*This is an Accepted Manuscript of an article published in Environmental Modelling and Software in June 2017 available online: http://dx.doi.org/10.1016/j.envsoft.2017.01.024*

# Evaluation of the OntoSoft Ontology for Describing Metadata for Legacy Hydrologic Modeling Software

Bakinam T. Essawy [a], Jonathan L. Goodall [a*], Hao Xu[b], and Yolanda Gill[c]

[a] Department of Civil and Environmental Engineering, University of Virginia, 351 McCormick Road, PO Box 400742, Charlottesville, VA, 22908, USA

[b] Data Intensive Cyber Environment Center, University of North Carolina, Chapel Hill, NC

[c] Information Sciences Institute and Department of Computer Science, University of Southern California

[*] To whom correspondence should be addressed (E-mail: goodall@virginia.edu; Address: University of Virginia, Department of Civil and Environmental Engineering, PO Box 400742, Charlottesville, Virginia 22904; Tel: (434) 243-5019)

*Highlights:*

- The OntoSoft Ontology and Portal are evaluated for capturing and sharing metadata for hydrologic modeling software.
- A data pre-processing software workflow for the Variable Infiltration Capacity (VIC) hydrologic model is used as a case study.
- 90% of required OntoSoft metadata was obtained for 13 of the 15 software resources.
- Metadata divided across six sources can now be organized in a constant, machine-readable form.

**Abstract**

Metadata for hydrologic models is rarely organized in machine-readable forms. This lack of formal metadata is important because it limits the ability to catalog, organize, provide attribution for, and identify unique model software; ultimately, it hinders the ability to reproduce past computational studies. Researchers have recently proposed an ontology for scientific software called OntoSoft for addressing this problem. The objective of this research is to evaluate OntoSoft for organizing the metadata associated with a data pre-processing software workflow used in association with the Variable Infiltration Capacity (VIC) hydrologic model. This is accomplished by exploring what metadata are available from online resources and how this metadata aligns with the OntoSoft Ontology. The results suggest that past efforts to document this software resulted in capturing key model metadata in unstructured files that could be formalized into a machine-readable form using the OntoSoft Ontology.

**Keywords:** hydrologic modeling; scientific workflows; metadata; computational reproducibility

## 1. Introduction

Hydrologists use many different computational models, with each model tailored to address specific questions and problems. Hydrological modeling has a long history, and many computational models have decades of development effort and many model versions behind them (Singh et al., 2002). In many cases, there has been splintering of the model code base where the original model code has started to be developed along different paths (e.g., MODFLOW). This causes confusion as to which specific version of software was used for a given modeling application. Further complicating the issue, models often have supporting software beyond the physical process-representations within the model engine itself. This software is used to create input datasets for the model (i.e., data pre-processing) and to analyze or visualize the output from the model (i.e., data post-processing). Organizing and categorizing this broad collection of modeling software so that it is possible to uniquely identify the software used to perform a study is a significant challenge.

The need to better manage the growing volume of software used for hydrologic modeling is central to the larger challenge of computational reproducibility. The common approach for achieving reproducibility has been for researchers to provide sufficient detail within a journal paper's methods section to allow for reproducing the study's results. Growing complexity in computational analyses means this approach is no longer sufficient. Scientific disciplines are trying different approaches to address this problem including model repositories, documentation, on-line model execution, and scientific workflows (De Roure et al., 2009; Essawy et al., 2016; JB et al., 2007; Lud et al., 2006; Roure et al., 2010). One of the main purposes of these approaches is to make models easier to reuse so that scientists can advance the model while achieving reproducibility and strengthening the decisions based upon these models (Cassey and Blackburn, 2006; Hutton et al., 2016; Scholten et al., 2000).

To achieve “reproducible software” (Peng, 2011) for hydrologic modeling, not only does the software and data need to be shared, but also their associated metadata. Metadata is structured information for describing and explaining a digital resource that makes it easier to manage, retrieve, and use that resource (NISO, 2004). Metadata is now a common term for describing data sets, but metadata is less commonly used for describing software. Software for data collection, storage, retrieval, processing, and management has improved greatly, and has significantly contributed to the development of comprehensive distributed hydrological models (Singh et al., 2002). Capturing metadata for hydrologic modeling software is one of the steps required to make the software reproducible (Higgins, 2007; Mcdougal et al., 2016). Little attention has been paid to metadata for describing these software advances. Computational reproducibility also requires other advanced uses of standard software practices beyond metadata tools including version control, strong commenting and documentation, and code modularity.

The limited past efforts to define metadata for hydrologic models have largely focused on describing well maintained and widely used hydrologic models as a single information resource. Like data, however, there is a long-tail of software used to perform and support hydrologic modeling (Heidorn, 2008). Models are often the combination of smaller software modules or components contributed over time by a large number of individuals and groups. Taking a more granular view of models by diving into the details of the software provenance and attempting to capture this provenance using metadata is necessary for many reasons. Some of these reasons include 1) providing attribution for software contributions, 2) maintaining and archiving existing models, 3) providing information that aids in installing and executing models, and 4) ultimately fostering reproducibility.

Metadata for hydrologic models is being collected and recorded, but it is unstructured, informal and distributed. The available metadata for these models are scattered across model documentation, source code repositories, model publication repositories, user forums, and other publically available resources. Metadata such as who created the model, when the model was created, and the type of input and output data for the model can be found from these sources for many scientific models, but are provided in human-readable form. Not having this information in a machine-readable form limits its utility and does not scale well to the growing volume of scientific software. Metadata needs to be in machine readable formats to be most useful (e.g. RDF, XML).

Efforts to establish more formalized, machine-readable formats for hydrologic model metadata include efforts through the Consortium of Universities for the Advancement of Hydrologic Science, Inc. (CUAHSI) HydroShare project and the Community Surface Dynamics Modeling System (CSDMS) project. HydroShare describes metadata for two key modeling concepts: a model program and a model instance. The model program is the software for executing the model and the model instance is the input files required for executing the model (Horsburgh et al., 2015; Morsy et al., 2014; Tarboton et al., 2014). A metadata framework has been proposed for both of these concepts that extend the Dublin Core Metadata Standard. The CSDMS project created a catalog of model programs across the surface dynamics community, which includes hydrology, and captured metadata for these model programs (Peckham and Goodall, 2013; Peckham et al., 2013)

Recent related activities have focused on designing standard metadata for describing software with a particular focus on scientific software. OntoSoft is a project that is part of the National Science Foundation EarthCube Initiative and provides an ontology and portal for addressing the challenge of capturing metadata for scientific software in a formal way (Gil et al., 2016b, 2015). The metadata captured by the OntoSoft Ontology focuses on the knowledge needed for software sharing and reuse (Ratnakar and Gil, 2015). It is recommended for documenting software in scientific papers that follow best practices for reproducible research, open science, and digital scholarship (David et al., 2016; Gil et al., 2016a), and has been used to document scientific software in published articles, e.g., (Fulweiler et al., 2016; Pope, 2016; Yu et al., 2016). OntoSoft is used in the research reported in this paper because it was designed and developed by experts in the semantic metadata community, in contrast to past efforts for hydrologic model metadata that was designed and developed by hydrologists. An underlying question that the research reported in this paper begins to address is whether this more general scientific metadata ontology is appropriate and useful for describing hydrologic modeling software.

The objective of this study is to advance prior efforts for formalizing model metadata in hydrology by evaluating the OntoSoft Ontology as a means for structuring model metadata. The evaluation is performed using a data pre-processing workflow for the Variable Infiltration Capacity (VIC) hydrologic model that consists of multiple software components written by different individuals over time. The VIC model is used by large community; over 500 publications used this model since 1993. The analysis begins by exploring what metadata hydrologists here already captured in unstructured forms. It then shows how this metadata could be organized into structured, machine-readable metadata using OntoSoft Ontology. Therefore, the primary contribution of this work is an evaluation of the OntoSoft Ontology for describing software relevant to hydrologic modeling. This is done by first understanding what metadata for hydrologic modeling software are already embedded in online resources, and then testing how this metadata maps to the OntoSoft Ontology.

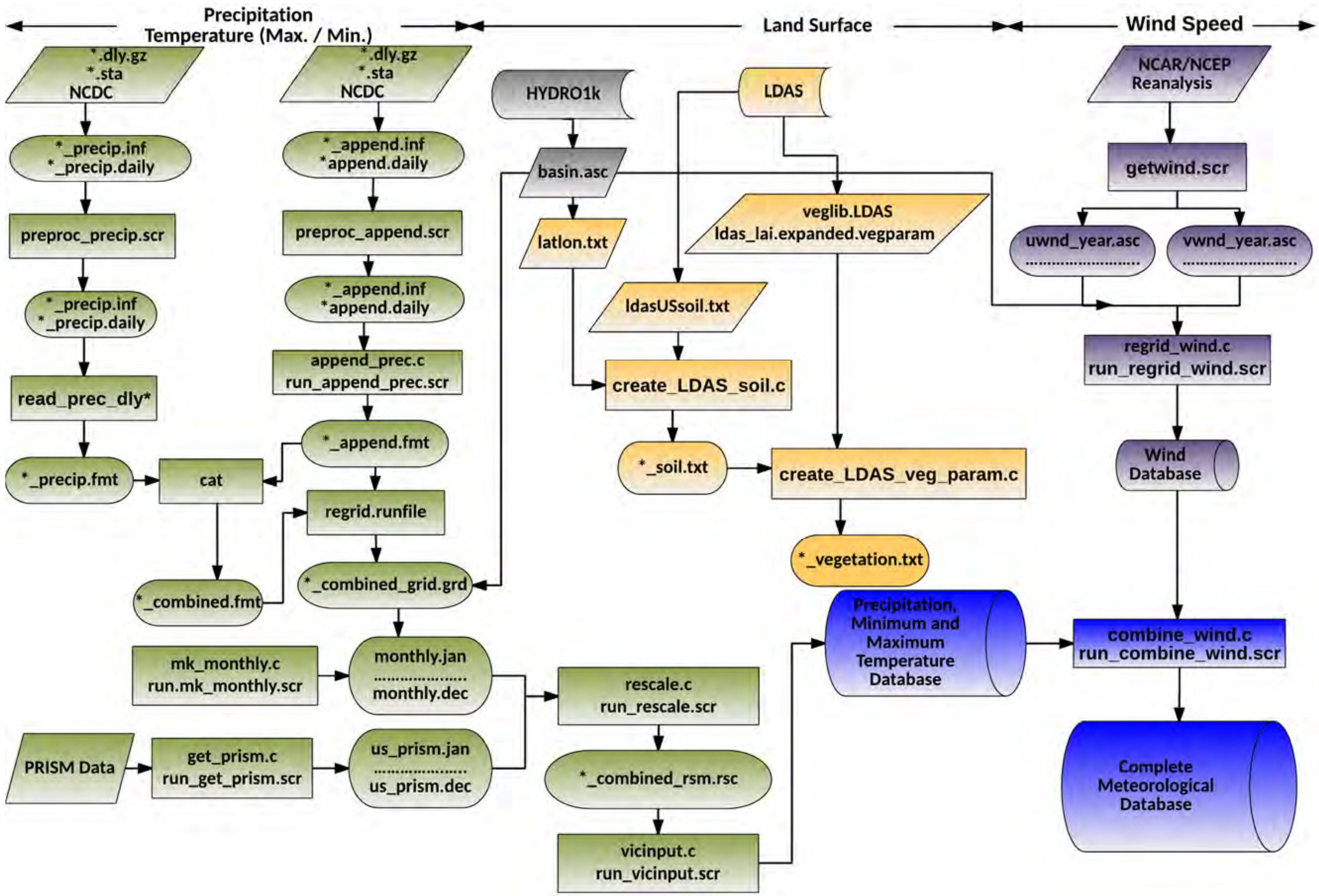
## 1. Background

### *1.1. Variable Infiltration Capacity (VIC) model pre-processing workflow*

VIC is a macro scale hydrologic model that applies water and energy balances to simulate terrestrial hydrology at a regional spatial scale (Liang et al., 1996). Like many hydrologic models, the VIC model requires significant effort to prepare its input data. Figure 1 shows the data processing workflow used to generate the meteorological and land surface input datasets for a VIC model simulation. This workflow consists of a sequence of 15 data processing steps, each step requiring input datasets from different sources, and many of the datasets having unique data models (Billah et al., 2016). These scripts are written with different programming languages including Fortran 77, C, and C++. Shell scripts are used throughout the workflow to execute these steps and perform other commands required to complete the data processing tasks.

The workflow is divided into four categories as shown in Figure 1. The first category of scripts process the precipitation and the air temperature datasets, the second category of scripts process the land surface datasets including topography, soil, and vegetation data, the third category of scripts process the wind speed dataset, and the last category of scripts create the final model input files for meteorological datasets. The datasets processed by the workflow are shown as ovals and include 1) meteorological forcing files (i.e., precipitation, wind, and minimum and maximum air temperature), 2) soil and vegetation parameter files, and 3) basin geospatial files. The primary inputs for the workflow are shown as parallelograms and include datasets from 1) the National Oceanic and Atmospheric Administration (NOAA) National Climatic Data Center (NCDC) (now the National Centers for Environmental Information (NCEI)), 2) the National Center for Atmospheric Research (NCAR) National Centers for Environmental Prediction (NCEP), 3) the National Aeronautics and Space Administration (NASA) Land Data Assimilation System (LDAS), 4) the United States Geological Survey (USGS) HYDRO1K dataset, and 5) the PRISM Climate Group PRISM dataset.

This work addresses the challenges of creating metadata for the individual scripts within the VIC data processing workflow shown in Figure 1. A significant amount of work by other scientists has gone into creating the software within this workflow, and it is important for the authors of this software to receive credit for their work. It is also important for scientific studies that make use of these lower-level scripts to properly document the specific sequence of software used to perform their analysis. One of the benefits of scientific workflow software (Gil et al., 2007) is capturing the provenance of data processing tasks that support scientific modeling. While workflow software can help to better capture the provenance, it is still important to have sufficient metadata for each step within the workflow. Workflow software alone does not provide this metadata. Instead, the metadata must be populated by scientists and the OntoSoft Ontology can be used to structure this metadata. The methodology section illustrates this process by focusing on the metadata population process for one script within the workflow as an example.




168 Figure 1. Data pre-processing workflow for the VIC hydrologic model (adapted from Billah et al, 2016).

*1.2. OntoSoft*

OntoSoft consists of an ontology to describe metadata for scientific software (Gil et al., 2015) and the OntoSoft Portal that serves as a user interface to manage that metadata (Gil et al., 2016b). The premise behind OntoSoft's development is that scientific software captures important knowledge and this knowledge should be transparent and shared widely. OntoSoft's ontology and portal support scientists in capturing the important knowledge encapsulated within scientific software. The OntoSoft Portal simplifies the metadata collection process by asking scientists a series of questions. These questions map to specific properties within the ontology. A property defines a relationship (e.g., authorship) between a subject (e.g., the software in question) and an object (e.g. an author). OntoSoft applies the word "software" broadly to include scripts as well as more complex software such as modeling software.

There are 46 properties in the OntoSoft Ontology, equally divided between required and optional properties. These properties are organized into six categories, shown in Figure 2. Each category has one or more classes for organizing metadata properties. The six OntoSoft categories are: 1) Identify, 2) Understand, 3) Update, 4) Do Research, 5) Execute and 6) Get Support. The *Identify* category provides a unique description for the software. The *Understand* category describes the metadata needed to increase the trust and domain knowledge about the software. The *Update* category has the metadata to track versioning for the software and how the software is being maintained and developed. The *Do Research* category has the metadata for the input and output data required by the software, relations to other software that can be used with this software, and the software citation. The *Execute* category has the metadata related to how to access, install, and run the software. The *Get Support* category has the contact information for the software developer.

**Identify**
*Locate – unique identifier*
- has name (Required)
- has short description (Required)
- has software category (Required)
- has project web site (Required)
- has unique ID (optional)

**Understand**
*Trust – quality and ratings*
- has creator (Required)
- has major contributor (Required)
- has salient qualities (Required)
- has publisher (Optional)
- commitment of support (Optional)
- has adopters (Optional)
- has use information (Optional)
- has use statistics (Optional)
- used in publication (Optional)
- has benchmark information (Optional)
- has funding sources (Optional)
- has rating (Optional)

*Relate – domain Knowledge*
- has domain keywords (Required)
- has uses and assumptions (Optional)
- has use limitation (Optional)
- similar software (Optional)

**Update**
*Track – evolution*
- has software version (Required)
- supersedes (Required)
- superseded by (Required)
- has version release date (Optional)

*Contribute – evolution*
- has active development (Optional)
- has software community (Optional)

**Do Research**
*Experiment – run with other data*
- has input (Required)
- has input parameter (Required)
- has output (Required)
- has relevant data sources (Optional)

*Compose – run with other software*
- has interoperable software (Required)
- has composition description (Optional)

*Cite – scientific publications*
- has preferred citation (Required)

**Execute**
*Access – download*
- has code location (Required)
- has license (Required)
- has executable location (Optional)

*Install – execution requirements*
- has documentation (Required)
- has installation instructions (Required)
- has implementation language (Required)
- has dependency (Required)
- requires average memory (Optional)
- supports operating system (Required)
- has average run time (Optional)
- has other implementation details (Optional)

*Run – testing execution*
- has test data (Required)
- has test instruction (Optional)

**Get Support**
*Discuss – support and community*
- has email contact (Required)
- has software support (Optional)

Figure 2. High-level overview of the OntoSoft Ontology (adapted from Gil et al., 2015).

## 2. Methodology

The first goal of this study is to extract metadata from various sources in order to create a metadata description for a VIC pre-processing workflow. We consider each step in the workflow to be a unique piece of software with its own metadata description. The second goal of this study is to populate the metadata for each step in the workflow using the OntoSoft Ontology. Five sources were used for metadata extraction: 1) the source code prior experience running the software, 2) VIC's official website, 3) the software publication in Zenodo, 4) the VIC documentation, and 5) the VIC user discussion wiki. We did not include publications as a metadata source because, after a search of the literature, we only found one publication that discussed VIC pre-processing workflow in any detail, and this paper did not include any new metadata beyond what we found in the other five sources. We used only online, publically available resources to populate the ontology and did not contact the software developers. The developers likely could have provided additional metadata for this software, however, a motivation of this research is to better understand what metadata was captured and recorded for this legacy software in online, publically available sources. Once the metadata is extracted, it is then used to populate the ontology through the OntoSoft Portal. The completed documentation includes who authored individual components of the workflow, what the goal of each component was, where each component is published, and other important attributes of the software within a formal, machine-readable form.

*2.1. Using the OntoSoft Portal for metadata management*

The OntoSoft Portal was used to insert metadata extracted the from five sources listed above into the OntoSoft Ontology. The OntoSoft Portal presents questions about the software to the scientist, and these questions are mapped to metadata properties in the OntoSoft Ontology. For example, through the OntoSoft Portal, the user is asked "What is the software called?" and the answer to this question is placed as the value for the "has name" property. Table 1 shows all the OntoSoft questions as they appear to the scientist on the OntoSoft Portal, along with the property each answer is mapped to. The table also shows the six categories within the OntoSoft Ontology, the classes for each property, and whether the property is required or optional.

*2.2. Example of metadata extracted from source code*

As an example, the metadata extraction procedure is illustrated for one metadata source (source code and prior experience) and for one software component within the workflow (read_prec_dly). Figure 3 shows a screenshot of how the metadata is encapsulated within the software's source code. Metadata extracted from this source code is shown in Table 2 and includes the name, programming language, author, and description. The description is interesting because it includes additional metadata information about input and output for the software, as well as workflow composition metadata in terms of upstream and downstream software. From prior experience using the software, metadata including the input and output data file names, operating system software dependencies and other relevant metadata was determined and are listed in Table 3.

Once the metadata is extracted, the next step is to map between the extracted metadata and the OntoSoft Ontology. From this one source it is possible to populate 12 of the 46 properties within the OntoSoft Ontology as shown in Figure 4. The OntoSoft Portal played an important role in populating the ontology for the software. Figure 5, provides an example of how the captured metadata from two different sources, the "source code" source discussed earlier and the "software publication website (Zenodo)" source, were mapped to questions presented through the OntoSoft Portal. The programer names, included as a comment within the source code, were set as the software's creators. The name for the software was assumed to be the file name in this case. The description from the source code was used as the short description of the software. Zenodo, which hosts this software as a part of the larger VIC source code repository, provides a DOI for the source code. This DOI was used as the software's unique identifier. The VIC model official website URL is used as the project website for the software.

Using additional sources allows for populating the other properties within the OntoSoft Ontology. This procedure was repeated for all metadata sources and all software components to determine the percentage of both the required and optional metadata properties that could be populated from just these publically available sources. As evident in this example, there is a level of interpretation required to perform this mapping. A discussion of the level of confidence in the mapping is reported in the Results and Discussion section along with the results of the metadata extraction process.

Table 1. OntoSoft Portal question and the associated metadata properties within the OntoSoft

| OntoSoft Portal Question | Metadata Properties | Required and Optional Metadata | Class | OntoSoft Metadata Category |
|---|---|---|---|---|
| What is the software called? | has name | Required | Locate | Identify |
| What is a short description for this software? | has short description | | | |
| What are general categories (keywords, labels) for this software? | has software category | | | |
| Is there a project website for the software? | has project web site | | | |
| What is the DOI or any other unique identifier for this software (or software version)? | has unique ID | Optional | | |
| Who created this software? (e.g., Project, Organization, Person, Initiative, etc.) | has creator | Required | Trust | Understand |
| Are there any additional contributors of note for this software? | has major contributor | | | |
| What useful features of this software are worth highlighting? | has salient qualities | | | |
| Who is the publisher of this software if not the author? | has publisher | Optional | | |
| How can a user get support for the software? (e.g., Report bugs, request features and extensions, etc.) | commitment of support | | | |
| Has the software been adopted in a project, organization or by a person? | has adopters | | | |
| Is there any information about uses of this software (e.g., papers, research labs, etc.)? | has use information | | | |
| Are there any statistics of its use? | has use statistics | | | |
| Are there any publications where the software is used? | used in publication | | | |
| Is there any benchmark information about the software? | has benchmark information | | | |
| What are the funding sources for this software? | has funding sources | | | |
| What are the ratings for this software? | has ratings | | | |
| What are domain specific keywords for this software? (e.g., hydrology, climate) | has domain keywords | Required | Relate | |
| Is there any other similar software that you know of? | similar software | Optional | | |
| What are the recommended uses and assumptions for the software? | has uses and assumptions | | | |
| Are there any constraints on use, situations it is not designed for, simplifications? | has use limitation | | | |
| How is the software being developed or maintained? | has active development | Optional | Contribute | Update |
| Are there any on-line resources for accessing the developer community for this software? (e.g., discussion board, wiki, etc.) | has software community | | | |
| What versions does the software have? | has software version | Required | Track | |

Table 1 (continued). OntoSoft Portal question and the associated metadata properties within the OntoSoft

| OntoSoft Portal Question | Metadata Properties | Required and Optional Metadata | Class | OntoSoft Metadata Category |
|---|---|---|---|---|
| What input files does the software require? | has input | Required | Experiment | Do Research |
| What are the input parameters used for this software? | has input parameter | | | |
| What output files does the software produce? | has output | | | |
| Are there any relevant data catalogs that can be used with this software? | has relevant data sources | Optional | | |
| What other software can interoperate with this one? | has interoperable software | Required | Compose | |
| Is this software typically used with other software in a workflow? (e.g., for visualization, preprocessing, post processing, etc.) | has composition description | Optional | | |
| Is there a preferred publication or citation for this software? | has preferred citation | Required | Cite | |
| What is the URL for the code? | has code location | Required | Access | Execute |
| What license is the code released under? | has license | | | |
| Is there a URL for the executable? | has executable location | Optional | | |
| Is there any on-line documentation about the software? | has documentation | Required | Install | |
| What language(s) is the software written in? | has implementation language | | | |
| What Operating Systems can the software run on? | supports operating system | | | |
| How can one install the software? | has installation instructions | | | |
| What other software does the software require to be installed? | has dependency | | | |
| Are there estimates of how long it takes to run this software on average? | has average run time | Optional | | |
| Are there any memory requirements for this software? | requires average memory | | | |
| Are there any other important details about the implementation of this code (e.g., parallelization, special hardware, etc.)? | has other implementation details | | | |
| Is there any test data available for the software? | has test data | Required | Run | |
| Are there any specific instructions for testing the software? | has test instructions | Optional | | |
| What is the e-mail contact for this software? | has email contact | Required | Discuss | Get Support |
| What is the support offered for this software? | has software support | Optional | | |

```
c     File:          read_prec_dly.f
c     Modified:      09.28.98 by G.M.O.D
c                    Looping rewired to reduce memory overheads.
c                    Code generally cleaned.
c                    Check data and information files are consistent.

c     Programmers:   Greg O'Donnell 1997
c                    Bernt Viggo Matheussen 1998
c                    Univeristy of Washington
c                    Dept of Civil Engineering
c                    Wilcox Hall, Box 352700
c                    Seattle, Washington 98105
c                    tempbvm@ce.washington.edu

      program read_prec_dly

c     This program reads the output from the script preproc_precip.scr
c     and formats the daily precipitation so the regrid program can read them
c     Only the output files from the preproc_precip.scr script (daily data
c     and station info files) are needed.

```

Figure 3. The header information for the source code of one of the software in the VIC pre-processing workflow. This is a comon approach to include unstructured metadata in scientific software.

Table 2. Metadata extracted from the read_prec_dly.f software's source code

| **has name** | **has creator** | **has major contributor** | **has short description** | **has input** | **has composition description** | **has implementation language** |
|---|---|---|---|---|---|---|
| read_ prec_ dly.f | Greg O'Donnell | G.O.M.D | This program reads the output from the script preproc_precip.scr and formats the daily precipitation so the regrid program can read them Only the output files from the preproc_precip.scr script (daily data and station info files) are needed. | daily data | reads output from preproc-precip.scr Provide input for regrid program | FORTRAN 77 |
| | Bernto Matheussen | | | Station info files | | |

Table 3. Metadata captured from experience applying the software

| **has name** | **used in publication** | **has input** | **supports operating system** | **has output** | **Has software dependency** |
|---|---|---|---|---|---|
| read_prec_dly.f | Billah, M.M., Goodall, J.L., Narayan, U., Lakshmi, V., 2015. Using a Data Grid to Support Regional-Scale Hydrologic Modeling. | Prcp.daily<br>Prcp.inf | Linux | Basin_prcp.fmt | F77 |

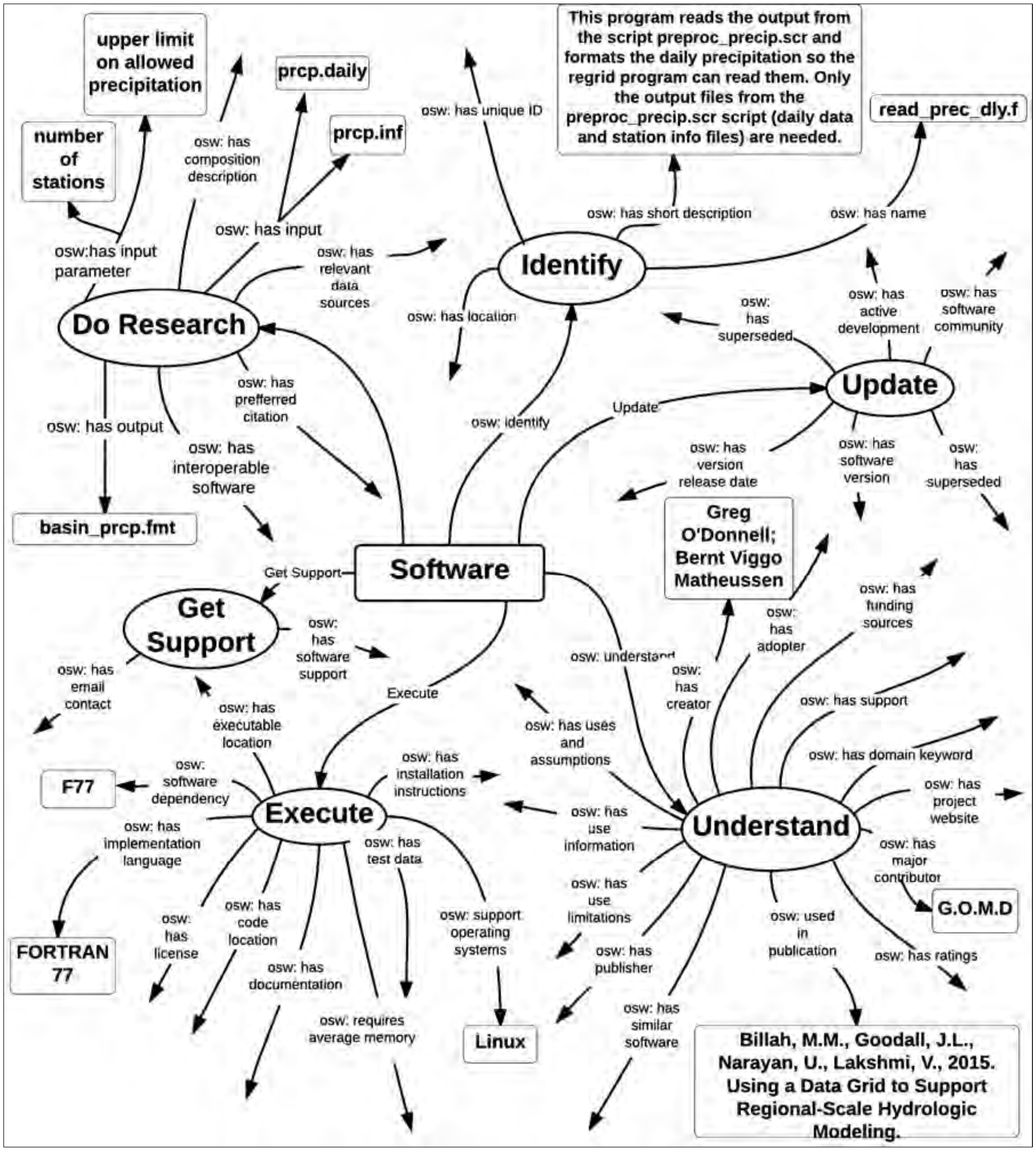



Figure 4. The OntoSoft Ontology for the read_prec_dly software component with properties populated from only one of the five sources: “source code and prior experience.” The prefix “osw” denotes to the OntoSoft Vocabulary namespce.

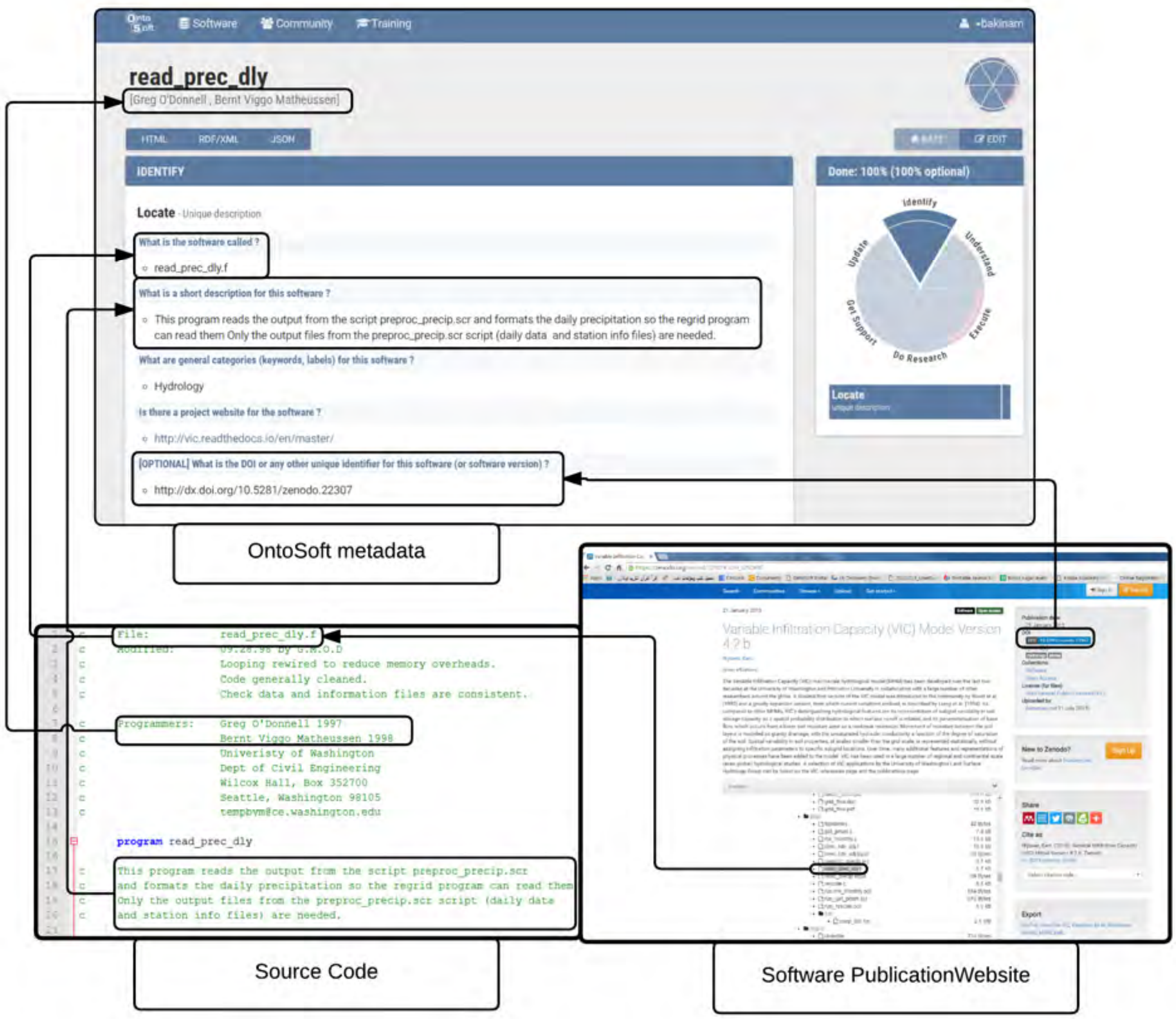



Figure 5. Origin and destination of the captured metadata through the OntoSoft Portal for the identify category.

## 3. Results and Discussion

### *3.1. Results of the Metadata Extraction*

Figure 6 shows the resulting metadata for two of the five OntoSoft categories (Identify and Understand) presented through the OntoSoft Portal for the software component (read_prec_dly) discussed in the Methodology section. The resulting metadata for this software and for the other software components in the VIC data processing workflow are available within the OntoSoft Portal system. Table 4 points to the URLs in the OntoSoft Portal for the 15 software components. The portal provides a user-friendly view of the metadata, but also machine-readable versions of the metadata. The metadata can be viewed using a Resource Description Framework (RDF) eXtensible Markup Language (XML) format or JavaScript Object Notation (JSON) format. These machine-readable formats are built by the system from the data provided by the scientist through the OntoSoft Portal user interface.

Table 4. URL in the OntoSoft Portal for the 15 software within the workflow

| **ID** | **Software** | **OntoSoft Portal URL** |
|---|---|---|
| 1 | preproc_precip | http://ontosoft.org/portal/#browse/Software-11IHopcxMu7x |
| 2 | read_prec_dly | http://ontosoft.org/portal/#browse/Software-3SirBaFht0YN |
| 3 | preproc_append | http://ontosoft.org/portal/#browse/Software-FYMaj4P7bKDb |
| 4 | append_prec | http://ontosoft.org/portal/#browse/Software-hVNbrGnWJ4Zd |
| 5 | run_append_prec | http://ontosoft.org/portal/#browse/Software-GoEvXyadBBVw |
| 6 | regrid | http://www.ontosoft.org/portal/#browse/Software-ZtA35mwlwFmi |
| 7 | mk_monthly | http://ontosoft.org/portal/#browse/Software-DlszQOw6g336 |
| 8 | get_prism | http://ontosoft.org/portal/#browse/Software-vw8DQn2SSnMQ |
| 9 | rescale | http://ontosoft.org/portal/#browse/Software-clQ0WKwjV3Js |
| 10 | vicinput | http://ontosoft.org/portal/#browse/Software-IPXGcujizwTr |
| 11 | create_LDAS_soil | http://ontosoft.org/portal/#browse/Software-AUqV48s3WrgH |
| 12 | create_LDAS_veg_param | http://ontosoft.org/portal/#browse/Software-MZosBxc1Hwl8 |
| 13 | getwind | http://ontosoft.org/portal/#browse/Software-mpNqVzc633VL |
| 14 | regrid_wind | http://www.ontosoft.org/portal/#browse/Software-2QGjMmxS9Du6 |
| 15 | combine_wind | http://ontosoft.org/portal/#browse/Software-ffgkh4iELbOn |


Figure 6. A screenshot for OntoSoft interface showing the captured metadata for the read_prec_dly software within two categories: Identify and a portion of the Trust metadata within the Understand category.

### *3.2. Metadata completeness*

One of the ways the OntoSoft Ontology was evaluated was by recording which OntoSoft properties could be extracted from available online resources for the VIC pre-processing software components. To do this the percentage of metadata completeness for each software within the workflow was calculated and is presented in Figure 7 and Table 5. The results show that for 13 of the 15 software in the workflow, 74% or more of the metadata mapped to terms in OntoSoft. It seemed that there were consistent practices for including metadata within the software with the exception of two of the software (ID 11 and 12). These two software entries are missing important metadata like author name, function of the software, etc. and only include the source code and few comments within the software itself. These poorly described software entries may have been perceived to play a minor role within the overall software system. This also could have been a result of a difference in practice regarding commenting in the source code for these two software, which were both related to soil and vegetation data preparation.

Table 5 also shows that the optional metadata for the Execute category is missing for all software. This category consists of three classes: “Access,” “Install,” and “Run.” These classes depend on the execution of the software with test data like: “has executable location,” “has average run time,” “requires average memory,” and “has test instructions.” These properties assume a standalone executable software, but the software analyzed in this study were lower-level software components within a larger software system. It is likely because the software analyzed was at such a fine granular level within the overall model code that such properties are not well documented. We suspect that some of these metadata would likely be available if we took a higher-level view of the software rather than focusing on components of the software system.

Table 5. Percent completeness of OntoSoft required and optional metadata for each OntoSoft category.

| ID | Software | OntoSoft Metadata Categories: Identify | | Understand | | Execute | | Do Research | | Get Support | | Update | | Average of % complete metadata |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Req | Opt | Req | Opt | Req | Opt | Req | Opt | Req | Opt | Req | Opt | |
| 1 | preproc_precip | 100 | 100 | 100 | 36 | 87 | 0 | 80 | 50 | 100 | 100 | 100 | 100 | 79 |
| 2 | read_prec_dly | 100 | 100 | 100 | 45 | 87 | 0 | 100 | 50 | 100 | 100 | 100 | 100 | 82 |
| 3 | preproc_append | 100 | 100 | 100 | 45 | 87 | 0 | 100 | 0 | 100 | 100 | 100 | 100 | 78 |
| 4 | append_prec | 100 | 100 | 100 | 45 | 87 | 0 | 80 | 50 | 100 | 100 | 100 | 100 | 80 |
| 5 | run_append_prec | 100 | 100 | 50 | 45 | 87 | 0 | 100 | 0 | 100 | 100 | 100 | 100 | 74 |
| 6 | regrid | 100 | 100 | 100 | 45 | 87 | 0 | 100 | 50 | 100 | 100 | 100 | 100 | 82 |
| 7 | mk_monthly | 100 | 100 | 100 | 45 | 87 | 0 | 100 | 50 | 100 | 100 | 100 | 100 | 82 |
| 8 | get_prism | 100 | 100 | 100 | 45 | 87 | 0 | 100 | 50 | 100 | 100 | 100 | 100 | 82 |
| 9 | rescale | 100 | 100 | 50 | 45 | 87 | 0 | 100 | 50 | 100 | 100 | 100 | 100 | 78 |
| 10 | vicinput | 100 | 100 | 100 | 45 | 87 | 0 | 100 | 50 | 100 | 100 | 100 | 100 | 78 |
| 11 | create_LDAS_soil | 100 | 0 | 50 | 27 | 87 | 0 | 80 | 50 | 100 | 0 | 0 | 100 | 50 |
| 12 | create_LDAS_veg_param | 100 | 0 | 50 | 27 | 87 | 0 | 60 | 50 | 100 | 0 | 0 | 100 | 48 |
| 13 | getwind | 100 | 100 | 50 | 45 | 87 | 0 | 100 | 50 | 100 | 100 | 100 | 100 | 78 |
| 14 | regrid_wind | 100 | 100 | 100 | 45 | 87 | 0 | 100 | 50 | 100 | 100 | 100 | 100 | 82 |
| 15 | combine_wind | 100 | 100 | 100 | 45 | 87 | 0 | 100 | 50 | 100 | 100 | 100 | 100 | 82 |

* Req. is required metadata through OntoSoft

* Opt. is for Optional metadata through OntoSoft

Focusing on only the required metadata, the results show that 13 out of 15 software components include 90% or more of the required metadata (Figure 7). The optional metadata completeness varied widely among the software between 30% and 66%. Most of the software were downloaded from the Zenodo website except for the software used for soil and vegetation data processing (ID's 11 and 12), which was downloaded from the VIC official website and was not available through Zenodo. Because this soil and vegetation data processing software was not available from Zenodo, it resulted in missing metadata terms associate with software publication (e.g., “has publisher,” “has preferred citation”). Also, as discussed earlier, the authors of these software did not include as much metadata within the source code comments compared to other software components. This resulted in the software associated with soil and vegetation data processing lacking metadata compared to the other software components.

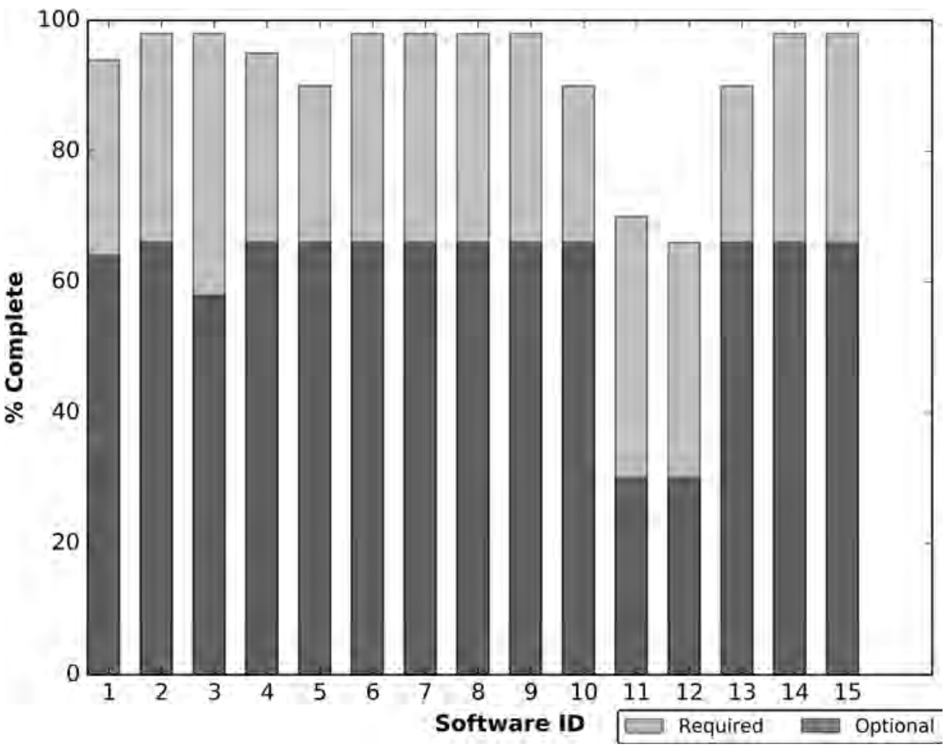



Figure 7. Percent Completeness of OntoSoft required and optional metadata for each software in the VIC pre-processing workflow.

There are common metadata that are missing from all of the software components. Table 6 shows the 10 optional and 1 required properties that were missing for all the software. The one missing required property, “has test data,” was not identified for any of the software through this research, as discussed earlier. It may be necessary to make this an optional rather than required property for more modular software components. Test data should always be included, even to support unit tests of modular components of a larger software system. However, given that this may not have been a common practice in the past, making this optional metadata to support legacy codes may be appropriate. Of the 10 missing optional properties, all are important but none could be captured for this software based on our analysis of available online resources. Some of the missing optional properties may be difficult to populate for other software as well, because they will be heavily dependent on applications of the software to specific use cases (e.g., “has average run time” and “requires average memory”).

Table 6. Common missing metadata across software in the workflow

<table>
<tr><th>Metadata Properties</th><th>Required and Optional Metadata</th><th>Class</th><th>OntoSoft Metadata Category</th></tr>
<tr><td>has use statistics</td><td rowspan="4">Optional</td><td rowspan="4">Trust</td><td rowspan="7">Understand</td></tr>
<tr><td>has benchmark information</td></tr>
<tr><td>has funding sources</td></tr>
<tr><td>has ratings</td></tr>
<tr><td>similar software</td><td rowspan="3">Optional</td><td rowspan="3">Relate</td></tr>
<tr><td>has uses and assumptions</td></tr>
<tr><td>has use limitation</td></tr>
<tr><td>has executable location</td><td>Optional</td><td>Access</td><td rowspan="5">Execute</td></tr>
<tr><td>has average run time</td><td rowspan="2">Optional</td><td rowspan="2">Install</td></tr>
<tr><td>requires average memory</td></tr>
<tr><td>has test data</td><td>Required</td><td rowspan="2">Run</td></tr>
<tr><td>has test instructions</td><td>Optional</td></tr>
</table>

*3.3. Metadata Sources*

Another interesting outcome of the results is a better understanding of the percentage of metadata that comes from each of the five sources used for metadata extraction (Figure 8). The "source code and prior experience" source provided the most metadata. The VIC documentation provided nearly the same amount of metadata as the software publication in Zenodo provided. Collectively, these three sources supplied 80% of the metadata with the other 20% being supplied by the VIC website and user discussion wiki. The results show how the metadata is distributed across the sources and further argues for the need to centralize metadata for hydrologic modeling software.

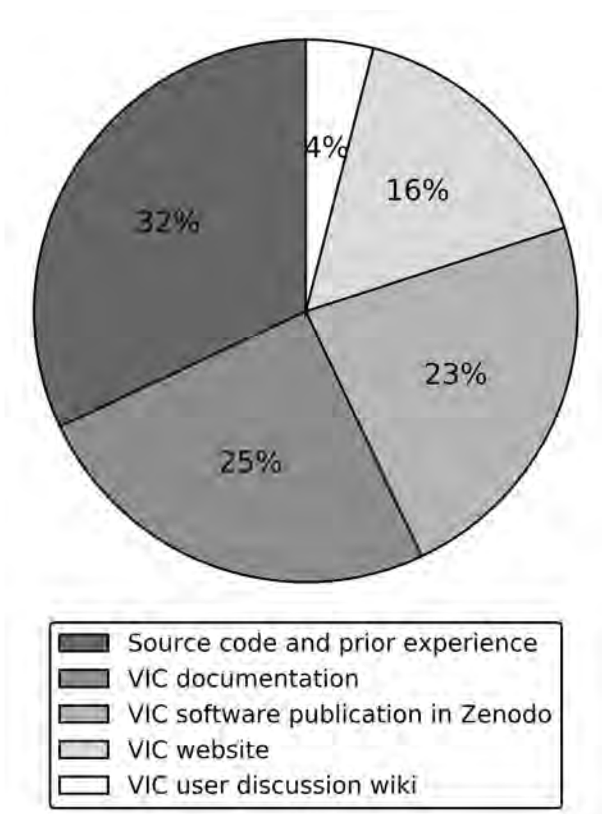



Figure 8. Percentage of extracted metadata coming from each of the five sources

When the metadata source data is broken down by OntoSoft categories, it is clear that some sources play a more major role than others in populating each category's metadata (Figure 9). For example, the VIC website was only used to populate metadata in the Update category. The VIC documentation and documentation were used to populate metadata in five of the six categories; no source was used in all six categories. Interestingly, metadata for Identify, Execute, and Do Research categories came from the same three sources: the VIC publication in Zenodo, the VIC documentation, and the source code and prior experience. This result shows how valuable metadata is being captured now, but even when broken into thematic categories, metadata is still widely distributed across sources.

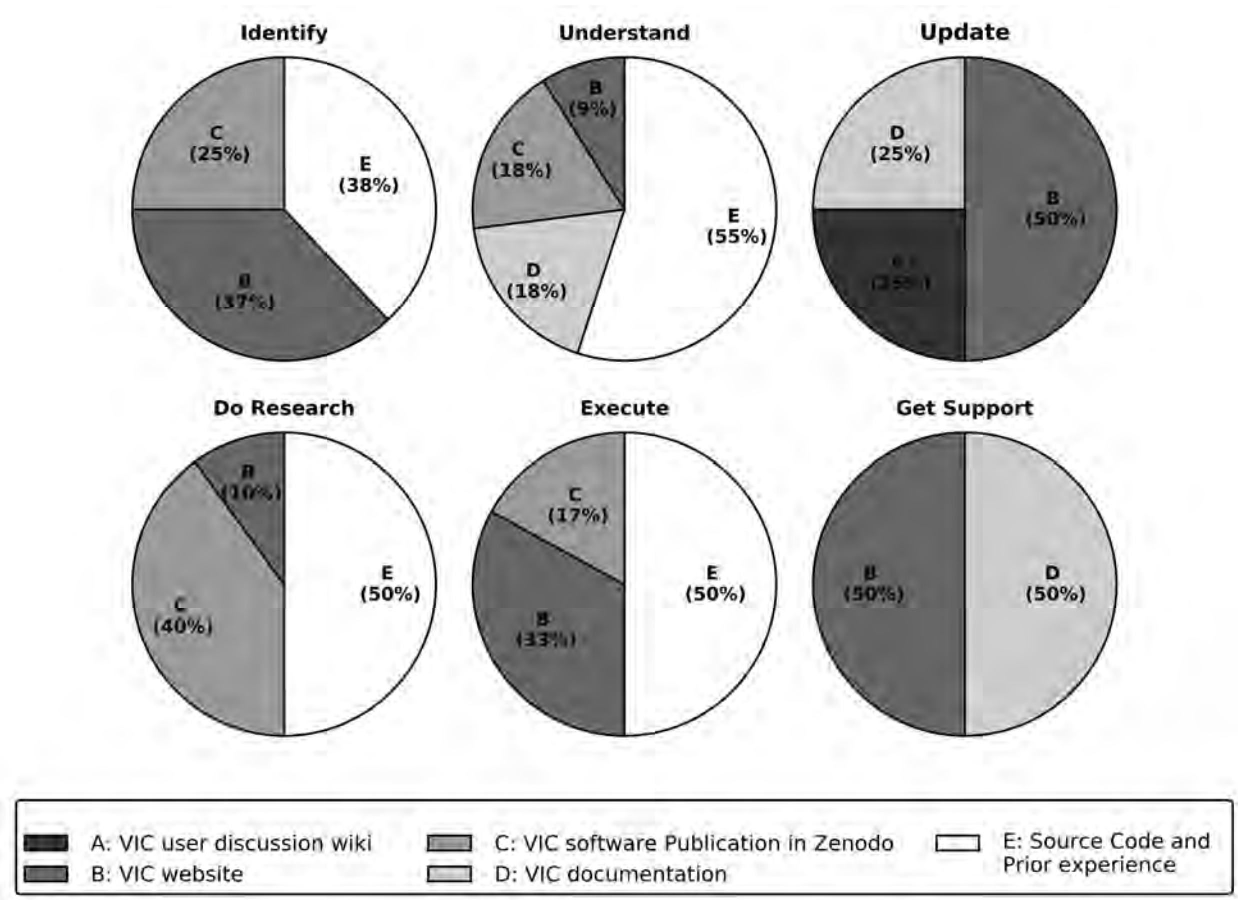



Figure 9. Source for extracted metadata for each OntoSoft Category

### *3.4. Confidence in Metadata Mapping*

Some the mappings for ontology properties are uncertain, meaning it is expected that not all will agree with how extracted metadata was mapped to ontology properties in this study. Table 7 shows the level of confidence the authors had for the ontology property mapping completed in this study. Some properties have high confidence, where it is likely others performing this same metadata extraction exercise would arrive at the same result. Other properties were rated as low confidence, meaning it is likely, in the opinion of the authors, that others may populate these fields differently than what was done in this study. In some cases, the low confidence properties for this study may have higher confidence if this procedure was completed for another model software. In other cases, the low confidence properties were the result of ambiguity as to how metadata from available sources should be mapped to these properties. These properties may require further consideration and explanation for use with hydrologic modeling.

Table 7. Level of confidence in metadata properties populated on OntoSoft

| OntoSoft Category | High Confidence | Low Confidence |
|---|---|---|
| *Identify* | has name<br>has project web site<br>has unique ID | has short description<br>has software category |
| *Understand* | has creator<br>has publisher | has major contributor<br>has short description<br>commitment of support<br>has domain keywords<br>has use limitations<br>has use information<br>used in publication<br>has salient qualities |
| *Update* | has software version<br>has active development<br>has software community | has version release date<br>supersedes<br>superseded by |
| *Do Research* | has input<br>has input parameter<br>has output<br>has preferred citation | has relevant data sources<br>has interoperable software<br>has composition description |
| *Execute* | has code location<br>has license<br>has documentation<br>has implementation language<br>has dependency<br>supports operating systems | has installation instructions |
| *Get Support* | has email contact | has software support |

## 4. Conclusion

This work evaluates the OntoSoft Ontology and portal for capturing and sharing metadata for legacy hydrologic modeling software. The OntoSoft Ontology is designed to focus on scientists rather than software developers (Gil et al., 2015), so it is important for scientists to evaluate the ontology. This work also supports the idea of sharing software and its associate metadata as an additional goal to complement the now commonly accepted idea of sharing data and its associate metadata. To achieve “reproducible software” (Peng, 2011), not only the software and data need to be shared, but also their associated metadata. Sharing software with metadata encourages future scientists to learn and build from prior work by reducing the time and effort to find and understand this prior work. This paper uses a pre-processing workflow for the VIC hydrologic model as a case study for evaluating the OntoSoft Ontology. Metadata was harvested from five sources: 1) Source code and prior experience, 2) Variable infiltration capacity (VIC) model official website, 3) Software published in website Zenodo, 4) VIC documentation for the software, and 5) VIC user discussion wiki. The large amount of effort and time devoted to capturing metadata from these various sources resulted in an improved description of the complex hydrologic VIC model workflow at a detailed level using the OntoSoft Ontology.

Results of the analysis showed that at least 90% of the required OntoSoft metadata properties could be captured from the online sources for 13 of the 15 software components within the workflow. The metadata was somewhat evenly distributed across four of the five sources. This result suggests that the vast majority of the metadata needed to populate at least the required properties in OntoSoft is recorded now by hydrologic modelers, but the information is distributed across sources and stored in unstructured forms. This study also showed that there are common missing properties across all the software used within the workflow. Out of 46 properties in the OntoSoft Ontology, there were 14 optional properties (< 30%) and one required properties (< 3%) missing for all 15 software. Some of the missing properties (e.g., memory size and run time) depend on a specific application of the software (i.e., to model a given domain for addressing a given research objective), and thus will differ from one application to another. Finally, the results of the study also suggested uncertainty in how to populate some of the metadata properties. Some of these terms, labeled as "low confidence" in Table 6, may have had less uncertainty if a different set of software were investigated (e.g., software at less of a fine-grain level than what was used in this study). Other terms may be ambiguous across hydrology models, requiring additional description and guidance.

Some limitations of this study are that (i) while it investigates 15 different software, these are all related to using a single hydrologic model and (ii) the metadata was extracted by one team of hydrologists. Broadening this work to additional geoscience models and having other scientists repeat the metadata extraction process would help to advance the evaluation of OntoSoft for capturing geoscience software metadata. In particular, having other groups of scientists repeat the process would benefit in testing the consistency of the metadata property mapping process. Expanding the effort to other geoscience models would help in improving the evaluation of OntoSoft for representing the metadata necessary for geoscience software more broadly. Despite these limitations, this study contributes both an important and necessary evaluation of OntoSoft as ontology for describing software relevant to hydrologic modeling. It also improves understanding of what metadata is being captured now in available online resources for hydrologic modeling software.

Finally, there are many possible future research goals that could be undertaken to advance the research presented here. 1) OntoSoft could be expanded to better track where metadata recorded within the ontology was obtained. 2) The extraction process, which is now manual and very tedious, could be more automated through text mining approaches, although from this experience we believe manual intervention will continue to be necessary at some level. 3) For the low confidence metadata, a mechanism for crowdsourcing the metadata collection and review (potentially through a user-supplied rating system) would be a helpful feature for gaining confidence in potentially ambiguous metadata. 4) Experiments, where a group of scientists repeat the same procedure outlined in this paper for gathering metadata on the VIC pre-processing workflow and entering it through the OntoSoft Portal, would be a potentially useful way to compare the completeness, confidence, and accuracy of metadata generation across scientists.

Lastly, an underlying premise of this study is that having metadata for software, including for software at a fine-grain level, is useful for increasing transparency and reproducibility in science. Future work could test this assumption by surveying VIC users to better evaluate how metadata presented through the OntoSoft Portal increases their understanding of the VIC software, and how it influences their use and communication of the software with other researchers going forward.

## 5. Acknowledgements

We gratefully acknowledge the National Science Foundation for support of this work under awards ACI-0940841, ICER-1343800, and ICER-1440323. We also acknowledge the assistance of Jeffrey Sadler and Mohamed Morsy from the University of Virginia in preparing the manuscript.

## 6. References


Billah, M.M., Goodall, J.L., Narayan, U., Essawy, B.T., Lakshmi, V., Rajasekar, A., Moore, R.W., 2016. Using a data grid to automate data preparation pipelines required for regional-scale hydrologic modeling. Environ. Model. Softw. 78, 31–39.

Cassey, P., Blackburn, T.M., 2006. Reproducibility and Repeatability in Ecology. Bioscience 56, 958–959.

David, C.H., Gil, Y., Duffy, C.J., Peckham, S.D., Venayagamoorthy, S.K., 2016. An introduction to the special issue on geoscience papers of the future. Earth Sp. Sci. doi:10.1002/2016EA000201.Received

De Roure, D., Goble, C., Stevens, R., 2009. The design and realisation of the Virtual Research Environment for social sharing of workflows. Futur. Gener. Comput. Syst. 25, 561–567. doi:10.1016/j.future.2008.06.010

Essawy, B.T., Goodall, J.L., Xu, H., Rajasekar, A., Myers, J.D., Kugler, T.A., Billah, M.M., Whitton, M.C., Moore, R.W., 2016. Server-side workflow execution using data grid technology for reproducible analyses of data-intensive hydrologic systems. Earth Sp. Sci. 3, 163–175. doi:10.1002/2015EA000139

Fulweiler, R.W., Emery, H.E., Maguire., T.J., 2016. A workflow for reproducing mean benthic gas fluxes. Earth Sp. Sci. 3, 318–325. doi:10.1002/2015EA000158

Gil, Y., David, C.H., Demir, I., Essawy, B.T., Fulweiler, R.W., Goodall, J.L., Karlstrom, L., Lee, H., Mills, H.J., Oh, J.-H., Pierce, S.A., Pope, A., Tzeng, M.W., Villamizar, S.R., Yu, X., 2016a. Towards the Geoscience Paper of the Future : Best Practices for Documenting and Sharing Research from Data to Software to Provenance. Earth Sp. Sci. 1–75. doi:10.1002/2015EA000136

Gil, Y., Deelman, E., Ellisman, M., Fahringer, T., Fox, G., Gannon, D., Goble, C., Livny, M., Moreau, L., Myers, J., 2007. Examining the challenges of scientific workflows. Ieee Comput. 40, 26–34. doi:10.1109/MC.2007.421

Gil, Y., Garijo, D., Mishra, S., Ratnakar, V., 2016b. OntoSoft : A Distributed Semantic Registry for Scientific Software. Proc. Twelfth IEEE Conf. eScience, Balt. MD.

Gil, Y., Ratnakar, V., Ca, R., Garijo, D., 2015. OntoSoft : Capturing Scientific Software Metadata, in: Eighth ACM International Conference on Knowledge Capture, Palisades, NY, 2015.

Heidorn, P.B., 2008. Shedding Light on the Dark Data in the Long Tail of Science. Libr. Trends 57, 280–299. doi:10.1353/lib.0.0036

Higgins, S., 2007. Using Metadata Standards [WWW Document]. Digit. Curation Cent. URL http://www.dcc.ac.uk/resources/briefing-papers/standards-watch-papers/using-metadata-standards#2 (accessed 5.10.16).

Horsburgh, J.S., Morsy, M.M., Castronova, A.M., Goodall, J.L., Gan, T., Yi, H., Stealey, M.J., Tarboton, D.G., 2015. Hydroshare: Sharing diverse environmental data types and models as social objects with application to the hydrology domain. JAWRA J. Am. Water Resour. Assoc. 52. doi:10.1111/1752-1688.12363

Hutton, C., Wagener, T., Freer, J., Han, D., Duffy, C., Arheimer, B., 2016. Most computational hydrology is not reproducible, so is it really science? Water Resour. Res. 50. doi:10.1002/2016WR019285

JB, G., PJ, G., SJ., W., 2007. OpenMI: Open modelling interface. J. Hydroinformatics 9, 175–191.

Liang, X., Lettenmaier, D.P., Wood, E.F., 1996. One-dimensional statistical dynamic representation of subgrid spatial variability of precipitation in the two-layer variable infiltration capacity model. J. Geophys. Res. Atmos. 101(D16), 21403–21422.

Lud, B., Altintas, I., Berkley, C., Higgins, D., Jaeger, E., Jones, M., Lee, E.A., 2006. Scientific workflowmanagement and the Kepler system. Concurr. Comput. Pract. Exp. 18, 1039–1065. doi:10.1002/cpe.994

Mcdougal, R.A., Bulanova, A.S., Lytton, W.W., 2016. Reproducibility in Computational Neuroscience Models and Simulations. IEEE Trans. Biomed. Eng. 63, 2021–2035.

Morsy, M.M., Goodall, J.L., Castronova, A.M., Bandaragoda, C., Greenberg, J., 2014. Metadata for Describing Water Models, in: In Proceedings of the 7th International Congress on Environmental Modelling and Software, DP Ames, NWT QuinnMorsy, M.M., Goodall, J.L., Castronova, A.M., Bandaragoda, C., Greenberg, J., 2014. Metadata for Describing Water Models, in: In Proceedings of the. pp. 978–988.

NISO, N., 2004. Understanding Metadata. Natl. Inf. Stand. Organ. 20. doi:10.1017/S0003055403000534

Peckham, S.D., Goodall, J.L., 2013. Computers & Geosciences Driving plug-and-play models with data from web services : A demonstration of interoperability between CSDMS and CUAHSI-HIS. Comput. Geosci. 53, 154–161. doi:10.1016/j.cageo.2012.04.019

Peckham, S.D., Hutton, E.W.H., Norris, B., 2013. A component-based approach to integrated modeling in the geosciences : The design of CSDMS. Comput. Geosci. 53, 3–12. doi:10.1016/j.cageo.2012.04.002

Peng, R.D., 2011. Reproducible research in computational science. Science. 334, 1226–1227.

Pope, A., 2016. Reproducibly estimating and evaluating supraglacial lake depth with Landsat 8 and other multispectral sensors. Earth Sp. Sci. 3, 176–188. doi:10.1002/2015EA000125

Ratnakar, V., Gil, Y., 2015. OntoSoft [WWW Document]. URL http://ontosoft.org/ontology/software/ (accessed 1.11.16).

Roure, D. De, Goble, C., Aleksejevs, S., Bechhofer, S., Bhagat, J., Cruickshank, D., Fisher, P., Hull, D., Michaelides, D., Newman, D., Procter, R., Lin, Y., 2010. Towards open science : the myExperiment approach. Concurr. Comput. Pract. Exp. 22, 2335–2353. doi:10.1002/cpe

Scholten, Huub, Waveren, R.H. Van, Groot, S., Geer, F.C. Van, Wösten, J.H.M., Koeze, R.D., Noort., J.J., 2000. Good Modelling Practice in water management, in: In Paper Presented on Hydroinformatics. pp. 23–27.

Singh, V.P., Asce, F., Woolhiser, D.A., Asce, M., 2002. Mathematical Modeling of Watershed Hydrology. J. Hydrol. Eng. 7, 270–292.

Tarboton, D.G., Idaszak, R., Horsburgh, J.S., Heard, J., Ames, D., Goodall, J.L., Band, L., Merwade, V., Couch, A., Arrigo, J., Hooper, R., Valentine, D., Maidment, D., 2014. HydroShare: Advancing Collaboration through Hydrologic Data and Model Sharing. Int. Environ. Model. Softw. Soc. 7th Int. Congr. Environ. Model. Software, San Diego, California, USA. www. iemss. org/society/index/php/iemss-2014-proceedings. doi:10.13140/2.1.4431.6801

Yu, X., Duffy, C.J., Rousseau, A.N., Bhatt, G., Álvarez, Á.P., Charron, D., 2016. Open science in practice: Learning integrated modeling of coupled surface-subsurface flow processes from scratch. Earth Sp. Sci. 3, 190–206. doi:10.1002/2015EA000155