A Semantic Model Catalog
to Support Comparison and Reuse

Daniel Garijo¹, Deborah Khider¹, Yolanda Gil¹, Lucas Carvalho², Bakinam Essawy³,
Suzanne Pierce⁴, Daniel Hardesty Lewis⁴, Varun Ratnakar⁵, Scott Peckham⁵,
Chris Duffy⁶, Jonathan Goodall³
¹University of Southern California
²University of Campinas
³University of Virginia
⁴University of Texas Austin
⁵University of Colorado Boulder
⁶The Pennsylvania State University
dgarijo@isi.edu, khider@usc.edu, gil@isi.edu, lucas.carvalho@ic.unicamp.br,
bte2rn@virginia.edu, spierce@tacc.utexas.edu, danielhardestylewis@utexas.edu,
varunr@isi.edu, scott.peckham@colorado.edu, cxd11@psu.edu, goodall@virginia.edu

Abstract: Model repositories are key resources for scientists in terms of model discovery and reuse, but do not focus on important tasks such as model comparision and composition. Model repositories do not typically capture important comparative metadata to describe assumptions and model variables that enable a scientist to discern which models would be better for their purposes. Furthermore, once a scientist selects a model from a repository it takes significant effort to understand and use the model. Our goal is to develop model repositories with machine-actionable model metadata that can be used to provide intelligent assistance to scientists in model selection and reuse. We are extending the OntoSoft semantic software metadata registry (http://www.ontosoft.org/) to include machine-readable metadata. This work includes: 1) exposing model variables and their relationships; 2) exposing model processes and how they group and relate to model variables; 3) adopting a standardized representation of model variables based on the conventions of the Geoscience Standard Names ontology (GSN) (http://www.geoscienceontology.org/); 4) capturing the semantic structure of model invocation signatures based on functional inputs and outputs and their correspondence to model variables; 5) associating models with readily reusable workflow fragments for data preparation, model calibration, and visualization of results. The extended OntoSoft framework will reduce the time to find, understand, compare, and reuse models.

Keywords: Model metadata, scientific software, model catalogs, model repositories

1 INTRODUCTION

Models developed by scientists contain important scientific knowledge that should be explicitly captured and disseminated to facilitate model reusability, comparison and composition. Scientists recognize the value of sharing these models to avoid replicating effort and to inspect and reproduce results from other models.

A key issue for reusing scientific models is their dissemination and documentation. Model repositories already exist and are used by many scientists (e.g., CSDMS [Peckham et al. (2013)]; CSDMS (2018)), ESMF [ESMF (2018)], HydroShare [Hydroshare (2017)]). However, they lack important information such as model variables or model processes, which are used by scientist to discern whether the model is appropriate for their analyses or not. Furthermore, once a model (or set of models) is selected, it takes significant effort to understand how to set up a model and how to interpret its results. The OntoSoft software metadata registry [Gil et al (2015); Gil et al (2016), OntoSoft (2018)]
was developed to capture extensive information that is needed by scientists to understand how models work. Most of that information is available, but scattered in publications, manuals, code documentation, and web sites [Essawy et al (2017)]. Having this information organized in a catalog saves scientists a lot of time in understanding and comparing models.

In this paper, we describe several proposed extensions to OntoSoft that capture additional model-specific metadata to facilitate model composition and reuse.

The paper is structured as follows. Section 2 describes a motivation scenario that illustrates the need for model-specific metadata capture (including processes, variables, etc.). Section 3 describes our proposed extensions, and we conclude the paper in Section 4.

2 MOTIVATING SCENARIO

Reusing and executing environmental models often requires significant domain knowledge. In our scenario, Alice, a hydrologist, wants to calculate the water budget of an aquifer by estimating the underwater storage during a period of time. She aims to use MODFLOW-USG [Panday et al (2017)], a groundwater model developed by the United States Geological Survey (USGS) that takes ground water recharge as an input to calculate a water budget. Alice knows that recharge can be derived from precipitation rate, which is information that would not be captured in a model.

Alice starts with a simple interpretation for calculating recharge rates (e.g., recharge rate is one fourth of the times of precipitation rate). She transforms a precipitation file downloaded from the National Oceanic and Atmospheric Administration (NOAA), then she overlaps the map to the grid of the region she is interested in and she finally transforms the result to the appropriate format required by MODFLOW. These data preparation steps are illustrated in the workflow shown in Figure 1 along with a representation of the rest of the inputs and outputs to MODFLOW, among which we can find the target budget she is interested in.

![Figure 1. A simplified workflow showing data preparation steps for MODFLOW to calculate a water budget for an aquifer by transforming precipitation data into a recharge estimate.](image-url)

Once she obtains results by executing MODFLOW, Alice proceeds to visualize the water budget results. She uses the open source software ZoneBudget [Harbaugh (1990)], developed by the USGS to process her water budget file. Alice designates sub regions of interest by specifying zone numbers and a separate budget is computed for each zone. Then a visualization is produced for each zone showing
a visualization of water capture flux and water storage flux along a period of time. These steps are illustrated as a workflow in Figure 2.

![Figure 2](image)

**Figure 2.** The steps to produce a visualization of the budget that results from the MODFLOW groundwater model are shown here as a workflow.

In order to be able to perform these analyses, Alice has to make sure that the variables and formats that she uses for creating the recharge file are consistent with those required by MODFLOW, using the same units and scale.

Alice also needs to understand which of the processes from MODFLOW are relevant for calculating water budget (in this case, recharge and infiltration). This becomes crucial for setting up MODFLOW (i.e., configuring which options to include in the execution of the model) and when composing models. For instance, if there was a river close to Alice’s target aquifer, she would be interested in assessing how the infiltration rate from the river would affect the aquifer’s recharge. Surface hydrology models such as TopoFlow [Peckham et al (2017)] can be used for this purpose, but they often offer different alternatives to calculate infiltration based on available data.

Scientists often explore different model setups or use alternative models to find which models are more accurate or reduce sensitivity or uncertainty. [Carvalho et al. (2017)] present several scenarios where a scientist uses a model and then explores a different setup, model version, or alternative models. These scenarios unveil the need to understand the variables, processes, and methods implemented by each use of a model. They also show that data preparation steps often require significant effort to generate from scratch, and deter scientists from exploring possible model choices.

To facilitate the kinds of analyses where scientists perform activities such as those described in this scenario, model catalogs need to support the following requirements:

1. **Exposing variables of a model:** In order to be able to use a model, it is necessary to describe explicitly as metadata all of its variables (e.g., water budget) and their dependencies. These dependencies become particularly relevant when composing models, as different models may refer to the same variable (e.g., infiltration rate) but calculate it under different assumptions.

2. **Exposing processes of a model:** Variables are associated to model processes (e.g., infiltration, recharge). Each process has one or more variables associated to it (e.g., infiltration rate) and may be calculated using different methods depending on the available information.

3. **A principled representation for variables:** It is necessary to identify if a given model uses or produces a variable which might be used by another model. If variables are named in ways that are not principled, it is difficult for a scientist to understand that two variables in separate models refer to the same physical quantity.

4. **Representing the semantic structure associated with the invocation of a model:** A model can be invoked to use different combinations of processes and using different methods. Each possible invocation needs to be described in terms of the processes and methods used, and the requirements for the input files and how model variables are represented in them as well as their associated metadata. For example, a model may assume that a variable is represented in mm/hour captured at hourly intervals in a NetCDF file.
5. **Describing common data preparation steps used with a model**: These include the most typical pre-processing and post-processing steps needed to carry out useful tasks such as creating input files for a model (e.g., from precipitation in NetCDF to recharge files), or visualizing its results (e.g., using ZoneBudget from a budget output).

---

**Figure 3**: Overview of the OntoSoft software registry

---

3 **A MODEL REGISTRY TO FACILITATE MODEL COMPARISON AND REUSE**

OntoSoft [Gil et al (2016)] is an online software registry for managing, curating, searching and sharing software metadata. It was designed to capture properties of scientific software that are useful for scientists to find, understand, and reuse. Metadata captured by OntoSoft is managed through an ontology [Gil et al (2015)] and organized into six major categories based on information that a scientist would seek about the software: 1) identify software, 2) understand and assess software, 3) execute software, 4) get support for the software, 5) do research with the software, and 6) update the software. Each of these six categories has a few subcategories with specific metadata properties. The metadata properties themselves can be either “recommended” or “optional”.

Figure 3 gives an overview of the OntoSoft user interface for creating and comparing software metadata. Major features include:

- Software metadata ingestion using forms that fill out metadata properties that correspond to activities that are familiar to scientists. Metadata indicators show the degree of completeness of a software entry.
- Software authors can open model metadata entries to crowdsourcing through an access control system.
- Some metadata can be imported automatically from GitHub, such as authors, contributors, and license.
- Software metadata can be exported in HTML, RDF, JSON, enabling users to include the metadata in their publications or attach it to the software.
• Semantic search on metadata properties
• A distributed architecture that enables distributing queries across multiple OntoSoft repositories.
• Comparison of software based on metadata properties

OntoSoft offers a very unique way to structure documentation about models. However, it does not satisfy the requirements described in Section 2 to facilitate model comparison and reuse. Although OntoSoft facilitates model comparison, it is based on general metadata such as the implementation language of the software or its license but does not support the comparison of models in terms of the variables, processes, and methods that they support or the format of their inputs. The remainder of the section describes the extensions of OntoSoft needed to address those requirements.

3.1 Exposing model variables and their dependencies

Different models use heterogeneous variable names in their internal representation. We have started gathering these variables and associating them with the inputs and outputs of models. An example is shown in Figure 4, illustrating on the left the OntoSoft entry for the Penn State Integrated Hydrologic Model (PIHM) [Qu and Duffy (2007)] and a sample of its input variables on the right (out of more than 60 variables). We are extending the OntoSoft ontology so that variables are not only entries in a table associated to a model, but entities that have their own metadata. By using this variable representation, we will enable annotating them with metadata such as their expected units or the interval at which their value has been measured. This representation also enables identifying variable dependencies, which play a critical role when assessing how to couple and compose variables from different models.

![OntoSoft entry for PIHM surface water model](image)

**Figure 4:** OntoSoft entry for PIHM surface water model on the left and its variables shown on the rightmost column on the right.

3.2 Exposing model processes and methods

Model variables are associated to different environmental processes, which may be implemented using different equations and assumptions. Figure 5 illustrates the processes related to infiltration in Topoflow.
These processes are meteorology, subsurface flow in a saturated zone, snowmelt and surface water flow in a network of channels. There are also different ways of implementing a process depending on the method used and the available data. For example, infiltration may be implemented using three different methods: Green-Ampt, Smith Parlange and Richards-Equation (1D). Each method uses different input variables, as shown in Figure 6, but produce the same output variables.

**Figure 5:** An overview of infiltration process in Topoflow. Different processes are highlighted in rectangles, which contain different methods that could be used to implement them (e.g., Richards 1D equation). The output of the process identifies the set of variables associated to it.

**Figure 6:** Comparison of inputs and outputs between components implementing methods for infiltration in TopoFlow. Three methods are available for infiltration, namely, Green-Ampt Method, Smith Parlange Method, Richards Equation Method.

We plan to extend OntoSoft to capture these dependencies between variables, processes and process implementations. Only then will we be able to assess automatically whether two different models can be composed in a sequence, as well as their common variables and processes for which input data is needed.
3.3 Using a unique representation for variables

In Section 3.1 we described the need to capture model variables in order to properly expose their metadata. However, models often refer to their variables using different names, even if they refer to the same concept (e.g., models might refer to temperature as “temp” or “t”). In order to enable linking different model variables together, we are using the Geoscience Standard Name Ontology (GSN) [Peckham (2014)]. GSN includes an extensible list of standardized variable names that follows principled guidelines for concept labelling in the geosciences.

An example of these guidelines can be seen on Figure 7, where GSN describes the mole concentration of phosphorous in sea water. GSN separates quantities from the objects they describe and the operations that can be performed on them. By following these guidelines, a model can uniquely refer to a variable and the transformations that would be required in order to be used by another model.

<table>
<thead>
<tr>
<th>sea_water_phosphorous-dissolved~inorganic _time_derivative_of _mole_concentration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single underscore separates parts of objects</td>
</tr>
<tr>
<td>Title indicates adjective for the object</td>
</tr>
<tr>
<td>Double underscore separates the object and operation/quantity</td>
</tr>
<tr>
<td>The word “of” indicates the end of the operation</td>
</tr>
<tr>
<td>Hyphen binds words into single-object, e.g. carbon-dioxide</td>
</tr>
</tbody>
</table>

Figure 7: Guidelines for variable names, as described in GSN

We have mapped the variables in PIHM to the GSN variables. We have also mapped the TopoFlow model [Peckham et al (2017)], which has more than 100 variables, and are working on others. We are extending OntoSoft to integrate these variable mappings.

3.4 Representing the semantic structure of model invocation

Model variables may be associated with files that are input or outputs of models. Therefore, we need to capture this information if a user needs to understand how to use a model. An initial proposal of the main concepts that are necessary to represent the structure of model invocation can be seen in Figure 8. Var1, var2 and var3 are input variables associated to two input files and var6 is an output variable described in a single output file. File1, File2 and File3 are types of file that can be further described with metadata, such as the standard format used for their encoding or the spatial grid that the model requires.

![Model invocation diagram](image)

Figure 8: Representing the semantics of model invocation in the MINT Model Catalog.

3.5 Capturing data processing workflow fragments

Workflow fragments for data pre-processing and post-processing are commonly used by users executing models, as it helps them prepare data or visualize model results. In order to facilitate model reuse, we are planning to describe these fragments within OntoSoft, as well as linking to them from different models. Some example workflow fragments can be seen in Figure 1 (pre-processing fragments used to prepare the data for MODFLOW) and Figure 2 (post-processing fragment to visualize results using ZoneBudget). [Carvalho et al (2017)] discusses data preparation workflows for MODFLOW.
4 CONCLUSIONS

In this paper we have motivated and described requirements for model understanding, composition and reuse. We have also introduced the extensions for the OntoSoft software registry to improve its descriptions of models in terms of their variables and processes and their respective metadata. We believe that the extended OntoSoft framework will reduce the time to find, understand, compare and reuse models.

ACKNOWLEDGMENTS

This work is funded by the Defense Advanced Research Projects Agency under award W911NF-18-1-0027.

REFERENCES


