraints could be used to infer which external libraries should be used based on the set of components in the model assembly.

The Flexible Unified Model Environment (FLUME) under development at the UK Met Office uses constraints to limit user options during configuration of a numerical model (Ford and Riley 2003). For example, by configuring a model to use a certain optional algorithm, some coupling options may be invalidated because the optional algorithm requires certain fields not provided by all models. Or, selecting a model that exclusively uses OpenMP for parallel programming will automatically eliminate models that strictly use MPI. As a final example, some model implementations are only available on certain hardware architectures. Knowing such constraints will allow tools to automatically infer possible configuration scenarios before the model is assembled.

The potential to automate configuration tasks, infer new knowledge, and reduce user error make constraint specification an important aspect of Curator. There are a number of formalisms for writing down constraints. Understanding the differences in power and expressiveness of these formalisms is important for determining which formalisms are suited for which applications. Schema languages, such as XML Schema, are primarily concerned with expressing structural constraints, but lack the expressiveness to define more semantic constraints such as relating the values of multiple elements appearing in an instance document. Languages such as XSLT (Clark 1999) and Schematron (Clark et al. 2006) are more powerful because they leverage the navigational abilities of XPath (Clark and DeRose 1999) and include functions for comparing node values.

Ontology languages such as RDF and OWL deal with constraints at a conceptual level. RDF Schema, for example, provides domain and range constraints on properties. OWL provides more sophisticated modeling constraints such as explicit cardinalities, universally and existentially quantified property constraints, symmetric and transitive properties, and class definitions based on the union, intersection, or complement of other classes. These types of advanced constraints provide a semantically-rich conceptual model and open the door to advanced inferencing capabilities.

Issue: unifying prospective and retrospective provenance

Clifford, et al. describe two distinct approaches for viewing instances of scientific metadata: prospective and retrospective (Clifford et al. 2008). The *retrospective* approach views a metadata instance as a historical account of what happened—that is, a description of a numerical model run that has already taken place. NMM and CF both fall into the category of retrospective metadata. An alternative viewpoint is to see a metadata instance as *prospective*—that is, as a blueprint or configuration manual for how to build and run a model. For example, the FRE configuration XML and FLUME metadata can be considered prospective because they are used to automate model configuration tasks. The retrospective approach is prevalent in the climate modeling community—likely because it is easier to provide metadata for informational purposes to a human reader than it is to alter existing processes to read metadata descriptions and translate them to the appropriate actions needed to build and run a model. Although retrospective provenance is not dependent on the existence of prospective provenance (Freire et al. 2008), the goal of Curator can be seen as an effort to unify these two approaches. Ideally, a metadata instance document resulting from a model run could be slightly modified (e.g., by changing only a single parameter value) and fed back into the system to produce a modified model run. Using this approach, a scientist can guarantee that the only difference between two model runs is the change he or she intended.

Seeing metadata as prospective or retrospective influences schema development. In particular, retrospective metadata that will be read by humans can be far less formal than prospective metadata that needs to be processed by a machine. For example, a free form textual description of a model run may convey enough information for the human reader to understand what happened at a high level, but

would hardly be sufficient as an instruction set for recreating the model run.

Because we wish to view metadata instances as both retrospective *and* prospective, we have the need to allow users to describe partially configured model components. The metadata is incomplete in the sense that the user is not forced to fill in all possible values required for a specifying a working model run. For example, a user may wish to query for a dataset by providing only a partial model configuration. Any dataset generated from a model run that matches the partial configuration should be returned to the user. Or consider the compatibility checking application described earlier. In this case, the user may provide a partial list of components and ask the compatibility checker to determine what other components could interact to form a viable simulation. As a final use case, consider a workflow tool designed to facilitate model assembly and execution. During the assembly stage, the user will provide only a partial model configuration before fixing any parameters required during execution.

Conclusions and future directions

We have much work to do in the final year of the Curator project. From a software development standpoint, our primary focus is to continue working with the Earth System Grid team to enhance the data portal with more detailed metadata. Our close interactions with ESG are one of the primary means to validate the quality of the schemata that we have created.

While Curator has focused primarily on describing the "technical" details of climate modeling components, much work is left to be done on describing the purely scientific aspects of climate models. Although it is currently beyond the state of the art, the ability to formally describe the scientific aspects of models is appealing because it opens the door to scientific—not just technical—compatibility of modeling components.

Another key focus for the final year of Curator (and beyond) will be the development of the FRE system at GFDL into a more sophisticated and robust workflow environment. The challenge is to make the workflow schema seamless: the FRE schema and the GFDL Curator schema are being merged. In 2008, all of the FRE metadata will be available in the GFDL Curator database.

We are also looking forward to the results of the newly funded European project called METAFOR[8] that continues the work that NMM, PRISM and Curator have started. A promising end result of this project will be a standard for climate model descriptions. The community as a whole is moving toward increasingly consistent and comprehensive metadata with the hopes of achieving levels of semantic interoperability that were previously not possible.

Acknowledgements The Curator team includes Cecelia DeLuca, V. Balaji, Chris Hill, Don Middleton, Julien Chastang, Sylvia Murphy, Serguei Nikonov, Luca Cinquini, Michael Burek, Spencer Rugaber, Leo Mark, Rocky Dunlap, Angela Navarro, and Swetha Patnaikuni. This work would not have been completed without our international collaborators: Lois Steenman-Clark, Katherine Bouton, Sophie Valcke, Ian Henderson, Rupert Ford, and Allyn Treshansky. The development of the ESG metadata ontology and the web-based search application was conducted in collaboration with the ESG-CET project. ESG-CET (Earth System Grid Center for Enabling Technologies) is sponsored by the DoE SciDAC program (Scientific Discovery through Advanced Computing). The National Center for Atmospheric Research is sponsored by the National Science Foundation. The Curator project is supported by NSF Grants 0513635, 0513762 and the NSF Graduate Research Fellowship.

References

Ananthakrishnan R, Bernholdt DE, Bharathi S, Brown D, Chen M, Chervenak AL, Cinquini L, Drach R, Foster IT, Fox P, Fraser D, Halliday K, Hankin S, Jones P, Kesselman C, Middleton DE, Schwidder J, Schweitzer R, Schuler R, Shoshani A, Siebenlist F, Sim A, Strand WG, Wilhelmi N, Su M,Williams DN (2007) Building a global federation system for climate change research: the earth system grid center for enabling technologies (ESG-CET). Journal of Physics: Conference Series 24–28

Bernholdt DE, Bharathi S, Brown D, Chanchio K, Chen M, Chervenak AL, Cinquini L, Drach B, Foster IT, Fox P, Garcia J, Kesselman C, Markel R, Middleton D, Nefedova V, Pouchard L, Shoshani A, Sim A, Strand G, Williams D (2005) The earth system grid: supporting the next generation of climate modeling research. Proceedings of the IEEE 93:485–495

Blumenthal MB, del Corral J, Bell M, Grover-Kopec E (2006) An ontological approach to geoscience dataset cataloging, Technical Report, FS-05-06, Association for the Advancement of Artificial Intelligence

Bose R, Frew J (2005) Lineage retrieval for scientific data processing: a survey. ACM Computing Surveys 37:1–28

Clark J (1999) XSL Transformations Version 1.0. http://www.w3.org/TR/xslt

Clark J, DeRose S (1999) XML Path Language (XPath) Version 1.0. http://www.w3.org/TR/xpath

Clark J, Holman K, Bryan M (2006) Information technology — document schema definition languages (DSDL) — Part 3: Rule-based Validation — Schematron (ISO/IEC 19757-3:2006). http://www.iso.org/PubliclyAvailableStandards

Clifford B, Foster I, Voeckler J-S, Wilde M, Zhao Y (2008) Tracking provenance in a virtual data grid. Concurrency and Computation: Practice and Experience 20:565–575

DeLuca C, Balaji V, Middleton D, Rugaber S, Marshall J (2005) Earth system curator: spanning the gap between models and datasets. http://www.earthsystemcurator.org

Evangelinos C, Lermusiaux PFJ, Geiger SK, Chang RC, Patrikalakis NM (2006) Web-enabled configuration and control of legacy codes: an application to ocean modeling. Ocean Modelling 13:197–220

[8] http://metaforclimate.eu/

Ford RW, Riley GD (2003) A flexible unified model environment: D1 system architecture. http://www.metoffice.gov.uk/research/interproj/flume/pdf/d1_r8.pdf

Freire J, Koop D, Santos E, Silva CT (2008) Provenance for computational tasks: a survey. Computing in Science and Engineering 10:11–21

IPCC (2007) Climate change 2007: synthesis report. contribution of working groups I, II, and III to the fourth assesment report of the intergovernmental panel on climate change. http://www.ipcc.ch/pdf/assessment-report/ar4/syr/ar4_syr.pdf

King T, Merka J, Walker R, Joy S, Narock T (2008) The architecture of a multi-tiered virtual observatory. Earth Science Informatics 1:21–28

Klyne G, Carroll J (2004) Resource description framework (RDF): concepts and abstract syntax. http://www.w3.org/TR/rdf-concepts/

Miles A, Brickley D (2005) SKOS core vocabulary specification. http://www.w3.org/TR/2005/WD-swbp-skos-core-spec-20051102/

Patel-Schneider PF, Hayes P, Horrocks I (2004) OWL web ontology language semantics and abstract syntax. http://www.w3.org/TR/owl-semantics/

Simmhan YL, Plale B, Gannon D (2005) A survey of data provenance in e-science. SIGMOD Record 34:31–36

Valcke S, Redler, R (2006) OASIS4 User guide (OASIS4_0_2). http://www.prism.enes.org/Publications/Reports/OASIS4_User_Guide_T4.pdf

Williams DN, Bernholdt DE, Foster IT, Middleton DE (2007) The earth system grid center for enabling technologies: enabling community access to petascale climate datasets. CTWatch Quarterly

Availability and requirements

The Curator XML schemata are available on Sourceforge at http://curatordb.cvs.sourceforge.net/. XML schemata may be viewed with an standard text editor or an XML editor such as the oXygen XML Editor (http://www.oxygenxml.com/) or XMLSpy (http://www.altova.com/products/xmlspy/xml_editor.html). Additional information about Curator, including links to UML diagrams are available from the Curator home page: http://www.earthsystemcurator.org.