HIDE – A Web-based Hydrological Integrated Data Environment

Nimmy Ravindran and Yao Liang
Advanced Research Institute, Department of Electrical and Computer Engineering
Virginia Polytechnic Institute and State University, Virginia, USA
{rnimmy,yliang}@vt.edu

Abstract

One of the challenging tasks in data management and interoperation is efficient data integration. Data integration in scientific community is affected by heterogeneities of the data and diversity of the participating data sources. We have developed a web-based data integration framework, HIDE, for hydrology datasets and data sources. The system is built on an extended XML mediator architecture with modular methodology and software engineering practices of simplicity, extensibility and flexibility. HIDE promotes on-demand data integration and carries functionalities such as search and data access, data integration and remote data visualization. Registration of a new data source to HIDE for integration involves minimal effort of adding a few XML metadata files. The system can help hydrologists to access and share, and dynamically integrate data from diverse data sources quickly. HIDE was primarily built for integrating data from National Weather services with datasets from various autonomous organizations.

1. Introduction

1.1. Motivation

In today’s changing global environment, investigations on trends and spatial distribution of precipitation at a global scale, improvement of hydrologic model forecasts of future floods and water availability play an invaluable role in hazard mitigation (e.g., droughts, and floods), agriculture and food production, human health, municipal and industrial supply, and environmental quality. With the advent of global observing systems (e.g., satellite remote sensing) and global field programs, large amounts of critical multi-variate data have been or are being generated for these studies. These new data measured by many federal organizations and transmitted using various technologies are heterogenous in structures and formats. These large-scale heterogeneous datasets require substantial efforts in data collections and analysis for any effective use. In fact, data preparation generally becomes one of the most time-consuming, expensive, and tedious tasks, and sometimes could take as much as 80% of the overall effort [1]. Clearly, if such challenges and issues in data acquisition, processing, and management are not sufficiently addressed, significant improvement in hydrologic research would not be achieved efficiently.

1.2. Data management methods in scientific community

The burden of accessing and processing heterogeneous data is a great challenge for the hydrology community compared to other scientific areas. There are tools and, information technology frameworks in scientific communities namely DODS (Distributed Ocean Data System) for oceanographic studies, SEEK for ecology, and GEON [9] in geosciences domain. The hydrologic researchers, however, usually relies on geographic information system (GIS) tools (e.g., Arc/Info and ArcView) - to delineate watersheds, obtain river networks, and display spatial images, and Modular Modeling System (MMS) utilities from USGS (U.S. Geologic Survey) to couple disparate environmental models and to manage some temporal and spatial data. Recently, with support from the National Science Foundation, an effort to develop a metadata standard for the hydrologic community is underway through CUAHSI (Consortium of Universities for the Advancement of Hydrologic Science Inc.).

A majority of information systems in a scientific data integration scenario fall into either of the two main categories: data warehousing systems - upload data from individual data providers to a central data repository (for e.g. [10]) and mediator based systems. The mediator systems are based on 3-level architecture...
constructed with respect to numerous relatively autonomous data sources, communicating using a standard protocol, enabling “on-demand” data integration. Mediation systems reduce the coupling between data integration systems and data sources, keeping data redundancy and data synchronizations at a minimal level. In semantic data integration, mediation based systems are mostly used and is demonstrated in [3], [9], and [15]. Another category of Mediation system, MIX [2], for example, uses XML based mediation architecture to achieve spatial data interoperability taking into consideration the flexibility and extensibility of XML.

The key issues to be considered in a scientific data integration scenario are: the diverse nature of datasets in terms of structure, schema, formats and coverage, the methods of data publication and management, and the volatile nature of the data sources [4]. A common approach to resolve semantic heterogeneity is in using ontology (for e.g. [7],[5],[2],[12],[9]). Reference [13] has demonstrated the need for domain ontology and application ontology in geographic dataset integration where the domain ontology models the real world while application ontology is modeled specific to a dataset. In our work, we try to address the challenges of data integration in hydrology regarding nature of data, data heterogeneities (semantic & structure) and autonomy of data sources using our tree based integration model.

1.3. Our work

We have developed an on-demand web based data integration model and system, HIDE, facilitating easier integration of heterogenous data from autonomous data sources in hydrology community. The system was developed for integrating datasets used in hydrological model calibration processes at the NWS (National Weather Service). The new model and system comprises of following four features:

(1). A search engine that coherently searches heterogenous datasets of various geographically distributed data sources on-the-fly using search conditions without uploading data to a centralized repository. The system and the model are built considering essential aspects of extensibility, scalability, uniformity, and transparency.

(2). Browse, analysis, and customization of the formats of retrieved data.

(3). A query engine that can query the datasets from the data sources using a uniform query interface and thus enabling a one - stop integration method.

(4) Visualization of retrieved datasets in the data selection process using our remote data visualization tools.

The system HIDE would significantly facilitate users in conducting search and analysis much more efficiently and thus dramatically reduce the pre-processing time required. Moreover, the user friendly interface of HIDE system promotes a simple data search and integration.

To our knowledge, our system is the first prototype of its kind in the hydrology community. In this paper, we focus on the systematic design and development of our data integration system, providing insights into data integration and interoperability of geo-scientific data sources in practice. The remaining paper is organized as follows. Section 2 describes the design and development of HIDE including architecture and the tree model. Section 3 describes the registration process in HIDE. Finally, we conclude our work in Section 4.

2. HIDE

2.1. System outline

The Hydrological Integrated Data Environment (HIDE) employs a labeled-tree integration model ([6], [8]) – DataNode tree, supported by an extended XML wrapper/mediator architecture (plugin-mediation) (Figure 1) The semantic and structural heterogeneities of datasets are addressed in HIDE by the DataNode tree.

The system identifies each data source/engine as a separate entity with a data access interface. This way, the complexity involved in data storage and search in existing data sources/engines is encapsulated within and is separated from the HIDE system.

The XML based wrappers (Figure 1) of HIDE, handle syntactic differences in the underlying data models, querying capabilities and low-level data access information of the data sources. Following a modular architecture, the mediation engine is further divided into DataModels – temporal, spatial and temporal-spatial model, Transformation engine and Access engines handling the respective responsibilities. The transformation engine performs transformation of the retrieved data from the original data model of the data source to the data models of HIDE. The access engine of mediation is the point of contact of HIDE with the external data source/engine. The module identifies connection strategies to the external data source, the parameters to be supplied, the management of the connections, and the operations to be performed if any error occurs, as well as retrieval of the datasets. The modular nature of the HIDE architecture (Figure 1) adds a great deal of flexibility and extensibility. Here, the overall integration problem is divided into sub-
problems and added as responsibilities to each module. The system uses an XML based metadata model which acts as the input templates describing how an external data source can be integrated to each module. This approach confines the technical issues of integration within each module simplifying the data integration task.

A Remote data visualization function is a significant component in a data integration system. In HIDE, users can make use of the remote visualization tools to view 2D, 2D/3D data transformed plots for temporal/spatial data, and perform many additional functionalities such as data animation, gray colored plots, contoured plots etc. HIDE uses IDL (Interactive Data Language) in conjunction with ION, a time saving computing environment for data analysis, visualization and cross-platform application development, for defining visualization functions.

2.2. Tree based integration model

HIDE adopts a labeled-tree integration model, DataNode tree, for the organization of datasets and data sources semantically and structurally. Building blocks of the DataNode tree, referred to as DataNodes, are defined as the smallest units of information integration in the model. A DataNode can represent an ontological concept (e.g. a database, a file), or a data source. In addition, the DataNodes are modeled as Time-Space-Attribute (TSA) nodes based on the temporal-spatial nature of hydrology data. The Time aspect of the model corresponds to time units, for instance, hourly, daily, monthly, or recent, while the space aspect characterizes spatial and geographical features such as states, watersheds, and latitude-longitude ranges. The “Attribute” aspect of the model is used in defining variables/features represented by the DataNode.

The associations between each DataNodes can either characterize a generic ontology of the domain (domain view) or a logical organization of data (logical view) at the data source (Figure 2a) (see Model formalization). These views defined as relations between the DataNodes are hierarchical forming DataNode sub-trees. Views are glued together for constructing a DataNode tree with the help of special nodes - Adaptation DataNodes (red colored nodes in Figure 2), for instance, nodes of data sources. Thus, we can define a DataNode sub tree, a DataNode path or a view based on The TSA definition of each DataNode involved. For example, the TSA path to the DataNode “Alabama” in Figure 2a is (root,A1,A2,A3,T1,S1) which can be expanded as {root}{precipitation}{USGS}{USGS Water data}{realtime}{Alabama}.

2.2.1. Model formalizations. A DataNode tree can be defined as $DT = (D, E)$, where $D$ is a set of DataNodes and $E$ is a set of unlabelled edges. Let $T$ be a set of time labels $\{t_1, t_2, \ldots\}$, $S$ be a set of spatial labels $\{s_1, s_2, \ldots\}$, $A$ be a set of attribute labels $\{a_1, a_2, \ldots\}$, and $V$ be a set of homonyms and synonyms of TSA labels, in the TSA modeling of the information space. A domain view over TSA is a TSA-DataNode sub-tree, and is a tuple $DV = (Do, Eo, \lambda, V_r)$ where $Do \subseteq D$, $E_o \subseteq E$, $\lambda : Do \rightarrow A$ is a node labeling function, and $V_r : \lambda(Do) \rightarrow V$ is a controlled vocabulary translating function.

Analogously, a logical view of a data source over TSA is a TSA-DataNode sub-tree, $LV := (D_o, D_l, E_l, \lambda, V_r, M_d)$ where $D_l \subseteq D$ is a set of logical TSA-DataNodes; $E_l \subseteq E$ is a set of edges representing parent-child relationship between logical TSA-DataNodes; $\lambda : D_l \rightarrow (TVS \times A)$ is a node labeling function; $V_r : \lambda(D_l) \rightarrow V$ is a controlled vocabulary translating function; $M_d$ an optional z-element tuple := $(y_i, (y_i \in T) \vee (y_i \in S) \vee (y_i \in A))$ is the number of user parameters to the external data source; and $D_o \subseteq D_l$, an adaptation node - the root of the subtree $LV$. The function $M_d$ represents a terminal leaf.
A TSA path over DataNode tree is a sequence
\[ P = (D_1, \ldots, D_n) \] where \( D_i \in D \) (1 ≤ i ≤ n). Similarly a logical path over LV to a logical TSA-DataNode \( D_x \) of a data source is an n-element sequence
\[ P_s(D_x, n) := (D_i, D_{i+1}, \ldots, D_n) \] where \( D_i < D_{i+1} \). Hence, a logical path to a given logical TSA-DataNode of a data source starts with an Adaptation TSA-DataNode and ends with the TSA-DataNode \( D_x \) in question. The label path of \( P_s \) denoted as \( \text{label}(P_s) \) is defined as \( (\lambda(D_1), \ldots, \lambda(D_n)) \).

**2.2.2. Query operation.** Generalization of semantic and structural nuances in a data source through DataNodes and DataNode tree presents transparency to user in queries. A high level query to the system is independent of the structure of the DataNode tree and is applied to TSA model of the tree. These queries are further translated into DataNode sub-queries and executed at intermediate nodes and leaf nodes, respectively. Consider a high level query of the form “What is the current precipitation for a period of 2 days at the site Uchee creek in the state of Alabama?”, which is handled in HIDE as a two-step process, “search-pop”. In the first step, “search”, the search engine of HIDE identifies the appropriate top-level DataNode by performing a best possible match of the keywords supplied by the user to the labels in the dictionaries associated with each DataNode. The dictionaries are configured through our metadata models while constructing the tree. The adaptation DataNodes play a key role in this process as they help in shifting a query from domain view to logical view. Most often, the query strings used by the user may not contain keywords identifying adaptation DataNodes. Hence, these nodes may carry additional information of its child DataNodes or logical views that help in the continuation of the search method. Using this methodology, the search for the above mentioned query is traced to DataNode “Alabama” (Figure 2b) using the search string “USGS real time precipitation from Alabama”. If the search engine is unable to find the best possible match, the user can perform a guided trace through the DataNode tree for the most appropriate DataNode. Each DataNode also carries a link to an interface which can be used later for the second step of query – “pop”.

Every DataNode, specifically leaf nodes, may employ a TSA model of the query interface of the data source as they act as a point of communication for data extraction (Md in model formalization). In case of intermediate DataNodes, the TSA model of the query interface is defined as the join operation of sub-query interfaces of its child DataNodes. The pop operation at a DataNode involves applying the remainder of the parameters of the query string (e.g: Uchee Creek, 2 days) to the TSA model of the external query interface. At the intermediate DataNodes, the pop operation is decomposed into multiple “pops” using the TSA model before sending out the actual queries to external data source. These aggregation pops may involve data extraction and integration from multiple external sources.

HIDE dynamically generates a web based pop interface of each DataNode using its metadata models. The interfaces are complete with conditions, result parameters and validation rules. The validation rules are enforced on the user entered parameters before querying the external source. Upon sending the query, the data is retrieved, transformed and can be viewed or saved.

**2.3. Metadata methodology**

The configuration of the HIDE system involves registration of new data sources, datasets or new domain sets. We simplify the process of registration with the help of our metadata model and can be summarized as addition of new XML files. We have developed four types of XML metadata handling varying complexities involved in the registration.
The XDMS - XML Description Model Specifications are used for describing individual DataNodes and DataNode tree of the system. The description includes but not limited to, a unique label identifying the TSA nature of the data and a small dictionary of keywords best describing the node. These metadata are used by HIDE in generating the DataNodes, views and finally constructing the DataNode tree. The second category, XOMS – XML operational model specification defines the pop interface, and information for translating the pop query to the external data source query. The specification describes data fields for each parameter such as conditions and results, and adequate validations rules associated. As the data models of datasets from various sources can differ immensely, we used a unified approach Time-Spatial data model for the datasets in HIDE. The information for data transform of the extracted datasets to HIDE data models is described by XTMS – XML transformation specification. Finally, the XFD – XML format definition files defines the data formats of the extracted datasets.

The transformation engine uses XFD metadata for understanding the data syntax, and the XTMS metadata files for the temporal/spatial characteristics of the data in the transformation process. Similarly XOMS metadata is used by Access Engine in to handle all the complexities of making a lower module connection to the external data source.

In the registration process, these metadata are developed and supplied to the system, which are used by object factories for the creation of necessary objects (DataNodes, Access engines, transformation engines) (Figure 3). In the access engine module, based on the type of external query interface (http, sql etc), defined in the XOMS as part of the translation details, an appropriate access engine object is constructed by the corresponding object factory. For example, if the external data source supports a web based interface, the access engine factory creates an “HttpAccessEngine” object for specifically handling the requirements of HTTP access. Similar factory objects and methods are also employed at the data model and transformation engine module.

Figure 3. An object level representation of the object factory at each level and its relationships with the metadata files.

3. Registration and integration Levels

One of the unique characteristics in HIDE is the capability to provide different levels (i.e., primary and secondary) of integration required for different data sources and users. The primary level integration can be adopted for complete integration of an external data source to HIDE system. This means that users would be performing query, data analysis and visualization without ever leaving the HIDE system. On the other hand, the secondary level integration is a partial integration of an external data source to HIDE system, in which any data analysis and visualization features of the HIDE system are not incorporated. Here, users can find the datasets but has to manually perform the data extraction and analysis.

There are multiple uses for the integration levels. If we would like to integrate a highly complex external data system which could not be completely defined in HIDE system (e.g: security concerns) and would like to limit the degree of integration, the secondary level integration would be an ideal choice.

A registration of a data source or a dataset in HIDE can be explained as a four step process as described below.

1. Identify the level of integration.
2. Identify the views, DataNode(s) in a DataNode tree where the integration is to be performed.
3. Create a DataNode sub-tree for the data source/dataset. The creation of DataNode sub-tree requires the development of the necessary metadata (XDMS, XOMS, XFD, and XTMS) for each DataNode. At the domain level, the creation of
DataNode sub-tree is based on the domain hierarchy or the taxonomical relationships between the DataNodes. At the logical level, creation of sub-tree is based on the structural organization of the data.

4. Integrate the new DataNode sub-tree to a parent DataNode with necessary changes.

4. Conclusions

In this paper, we have systematically described design and development of HIDE, a web-based data integration system for the integration of heterogeneous hydrology datasets from diverse data sources based on an XML mediator architecture. The system uses a data integration model – DataNode tree to represent datasets and data sources. Our system follows a modular design approach with many software engineering practices emphasizing flexibility, simplicity and extensibility. The HIDE currently is implemented in Java, and has incorporated several data sources validating its design and functionalities. The uniqueness of our system design makes it possible to integrate user’s requested data on-demand from diverse data sources, and remotely analyze and visualize the selected data sets effectively and efficiently. More thorough system performance evaluation is to be conducted as our future work.

5. Acknowledgements

We are grateful to Professor Xu Liang of the University of Pittsburgh and Mr. Thomas Adams at the National Weather Service for their insights and help on hydrological knowledge and data sets. This work was partially supported by NOAA University Corporation For Atmospheric Research under UCAR Subward No. S04-44686.

6. References


