Observations Data Model 2: A community information model for spatially discrete Earth observations

Jeffery S. Horsburgh a, *, Anthony K. Aufdenkampe b, Emilio Mayorga c, Kerstin A. Lehnert d, Leslie Hsu d, Lulin Song d, Amber Spackman Jones e, Sara G. Damiano b, David G. Tarboton e, David Valentine f, Ilya Zaslavsky f, Tom Whitenack f

a Department of Civil and Environmental Engineering, Utah Water Research Laboratory, Utah State University, 8200 Old Main Hill, Logan, UT 84322-8200, USA
b Stroud Water Research Center, 970 Spencer Road, Avondale, PA 19311, USA
c Applied Physics Laboratory, University of Washington, 1013 NE 40th Street, Seattle, WA 98105-6698, USA
d Lamont-Doherty Earth Observatory, Columbia University, 61 Route 9W, Palisades, NY 10964-8000, USA
e Utah Water Research Laboratory, Utah State University, 8200 Old Main Hill, Logan, UT 84322-8200, USA
f San Diego Supercomputer Center, University of California San Diego, 9500 Gilman Drive #0505, La Jolla, CA 92093-0505, USA

A R T I C L E   I N F O

Article history:
Received 1 July 2015
Received in revised form 29 January 2016
Accepted 29 January 2016
Available online 8 February 2016

Keywords:
Observations
Information model
Data management
Interoperability
Cyberinfrastructure
Data publication

A B S T R A C T

Integrated access to and analysis of data for cross-domain synthesis studies are hindered because common characteristics of observational data, including time, location, provenance, methods, and units are described differently within different information models, including physical implementations and exchange schemas. We describe a new information model for spatially discrete Earth observations called the Observations Data Model Version 2 (ODM2) aimed at facilitating greater interoperability across scientific disciplines and domain cyberinfrastructures. ODM2 integrates concepts from ODM1 and other existing cyberinfrastructures to expand capacity to consistently describe, store, manage, and encode observational datasets for archival and transfer over the Internet. Compared to other systems, it accommodates a wider range of observational data derived from both sensors and specimens. We describe the identification of community information requirements for ODM2 and then present the core information model and demonstrate how it can be formally extended to accommodate a range of information requirements and use cases.

© 2016 The Authors. Published by Elsevier Ltd. This is an open access article under the CC BY license (http://creativecommons.org/licenses/by/4.0/).

Software availability

Name of software: Observations Data Model 2 (ODM2)
Developers: Jeffery S. Horsburgh, Anthony K. Aufdenkampe, Emilio Mayorga, Kerstin A. Lehnert, Leslie Hsu, Lulin Song, Amber Spackman Jones, Sara G. Damiano, David G. Tarboton, David Valentine, Ilya Zaslavsky, Tom Whitenack

Contact: Jeffery S. Horsburgh; Address: 8200 Old Main Hill, Logan, UT 84322-8200, USA; Email: jeff.horsburgh@usu.edu
Year first available: 2015
Required hardware and software: ODM2 is available for use with Microsoft SQL Server, MySQL, PostgreSQL, and SQLite on Windows, Macintosh, and Linux based computers. Information about additional software available for working with ODM2 is available at https://github.com/ODM2/ODM2.
Cost: Free. Software and source code are released under the New Berkeley Software Distribution (BSD) License, which allows for liberal reuse. All source code, examples, and documentation can be accessed at https://github.com/ODM2/ODM2.

* Corresponding author.
E-mail addresses: jeff.horsburgh@usu.edu (J.S. Horsburgh), aufdenkampe@stroudcenter.org (A.K. Aufdenkampe), mayorga@apl.washington.edu (E. Mayorga), lehnert@idso.columbia.edu (K.A. Lehnert), lhsu@idso.columbia.edu (L. Hsu), song@idso.columbia.edu (L. Song), amber.jones@usu.edu (A.S. Jones), sdamiano@stroudcenter.org (S.G. Damiano), dtarb@usu.edu (D.G. Tarboton), valentin@sdsc.edu (D. Valentine), zaslavsky@sdsc.edu (I. Zaslavsky), twhitenack@sdsc.edu (T. Whitenack).

http://dx.doi.org/10.1016/j.envsoft.2016.01.010
1364-8152/© 2016 The Authors. Published by Elsevier Ltd. This is an open access article under the CC BY license (http://creativecommons.org/licenses/by/4.0/).
1. Introduction

As volumes of Earth observations increase, so does the importance of their efficient management and use. In the past several years, a number of cyberinfrastructures have emerged for sharing spatially discrete Earth observations data, including the Consortium of Universities for the Advancement of Hydrologic Science, Inc. (CUAHSI) Hydrologic Information System (HIS) (Tarboton et al., 2009), the Critical Zone Observatory Integrated Data Management System (CZOData) (Zaslavsky et al., 2011), the Integrated Earth Data Applications (IEADA) and EarthChem system (Lehner et al., 2011a, 2004, 2009), and the Integrated Ocean Observing System (IOOS) (IOOS, 2010a; Lubchenco, 2010). These systems are built using the principles of service-oriented architecture (SOA) (Josuttis, 2007; Goodall et al., 2008) and rely on standard data encodings and, in some cases, standard semantics for classes of geoscience data. A core focus of most of these systems is on publishing or sharing data on the Internet via web services and domain-specific encoding or markup languages.

While these systems have made considerable progress in making data interoperable and available, it still takes a knowledgeable investigator substantial effort to discover and access datasets from multiple domain-specific repositories for analysis because of inconsistencies in the way the different domain systems describe, encode, and share data. First, data structures used by existing domain cyberinfrastructures are often insufficient to store or describe the entire range of Earth observations. Here we refer to the sufficiency of metadata with respect to both data discovery and ultimate use. For example, data structures and encodings used by the CUAHSI HIS contain the necessary metadata to describe time series of in situ observations made at point locations such as streamflow gages and weather stations. However, they are inadequate for water quality or solid Earth geochemical samples taken in the field and analyzed later in a laboratory because existing method and sample descriptions do not contain all of the needed metadata and are not extensible to allow, for example, important data structures such as sample fractions and sub-sample parent-child relationships. Conversely, the EarthChem system contains the necessary metadata elements and structures to effectively describe observations derived from ex situ analysis of geochemical samples, but is not well structured to support time series of observations from in situ sensors.

Yet, there are many research scenarios that require efficient integration of these data types across different domains of observational Earth science. For example, understanding a soil profile’s geochemical response to extreme weather events requires integration of hydrologic and atmospheric time series with geochemical data from soil sample fractions collected over various depth intervals from soil cores or pits at different positions on a landscape. Similarly, understanding spatial and temporal patterns in suspended sediment fluxes, sources, and associated contaminants in response to land use and climate change requires close integration of hydrologic time series with a variety of geochemical data analyzed in different laboratories on separate sample fractions (e.g., acid extract of fine sediments for heavy metals, solvent extract of whole water for organic contaminants, dried filter for suspended solids concentration). Currently, integrated access to and analysis of data for such studies are hindered because common characteristics of observational data, including time, location, provenance, methods, and units are described using different constructs within different systems. Integration requires multiple syntactic and semantic translations that are, in many cases, manual, error-prone, and/or lossy. Management of data across multiple repositories/systems is similarly complicated, and data managers such as those managing diverse datasets from large projects like Critical Zone Observatories may prefer a single schema that enables a more efficient data integration strategy (e.g., more straightforward, lossless, sustainable, reusable, etc.) rather than managing multiple different domain databases with different schemas.

While there are many properties of observations that are common across the various types of observational data acquired and used within the geosciences, each domain also presents observation types that are unique. In many instances, data structures have been built to support the most common types of observations within a specific domain, without consideration of the broader context of available observations across domains, leading to substantial syntactic and semantic heterogeneity in observational data representations. Semantic and syntactic heterogeneity are major hurdles to be overcome, especially across data types and scientific domains (Beran and Piasecki, 2009; Horsburgh et al., 2009; Hankin et al., 2010a, 2010b). Because many systems (including those mentioned above) already have their own existing data structures and/or database implementations, one solution to achieving interoperability between systems is to agree upon a common information model to which data in the existing systems can be mapped. The common element, then, across systems and the services that they provide is the information model, with physical implementations within various file systems and databases for data storage, within extensible markup language (XML) schemas and file formats for data transfer, and within web service interfaces that provide access to data. Indeed, overcoming heterogeneity and achieving interoperability and more reliable data integration within SOAs depends on standardizing descriptions of common characteristics within a common information model and well-defined interfaces and data encodings that implement it.

An information model is a representation of concepts, relationships, constraints, rules, and operations that specify the semantics of data for a chosen domain of discourse (Lee, 1999). At its simplest level, an information model defines the domain’s entity types and their properties, relationships, and allowed operations on the entities. In relational database design terminology, an information model is essentially equivalent to a “conceptual database model” (e.g., Connolly and Begg, 2005). In a relational database implementation of an information model, entities become tables and their properties become table columns. More generally, an information model provides a sharable, stable, and organized structure of the information requirements for a domain context, without constraining how that description is mapped to an actual implementation in software (Fulton, 2006). There may be many mappings of the information model. Such mappings are called data models, irrespective of whether they are object models, entity relationship models such as those used by relational databases, or XML schemas. The fundamental elements within the information model are based on the domain of discourse to be described and how the model will be used—e.g., to support data discovery or data storage. For example, a rich set of descriptive metadata about the variables that were observed and the context within which an observation was made is fundamental for both discovering and interpreting observational data (Madin et al., 2007). The information model behind a data system is thus critically important to the effectiveness and interoperability of any cyberinfrastructure.

Domain-agnostic information models for observations have been developed and standardized (e.g., the Open Geospatial Consortium’s (OGC) Observations and Measurements (O&M) standard (Cox, 2010)). While they provide a general framework and key constructs for describing different types of observations, these models are expected to be used as a common basis for domain-specific profiles of information exchange. In this paper, we demonstrate how a single, detailed profile can be developed for a wider range of geoscience domains. As architects and developers of
We sought to develop a model that is sufficient to accompany observational datasets for them to be archivable and interoperable across multiple disciplines and systems to support a broad class of Earth science data management of both hydrological observations data with a specific focus on time series structures (Taylor, 2014). ODM was used to structure a relational database schema for storing local copies of observational data as examples of how ODM2 can be implemented and end with a discussion of implications for data management within geoscience domain cyberinfrastructures.

2. Spatially discrete Earth observations in existing cyberinfrastructures

A spatially discrete Earth observation is one that applies to an entire vector feature or phenomenon represented by one or more geometric primitives (i.e., points, curves, surfaces, or solids) (ISO, 2003; ISO, 2005). In practice, the most widely implemented geometric primitives are 0-to-2-dimensional point, line, and polygon “simple features” and collections thereof (Herring, 2011). Examples include time series of hydrologic observations from a stream gage, water temperature observations recorded by a sensor onboard a buoy drifter, total ecosystem respiration over a watershed area, sediment samples from a lake bed, and solid Earth geochemical samples taken from a soil core. Observations may be generated by in situ sensors that make measurements regularly (e.g., at high sampling rates with constant intervals) or sporadically (e.g., triggered by environmental conditions). Or, they may result from ex situ analysis of environmental samples or sample fractions. In practice, it is likely that multiple types of observations are made at a particular monitoring location — e.g., water quality samples collected at a stream gage where streamflow and water temperature are also measured in situ.

The scope of ODM2 and of this discussion specifically excludes spatial fields, including 2D rasters and 2-3D irregular tessellations (ISO, 2005; Nativi et al., 2008; Unidata, 2014), though it is recognized that the discrete and continuous geographic phenomena are not mutually exclusive (ISO, 2005). Many cyberinfrastructures exist in the geoscience domain that encompass the publication of spatially discrete Earth observations. The following sections provide an overview of several of these systems that informed the development of ODM2. Although only a cross section of existing systems within the geosciences, these cyberinfrastructures, their requirements, and their deficiencies are illustrative of much of the domain.

2.1. CUAHSI HIS

Over the past several years, the CUAHSI HIS has achieved tremendous success in advancing the interoperability of hydrologic observations made at fixed monitoring points (e.g., streamflow gauges, weather stations) through the development and standardized use of the Observations Data Model (ODM) (Horsburgh et al., 2008), which was implemented for data storage in a relational database (ODM Version 1.1.1), translated into an XML schema for data transfer via web services (WaterML 1.1 and WaterOneFlow, respectively) (Zaslavsky et al., 2007). This then led to the development of WaterML 2.0 as an OGC standard for the representation of hydrological observations data with a specific focus on time series structures (Taylor, 2014). ODM was used to structure a central metadata catalog database that supports data discovery services (Whitenack et al., 2010), and was implemented as a relational database schema for storing local copies of observational data retrieved from HydroServers by the CUAHSI HIS HydroDesktop client application (Ames et al., 2012). The information model is also supported by a set of controlled vocabularies (Horsburgh et al., 2014) that promote semantic consistency in the language used to describe observations.

Since 2005, the EarthChem project (http://www.earthchem.org/) has developed and operated a suite of data systems and services for solid Earth geochemical data that are acknowledged worldwide as a leading resource for sample-based analytical data. EarthChem’s systems include: the Petrologic Database (PetDB — http://www.earthchem.org/petdb); the EarthChem Portal, which provides a central access point to data in federated databases; the EarthChem
Library, which is a repository and publication agent for geochemical datasets; and Geochron (http://www.geochron.org/), which provides data management for geochronological and thermochronological data. One of EarthChem's main achievements has been the community-driven development of metadata standards for geochemical data through workshops, the Editors Roundtable (Goldstein et al., 2014), and collaboration with other geochemical data systems (Lehnert et al., 2007). EarthChem has developed templates for investigators to format their data and assemble metadata according to these standards. The EarthChem development group has also led the development of standards for identification, registration, and documentation of physical samples in the geosciences, creating the International Geo Sample Number (IGSN) and the System for Earth Sample Registration (SESE (www.geosamples.org; Lehnert et al., 2005; Lehnert et al., 2011b).

EarthChem data collections employ a modified version of the data model that was developed for the PetDB and Geochemistry of Rocks of the Oceans and Continents (GEOROC) databases (Lehnert et al., 2000), and that has been adopted by various other geochemical databases such as the Metamorphic Petrology Database (HetPetDB — Spear et al., 2009) and the Critical Zone Geochemical Database (CZChemDB — Niu et al., 2011). In order to develop interoperability among geochemical databases and allow users to seamlessly discover and access data in distributed systems, EarthChem developed an XML schema for sample-based geochemical data that is now used as the standard data transfer protocol for databases in the EarthChem federation. EarthChem's information model contains information necessary to describe geochemical samples, subsamples, sample fractions, the observations derived from them, and provenance of the data. However, despite its successes for supporting specimen-based data, the EarthChem information model needed improvement because it lacked important informational elements for describing subsamples. Additionally, since all data in EarthChem are related to a discrete specimen, the EarthChem information model was inappropriate for time series-based information.

2.3. Critical Zone Observatory Integrated Data Management System (CZOData)

Developed over the past several years, the prototype CZOData system focused on publishing hydrologic time series observations collected at Critical Zone Observatory (CZO) sites and leveraged SOA approaches and software components developed by the CUAHSI HIS project (Zaslavsky et al., 2011). The prototype CZOData system used a group of ASCII files that followed a relatively simple, text-based format called "CZO Display Files." The Display File format was created to provide a simple way for CZO site data managers to publish their data. The Display File format was based in large part on the CUAHSI HIS ODM 1.1.1 information model. Once published at an individual CZO web site, Display Files were automatically harvested into the CZO Central Data Repository at the San Diego Supercomputer Center (SDSC). The harvested data were then validated against shared vocabularies and a variable ontology, archived in a set of ODM 1.1.1 databases established for each CZO, and then published via standard CUAHSI WaterOneFlow web services that transmitted the data using WaterML 1.1 and WaterML 2.0. CZO shared vocabularies were also adapted from the CUAHSI HIS ODM 1.1.1 controlled vocabulary management system and established semantic conventions within the CZOData system. However, because of the lack of an information model that could accommodate both time series and specimen-based data, the prototype CZOData system lacked the ability to integrate both hydrologic time series data and geochemical and other specimen-based datasets, which are commonly collected within the CZOs.

Efforts are now ongoing to operationalize the prototype CZOData system and to improve support for specimen-based datasets. Enabling data integration at the information model and storage level would permit a more flexible data publication infrastructure and additional types of cross-domain database queries.

2.4. Integrated Ocean Observing System (IOOS)

IOOS is a U.S. federal-regional partnership enhancing the nation’s ability to collect, deliver, and use data and information needed to better understand our oceans, coasts, and Great Lakes, and to contribute to global ocean observing efforts (Lubchenco, 2010). Central to IOOS is the presence of a Data Management and Communication (DMAC) subsystem capable of delivering real-time, delayed-mode, and historical data for in situ and remotely-sensed physical, chemical, and biological observations, as well as model-generated outputs, integrated across many diverse providers and data types (IOOS, 2010a; IOOS, 2010b; Haines et al., 2012). IOOS DMAC is an evolving, loosely coupled system-of-systems focused on the implementation and refinement of existing community standards for web service interfaces, metadata, and data encodings (IOOS, 2010a) to enable interoperable distribution of marine data. It is built on ODM1.1.1-supported standards and software stack for ocean data interoperability composed of the netCDF file format, Climate and Forecast (CF) conventions, Unidata Common Data Model (CDM) and OPeNDAP data access protocol (Hankin et al., 2010b; Unidata, 2014; Nativi et al., 2008), for both continuous and discrete data; and 2) complementary OGC geospatial and sensor standards, particularly the Sensor Web Enablement (SWE) suite of standards (Bröning et al., 2011) for in situ data, with IOOS customizations to sensor metadata conventions and data encodings that include harmonization with CF conventions and CDM Discrete Sampling Geometries (also known as Point Scientific Feature Types). This approach has greatly facilitated the joint use of continuous data (e.g., model forecasts, remote sensing) and in situ sensor-based observations from fixed monitoring stations, depth profilers, and autonomous vehicles. However, integration with sample-based observations and hydrological data is a continuing challenge that hinders some marine applications as well as cross-domain “summit-to-sea” integrative assessments of watershed freshwater exports with coastal impacts such as eutrophication and pollutant runoff.

3. Community context and design requirements

Many of the information requirements for ODM2 were identified through our experience in developing existing geoscience cyberinfrastructure over the past several years and through jointly reviewing the needs of existing systems. We spent considerable time examining data use cases from our own data collection activities, those ongoing within the CZOs, and those from users of the data management and publication systems we have developed. We also considered examples from the literature where different groups had used ODM 1.1.1 (e.g., Muste et al., 2010; Conner et al., 2013; McEnery et al., 2013; Muste et al., 2013). We assembled and used a broad set of data use cases to establish a scope for the types of observational data that ODM2 needed to support and to provide a means for testing the generality and effectiveness of ODM2 with real data.

We gained additional insight from those who had modified or extended ODM 1.1.1 to support additional types of data or new functionality. For example, Beran et al. (2008) adopted many of the concepts from ODM 1.1.1, but suggested a new structure with a core data model and a set of profiles that extended the core to provide functionality to support additional data types. Winslow et al.
m. Mason et al. (2014) derived a new data model called VOEIS Data Model (VODM) that added data streams, data sets, users, roles, memberships, and other features to ODM. More recently, Hersh and Maidment (2014) extended ODM to better support applications with physical, chemical, and biological oceanographic data. Each of these efforts made important modifications and improvements to the original ODM design that we were able to consider for ODM2.

We also closely examined the functionality that ODM2 could support within the SOAs of existing cyberinfrastructures to identify requirements driven by the needs of these physical implementations. For example, a metadata catalog implementation of ODM2 has different requirements than a data transfer schema, and both contributed unique requirements for the information model design. We elicited input from community members through surveys, informal communications, and through a review of considerable informal feedback on our existing systems received from community members over the past several years.

We held two community design workshops during 2012 and 2013 with 39 participants from the geoscience community. At these workshops, we examined data use cases, identified requirements for physical implementations, and vetted preliminary designs. We asked participants to articulate questions or queries that they would like geoscience domain cyberinfrastructures to support. Most of their responses were in the form of discovery queries of types not typically supported by existing geoscience cyberinfrastructures (see Table 1 for sample queries that were generated). We examined each of the queries and identified the information that would be required to support them. For example, the first query in the list (“Find stations that have soil conductivity measurements deeper than 1 m below the surface”) would require descriptive information about which properties have been measured at a location along with information about spatial offsets (e.g., depth below the surface at which the observation was measured). We then sought to support the information requirements of as many of these queries as possible in our design for ODM2. We combined the information requirements extracted from our workshop activities, those that we identified through our own experience, and those from our review of the literature into our design.

Finally, we identified a range of specific data use cases (e.g., specific observational datasets) from multiple scientific domains that we used to “flex” and test the information model. We tested multiple physical implementations of the information model (e.g., relational database implementations in multiple RDBMS, markup file based implementations for Internet exchange, etc.) by loading data from these data use cases, writing queries, testing file parsers, etc. We then used what we learned from our prototyping to feed back into the information model design.

### 4. ODM2 information model design

The OGC O&M standard (Cox, 2010) provides a definition of an observation that is useful as context for the description of the ODM2 information model:

> An observation is an act associated with a discrete time instant or period through which a number, term, or other symbol is assigned to a phenomenon. It involves application of a specified procedure, such as a sensor, instrument, algorithm, or process chain. The procedure may be applied in situ, remotely, or ex situ with respect to sampling location. The result of an observation is an estimate of the value of a property of some feature.

A general, domain-agnostic definition of observations, such as the one above, provides the basis for a high-level information model that supports observations of many types from many domains. However, in practice the specific information requirements of the domain must be mapped to the general concepts of “observation,” “act,” “phenomenon,” “procedure,” “result,” “property,” “feature,” etc. ODM2 does this for the domain of spatially discrete Earth observations. The ODM2 information model design adopts several of these terms from O&M to provide a high level structure within which we were able to organize the information requirements we identified. In fact, ODM2 can be considered an application profile of O&M for spatially discrete Earth observations. However, we also extended O&M in some significant ways to meet the needs of our data use cases and functional requirements.

The ODM2 information model is organized with a “Core” and multiple “extension” schemas. The Core contains entities, attributes, and relationships that are common for all observations, regardless of type. The extension schemas contain entities,
attributes, and relationships that enhance the functionality of the Core to meet specific use cases. We chose this organization for modularity. Every implementation of ODM2 will use the Core schema, but extension schemas need only be used for specific use cases. Users can adopt the full ODM2 model or they can use the Core and only the extensions they need, enabling control over the level of complexity in implementation. Additionally, this makes it easier to add new functionality via future extensions. This model for metadata organization is similar to that used by other metadata standards such as Dublin Core (DCMI, 2012) and Darwin Core (Darwin Core Task Group, 2015).

In the following sections we describe in more detail the Core and each of the extension schemas that we have developed to date. Although the ODM2 information model described here is independent of physical implementation considerations and could be expressed using a number of different conceptual representations, much of our design work for ODM2 was completed using entity relationship modeling (ERM) because it was a common “language” understood by all of the members of our multi-domain design team, who had varying levels of experience with information model, data model, and software design and implementation. For the purposes of this paper, we have chosen to illustrate the design of ODM2 using “entity-relationship” diagrams showing subsets of the full ODM2 model to take advantage of the ERM’s simplified notation and to reflect our expectation that relational databases will be one of the most common physical implementations of ODM2. The full data dictionary, including definitions of each entity and attribute as well as data type descriptions, can be accessed via HTML-based documentation on the ODM2 GitHub website (https://github.com/ODM2/ODM2).

4.1. ODM2 core schema

The ODM2 Core schema is shown in Fig. 1. The name of each entity in the ODM2Core schema is prefixed with “ODM2Core,” and each is drawn with a red title box and outline. Only those entities shown in Fig. 1 are part of the ODM2Core schema. In subsequent sections (4.2–4.8) we describe extensions to the ODM2Core schema. Within these sections we use similar symbology (name prefixes and unique colors surrounding entity titles) to indicate the extension to which each entity belongs. In some diagrams, we have included entities from the ODM2Core schema to illustrate where the extension entities connect to the ODM2Core entities.

In the ODM2 Core, an “observation” is made up of two elements: an Action performed on or at a SamplingFeature that produces an observation Result, and a Result that is the outcome of that Action. Partly adapted from a laboratory data model (e.g., Wendel et al., 2007), this is a critical distinction as ODM2 explicitly models Actions for which a formal relationship has been created (i.e., “isRelatedTo” or “isRelatedTo”). An Action and its Result may have other Actions and their Results as relatives or parents (e.g., a “Specimen analysis” Action with a “Measurement” Result or an “Instrument deployment” Action with a “Time series” Result). It also enables the creation of Actions that do not have Results but that may be related to Actions that do produce Results (e.g., a “Specimen preparation” Action that is related to a “Specimen analysis” Action that produces a Result), which is important in representing sampling workflows (see details below). An Action that produces a Result is performed by one or more People on or at a SamplingFeature (see Section 4.2 for a description of SamplingFeatures). People may be affiliated with an Organization when they perform an Action. Every Action is performed using a Method, which is the equivalent of O&M’s “Procedure.” Like Actions, Methods can be of many different types (e.g., “Instrument deployment,” “Specimen collection,” “Specimen analysis,” etc.), and terms used to specify Method types are identical to those used to specify Action types (e.g., a “Specimen analysis” Action is described by a “Specimen analysis” Method).

Actions are modeled generically so that many different types of Actions can be recorded and associated either directly or indirectly with a Result. Actions that produce Results may have other Actions associated with them. For example, a “Specimen analysis” Action that generates a Result may have a related “Specimen collection” Action and a related “Specimen preparation” Action. Relationships among Actions (e.g., in the case of a workflow of many Actions that culminates in an observation Result) can be expressed using specific semantic terms chosen from a controlled vocabulary (e.g., “isChildOf” or “isRelatedTo”) that specify parent/child or other relationships among Actions. Using relationships among Actions, complex workflows of sensor deployments or specimen collection and handling that culminate in observation Results can be expressed in ODM2.

Fig. 2 provides an example of an hierarchy of Actions that all originated from a single visit to a monitoring Site. There are multiple linear workflows within this example, where a workflow is a path from the first Action (the site visit) to an independent Result (i.e., a measurement of the concentration of some constituent in one of the Specimens). There is also a partial workflow where a Specimen was collected, but then no further Actions were taken. In this example, even though each of the Specimens may have been collected at the same location and time, each of the Results may be difficult because the Specimen preservation and preparation Actions are different. However, each of the Results can be traced all the way back to the site visit during which the Specimens were collected by tracing up the hierarchy using each Action’s immediate parent, facilitating complete metadata to support interpretation of each of the Results.

In contrast to Specimen workflows, deployment of in situ sensors involves ongoing data collection, and additional site visit Actions may be required over time for sensor cleaning, maintenance, and calibration. Representing the sequence of Actions related to a sensor deployment and the generated time series Results involves potentially multiple, separate hierarchies of Actions, all of which must be related to the initial sensor deployment (because the sensor deployment is the Action to which the Result is linked). For example, Fig. 3 shows that not all relationships among Actions are simple parent/child relationships that result in linear workflows. Instead, multiple types of relationships are required, with relationships among Actions expressed using semantic terms chosen from a controlled vocabulary. A user wishing to retrieve all Actions related to a time series Result would get not only the relationship with the site visit during which the original deployment was made, but also all subsequent Actions that have been related (e.g., instrument cleaning, calibration, and retrieval Actions or any other Actions for which a formal relationship has been created).

A Result consists of metadata describing one or more data values that are the consequence of an observation Action. Every Result must be associated with a single Action that generated it. Results can be of many different types, which are described in a controlled vocabulary of Result types. The data values themselves are not encoded in the ODM2 Core schema, but rather in a set of linked entities in the Results extension (see Section 4.3 for details). Separating metadata about Results from their data values enables catalog implementations of ODM2 that would use metadata for Results in the Core schema to enable data discovery, but would not require the data values themselves. In a relational database context,
an additional benefit for Result types involving many data values per Result (e.g., TimeSeries) is that inserts and reads of data values will execute much more quickly given that the data values are no longer in the center of the schema and joined to all of the metadata tables as was the case with ODM 1.1.1.

Results are associated with a Variable that specifies the observed property, Units specify the units of measure for the data values within a Result, and a ProcessingLevel defines the level of processing to which a result has been subject (e.g., "Raw" or "Quality Controlled" data). Each Variable has a name and a type that are chosen from controlled vocabularies in ODM2. Unlike ODM 1.1.1, Units are not treated as a controlled vocabulary in ODM2. Rather, ODM2 contains an entity within which Units of measure can be described with a link to an external system from which the units were chosen (e.g., the ODM 1.1.1 Units controlled vocabulary or the Quantities, Units, Dimensions and Data Types Ontologies (QUDT) (Hodgson et al., 2014)).

TaxonomicClassifiers provide a way to classify Results according to terms from a formal taxonomy. This entity permits an additional dimension for Results, enabling users to assign both a name for the Variable that was observed and a taxonomic term that qualifies the Variable name. Examples include specification of the taxonomic name (TaxonomicClassifier) of a species for which a presence/absence or count observation has been made (Variable), specification of the taxonomic name of a mineral (TaxonomicClassifier) for which a percent composition observation has been made on a specimen (Variable), or specification of the taxonomic name (TaxonomicClassifier) for a soil type on which an observation has been made (Variable). Use of TaxonomicClassifiers avoids the repetition and complexity of overloading Variable names with taxonomic terms. For example, if an investigator was observing the number of fish for multiple species within a given stretch of river, a VariableName would be required for each fish species that might be observed (e.g., "Count of Oncorhynchus mykiss," "Count of Salmo trutta," "Count of Oncorhynchus clarkii," "Count of Salvelinus fontinalis," etc.). ODM2 enables specification of a single Variable with a name of "Count." The entity being counted can then be specified using terms from a taxonomy. In this example, terms from the taxonomy specify the count "of what." Similarly, if a user were observing percent composition of many different minerals in a rock sample, a single Variable name of "Percent composition" could be used along with a taxonomy that formally defines the names of the minerals.

The Datasets entity is used to encode information about groups of Results that are logically related. Datasets may consist of a single Result (e.g., a time series Result for a single Variable collected at a monitoring Site) or could be a group of many Results (e.g., all of the time series Results for all Variables collected at a monitoring site). ODM2 does not constrain how Results are grouped into Datasets. Groupings are determined by users. Each Dataset may be assigned...
one or more Citations using the Provenance extension. If the Dataset was loaded into an ODM2 database instance from an external source (e.g., a data table from a published manuscript), the Citation would point at the original source of the data. If the ODM2
database instance in which the data are stored is the original source of the data (e.g., where the ODM2 database is used by a research group or system to publish original data), the Citation information specifies how that Dataset should be cited by others. See Section 4.5 for more details about Citations and provenance.

4.2. Sampling features extension

In many cases, scientists wish to make observations on spatial features that can be inaccessible or whose properties are not directly or readily observable given a feature’s size or location. For example, it is impossible using current methods to measure many properties of an entire aquifer or over an entire rock outcrop. The O&M standard defines these real-world, geospatial features as “features of interest,” or the feature of ultimate interest to an investigation (Cox, 2010). A specific set of observations may also be collected to address more than one feature of interest at different scales (e.g., a stream reach and the entire drainage area), or may vary with time. To overcome these ambiguities and physical challenges regarding features of interest, proximate “sampling features” are used for observation. Sampling features are accessible and have properties that are sensible. ODM2 adopts the term “SamplingFeature” from O&M to describe the geospatial or physical entity on which or at which observations are made and other actions are performed. SamplingFeatures can be of many different types, chosen from a controlled vocabulary, including Sites (as defined in ODM 1.1.1), Specimens, Transects, Sections, Profiles, Bore holes, etc. The SamplingFeatures extension enables the encoding of detailed information about SamplingFeatures. While optional, it is anticipated that this extension will be valuable for most ODM2 use cases. ODM2 currently focuses on SamplingFeatures that are Sites or Specimens, which are relevant to a large percentage of the data use cases we examined; however, several other types of SamplingFeatures can be represented.

Every SamplingFeature, regardless of type, is described first in the SamplingFeatures entity in the Core schema. Then, specific attributes are encoded in an entity customized for the SamplingFeature type (e.g., Sites or Specimens) and organized within the SamplingFeatures extension (Fig. 4). Currently, only Sites and Specimens entities have been created in ODM2. For other SamplingFeature types, users may wish to add custom entities to the SamplingFeatures extension if additional detail is needed. If relevant, each SamplingFeature may have a spatial geometry encoding and a corresponding geospatial primitive type (e.g., point, line, polygon, etc.) chosen from a controlled vocabulary. The method for expressing the geometry depends on the physical implementation of the information model. For example, it may be expressed using a native geospatial data type within a relational database management system or it may be encoded in Well Known Text (WKT) (Herring, 2011) or Geography Markup Language (GML) (ISO, 19136, 2007) for text-based or XML encoding, respectively.

SamplingFeatures may also be related. A Specimen SamplingFeature may be collected at a Site SamplingFeature or may be subsampled from another Specimen. A Profile SamplingFeature may be located at a Site SamplingFeature. In all cases, these relationships are encoded in the RelatedFeatures entity. Each SamplingFeature may be associated with a SamplingFeature that is its parent (e.g., a Specimen SamplingFeature is collected at a parent Site SamplingFeature), or may participate in other specific relationship types chosen from a controlled vocabulary. Finally, spatial offsets for SamplingFeatures provide the ability to describe relationships among SamplingFeatures that involve some sort of geographical separation between the two features. An example would be a Specimen that was collected at a Site, but at a specific distance below the surface of the water or land. Spatial offsets are specified in the SpatialOffsets entity and are expressed using up to three coordinates, each of which has a value and units. The SpatialOffset is also described by a SpatialOffsetType, which is chosen from a controlled vocabulary. The SpatialOffsetType determines how each of the three coordinates is used. For example, a simple depth offset can be expressed using a single coordinate that records the depth below the land or water surface. In this example, the first offset coordinate would be used to record the depth. The other two offset coordinates would be NULL.

4.3. Results extension

ODM2 currently supports several Result types that allow for encoding data of different types (i.e., numeric versus categorical) and data that vary over space, time, and potentially other dimensions. In describing the individual Result types supported by ODM2, it is useful to describe the concept of a “measurement
framework.” A measurement framework is a conceptual scheme that establishes rules for control of components of a phenomenon to permit measurement of one component (Sinton, 1978; Chrisman, 1999). In ODM2, each Result type has a spatial component, a temporal component, and an observed Variable component. According to Chrisman (1999), there are generally three possible roles for the spatial, temporal, and observed Variable components—in order to measure one component, one of the others has to be fixed, while the third serves as a control. Control denotes a mechanism of restraint on the variation of a component. For example, a water level sensor must be fixed in one location, and the rate of temporal sampling controlled so the depth of water (the observed Variable) can be measured. Explicit measurement framework conceptualization (e.g., the role of space, time, and observed Variable) is useful in understanding how data can be integrated and is captured in the metadata expressed in ODM2 for each Result type. Nevertheless, it is possible for all three components (the triplet of space, time, and observed Variable) to be varying (i.e., measured) jointly.

ODM2 does not constrain the measurement framework for Result types (i.e., it does not require every Result type to have a fixed, controlled, and measured component). However, a fundamental, structural choice has been made that for all Result Types the observed Variable component is measured. In ODM2, ProcessingLevel, Units, data collection Status, and SampledMedium, which are all attributes of a Result, are considered part of the observed Variable component that is measured in the measurement framework for each Result type. Given that the Variable component is always measured, it is the spatial and temporal components that distinguish between the different Result types.

The spatial component may be fixed, controlled, or measured. In a fixed space example, a Result may be measured on a single, fixed SamplingFeature (e.g., on a Specimen or at an individual Site). Similarly, a Result may be measured on a single, fixed SamplingFeature, but at a location that is offset from the SamplingFeature (e.g., a sensor that is installed at a Site, but located some distance below the soil surface or above the ground). Where space is controlled, a Result may be measured with values at controlled X, Y locations within a SamplingFeature (e.g., values representing a Variable measured at equally spaced locations within a SamplingFeature that is a line or polygon). Measurements along a sampling feature that is a trajectory (e.g., a glider where X, Y, and potentially Z may vary) is an example of a Result type where space is measured.

Similar to the spatial component, the temporal component may be fixed, controlled, or measured in ODM2. For a sensor installed at a Site, individual data values are regularly recorded with a time spacing that is controlled by the datalogger. However, in most cases a Specimen is collected at single instant in time, effectively fixing the temporal component for any Results created from that Specimen. In some special cases, the temporal component may be measured (e.g., the time at which a particular value of a variable occurred). In this case, the temporal component and the observed Variable component correspond.

The data values and Result-specific attributes for each Result type are represented in two entities per Result type. The first entity stores the attributes of the Result that are fixed (i.e., attributes that are the same for every data value within the Result). The second entity stores the actual data values and attributes of the Result that vary or are measured with each individual data value. Fig. 3 shows this structure for a subset of Result types, including measurement, time series coverage, section coverage, and transect coverage. In addition to the Result types shown in Fig. 5, we have modeled categorical, depth profile, point coverage, trajectory coverage, and spectra coverage Result types. Additional Result types beyond these can be defined by specifying a measurement framework and designing a similar, two-entity construct that can be added to the Results schema without affecting the structure of the Core or other schemas.

Regardless of the Result type, each individual data value within a Result may have a spatial offset associated with it. Spatial offsets for individual data values are used where an offset from a SamplingFeature location is needed, but where that offset does not constitute creation of a new SamplingFeature. Handling of spatial offsets for data values is specific to the Result type. Examples include time series Results at a weather station Site where sensors are located a fixed distance above the ground (each data value has the same spatial offset) or a water quality depth profile Result where an instrument is lowered through a water column from a boat anchored at a fixed Site location (each data value has a different spatial offset). Table 2 lists the measurement framework for each of the Result types shown in Fig. 5 and describes how the spatial offsets are handled. Fig. 6 provides an illustration of each.

Depending on the Result type, ODM2 allows data values that have been aggregated in both space and time. For example, a measurement Result may be based on a composite Specimen that was collected over a period of time. Each recorded value in a time series Result may represent an average over a specific time support interval. A recorded value in a depth profile Result may represent an average (or some other statistic) over a specific depth interval rather than an instantaneous value at a single depth. Spatial aggregation is accomplished by specifying the interval over which the aggregation occurred. For aggregation in time, a temporal aggregation interval can be specified. A single aggregation statistic (e.g., average, minimum, maximum) can be chosen from a controlled vocabulary that defines the type of aggregation performed on data values over the combined space/time support of the recorded value.

4.4. Equipment and LabAnalyses extensions

The optional Equipment extension (Fig. 7) plays a vital role in linking observations to the equipment that was used to create them. Which sensor was used to measure temperature at a Site? Which datalogger was used to record the data? What was the model number of the mass spectrometer that made that measurement, and whose lab was it in? These details are usually kept in field or laboratory notebooks and, in most cases, separate from the observation values themselves. This makes it difficult to link an observation to the particular pieces of equipment used in the measurement. Yet, it is important to preserve these metadata for later use in post processing data for quality control and, in some cases, in interpreting the resulting data values.

The Equipment extension, along with the DataQuality, and/or LabAnalyses extensions (depending on the type of data being collected), makes it possible to explicitly link observed data values with the pieces of equipment used to record them. Direct linkage can be made between Actions and the equipment used to perform them. Individual pieces of equipment can be described, including type, serial number, and purchase information. Owner and vendor information are linked to Organizations and People. The Equipment entity is linked to the EquipmentModels entity, which is used to describe models of Equipment for which there may be multiple instances (e.g., multiple dataloggers of the same model). For instruments capable of generating Results, information about Variables that can be measured or output and the Methods used by the instrument can be recorded. Attributes of instruments and the variables they measure include accuracy, resolution, and raw output units.

Instances of Equipment can be related to each other via the RelatedEquipment entity (e.g., instruments that are part of an in situ sensor suite). Because these relationships are not static, the entity
Table 2: Description of and measurement framework for measurement, time series coverage, section coverage, and transect coverage Result types.

<table>
<thead>
<tr>
<th>Result type</th>
<th>Space</th>
<th>Time</th>
<th>Variable</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Measurement</strong></td>
<td>Fixed: Usually described by a SamplingFeature that is a Site (fixed X, Y, and Z coordinates). The location at which a Specimen was collected may be offset from the Site location (e.g., below the water surface in a lake).</td>
<td>Fixed: Fixed in time, with the timestamp being associated with the instant at which the Specimen was collected, or, where the measurement was made in the field and does not involve a Specimen, the instant at which the Measurement was made.</td>
<td>Measured: The value is a floating point number and represents an observation of a Variable. ProcessingLevel, Units, Status, and SampledMedium describe the Result value.</td>
</tr>
<tr>
<td><strong>Time Series Coverage</strong></td>
<td>Fixed: Fixed in time, with the timestamp being associated with the instant at which the Specimen was collected, or, where the measurement was made in the field and does not involve a Specimen, the instant at which the Measurement was made.</td>
<td>Controlled: Spacing is recorded via the timestamp for each Result value. Spacing and time support are controlled by the sensor or logger.</td>
<td>Measured: Values are floating point numbers and represent observations of a Variable. ProcessingLevel, Units, Status, and SampledMedium are the same for every Result value in a Time Series Coverage Result.</td>
</tr>
<tr>
<td><strong>Section Coverage</strong></td>
<td>Fixed or Controlled: Transect Coverage Results may represent an instant in time (e.g., all values along the transect have the same timestamp) or each value within the Transect may have its own timestamp (e.g., where a sensor is moved sequentially to different locations along the transect).</td>
<td>Fixed or Controlled: May represent an instant in time (e.g., all values have the same timestamp) or, each value within the Result may have its own timestamp (e.g., a sensor is moved sequentially to different locations within the section).</td>
<td>Measured: Values are floating point numbers and represent measurements of a single Variable at each X, Z location within the section. ProcessingLevel, Units, Status, and SampledMedium are the same for every value in the Result.</td>
</tr>
<tr>
<td><strong>Transect Coverage</strong></td>
<td>Controlled: Covers a SamplingFeature that is a 2-dimensional transect line. Z is fixed for each value, but X and Y may vary. The transect line is defined by connecting the X, Y coordinate pairs. Spacing along the transect may be variable or consistent, in which case an intended transect spacing can be specified.</td>
<td>Controlled: Covers a SamplingFeature that is a Site (fixed X, Y, and Z coordinates). Location of the measurement device may be offset from the Site location (e.g., installed below the soil surface or above the ground).</td>
<td>Measured: Values are floating point numbers and represent measurements of a single Variable at each X, Y location along the transect. ProcessingLevel, Units, Status, and SampledMedium are the same for every value in the Result.</td>
</tr>
</tbody>
</table>
includes attributes to describe the date range of the relationship. Actions and Equipment are linked via a bridge entity indicating a many-to-many relationship between Actions and Equipment because an Action may be performed using multiple pieces of Equipment, and multiple Actions may be performed on or with a single piece of Equipment.

The Equipment extension also describes equipment-related Action types that do not generate Results but are directly related to Actions that do (e.g., an “Instrument calibration” action can be related to an “Instrument deployment” Action and a “Specimen collection” Action can be associated with a “Specimen analysis” Action, similar to the cases illustrated by Figs. 2 and 3). Some equipment-related Actions require metadata in addition to those used to describe generic Actions. In particular, the CalibrationActions entity permits multiple calibration Methods, including using a reference material or a reference sensor as the calibration standard. For calibrations using reference equipment, the Action is linked to the Equipment entity, and similarly, for calibrations using a reference material (e.g., calibration solutions), the Action is linked to entities describing reference materials from the DataQuality extension. The reference material could also be a specimen, in which case it can be linked to the SamplingFeatures entity.

Sensor suites and data collection activities at monitoring sites change over time, and it is often important to track datalogger programs used in field deployments. The DataloggerProgramFiles entity records information about datalogger programs. Each datalogger program can generate multiple output data files, which are described by the DataloggerFiles entity. A datalogger file typically consists of multiple columns, one for each Variable. In most cases, a sensor deployment that uses a datalogger and has an associated datalogger file would generate a Result of type “Time series coverage.” Time series coverage Results can be roughly mapped to a column in a datalogger file; however, depending on how datalogger files are managed, a single Time series coverage Result may span many datalogger files (e.g., in the case where a new datalogger file is created each time data are downloaded from the datalogger). The DataloggerFileColumns entity is related to Results, and relates datalogger files and programs to the Results they generate.

The LabAnalyses extension can be implemented for use cases involving ex situ laboratory analyses of Specimens and adds a small number of entities to the Equipment, SamplingFeatures, and ReferenceMaterials extensions. These entities enable association of Specimen collection or analysis Actions to Directives, which can provide rationale for a data collection effort or project and the resulting analyses. A SpecimenBatchPosition entity allows recording of the order for which each Specimen in a batch was run as part of an Action of type “Specimen analysis.” Finally, metadata to associate specimens with standards are stored using the ReferenceMaterials entity.

4.5. Provenance and external identifiers extensions

The optional Provenance extension (Fig. 8) is used to encode Citations for Annotations, Datasets, and Methods, the author list for Citations, and relationships among Annotations, Citations, Datasets, and Results. The entities in the Provenance extension are primarily...
used to record Citations that specify the original source of information (e.g., where a Method or Dataset was added to an ODM2 instance from an existing source). However, where the ODM2 instance is the original source of a Dataset, a Citation can be created that specifies how the Dataset should be cited. A list of People and their respective order of authorship can be specified for each Citation.

The remaining entities in the Provenance extension were created to encode relationships among Datasets, Results, Citations, and Annotations. Specific relationship types are selected from a controlled vocabulary. For example, a relationship can be expressed between two Datasets indicating that one is a newer version of another using a relationship type term of "isNewVersionOf." Semantic terms describing relationship types expressed by DataCite (DataCite, 2013) were adopted for use within ODM2. Relationships among Results are a special case given that a Result may be derived from one or more other Results (e.g., a time series of river stage might be converted to a time series of discharge). Where desirable, the relationship between a derived Result and the Result(s) from which it was derived can be specified using the RelatedResults entity. A relationship type term of "isDerivedFrom" would be selected from the controlled vocabulary. An optional VersionCode can be specified if a specific version number or code is to be applied. In cases were a Result is derived from multiple other Results, multiple entries would be entered in the RelatedResults entity. Each could be assigned a RelatedResultSequenceNumber that identifies the order in which the Result was used in the derivation. Where it is required to store an equation by which the conversion was performed, it can be encoded in the DerivationEquations entity. The DerivationEquation entity can encode the equation in such a way that it records the sequence in which the related Results were used so the derivation is repeatable.

As another measure for recording provenance information, ODM2 provides an optional External Identifiers schema for establishing linkages between entities within an ODM2 instance and representations of those entities in other systems. For example, investigators from a CZO may register Specimens in the SESAR system and then want to encode the IGSN for each Specimen in an ODM2 database instance. The entities in the External Identifiers schema required to make this linkage are shown in Fig. 9. Similar linkages to external systems can be made for People (e.g., Open Researcher and Contributor ID – ORCID, http://orcid.org), Reference Materials, Methods (e.g., National Environmental Methods Index ID – http://www.nemi.gov), Variables, TaxonomicClassifiers, and Citations (e.g., Digital Object Identifier – DOI).

4.6. Annotations and data quality extensions

One limitation of ODM 1.1.1 was that each data value could only be annotated with a single data qualifying comment, and no information could be stored about who created a qualifier or when it was applied. Additionally, annotation of the other entities was impossible without modifying the schema. The optional ODM2 Annotations extension addresses this shortcoming and enables users to create qualifying comments or notes about entities within an ODM2 instance. The Annotations extension (Fig. 10) was designed such that one or more annotations can be added to instances of the following entities: SamplingFeatures, Actions,
Methods, Results, ResultValues (the ResultValue entities for any of the ResultTypes), and Equipment. This met the needs of our data use cases. With minor modification, the annotations entity could be bridged with any of the other ODM2 entities.

Each text Annotation is classified by an AnnotationType selected from a controlled vocabulary (e.g., “Action annotation” or “Figures”).
"SamplingFeature annotation," etc.) and an optional AnnotationCode (e.g., “a” as a code for a text annotation of “Approved”). Each text Annotation is also optionally described by a date on which it was created and is linked to the Person that created it. An annotation may also be linked to a Citation via the ODM2 Provenance extension where it is desirable to store information about the source of a particular Annotation. Annotations are added to individual instances of the entities listed above. For example, an Annotation can be added to a single SamplingFeature without requiring a similar value for all other SamplingFeatures. This allows users to make very granular text Annotations. The design also allows for multiple Annotations for a single entity instance — e.g., in the case that a user needs to add many Annotations to a single Result or data value.

The optional DataQuality extension (Fig. 11) enables users to encode specific information about the quality of their Results. Values for the accuracy, precision, method detection limit, reporting level, etc. for the data values within a Result can be encoded using the DataQuality entity. Additionally, the DataQuality extension provides the ability to encode information about reference materials. Reference materials, sometimes called calibration standards, are materials or substances whose properties are sufficiently homogenous and well established to be used for the calibration of an apparatus or instrument, the assessment of a measurement method, or for assigning values to unknown specimens. Most analytical instrumentation requires the analysis of a reference material for accurate instrument calibration and/or for normalization of the resulting data values to established scales and units. Tracking information about the specific reference material used for a measurement is critical for interpreting many Results. The ReferenceMaterials entity provides a means to track materials used to develop calibrations and normalizations, and the ReferenceMaterialValues entity records the specific established values used for the calibration or normalization.

4.7. ExtensionProperties extension

As an optional extensibility mechanism for ODM2, we included an ExtensionProperties extension. This extension enables users to add new attributes to entities within ODM2 without modifying the structure of the base information model. Fig. 12 shows an example of the general structure of the ExtensionProperties extension. Extension properties are created by specifying a name and description, data type, and Units (where appropriate) for a property, and then assigning a value for that property to related instances of an entity (e.g., one or more particular SamplingFeatures or Actions). This structure enables the addition of extension properties to individual instances of each entity. For example, an ODM2 user may want to add a specific property called “ContainerVolume” with an integer data type and Units of milliliters to all water quality Specimen SamplingFeatures in an ODM2 database. That same property would be inappropriate for SamplingFeatures of other...

Fig. 10. Entity-relationship diagram showing a subset of the Annotations schema for SamplingFeatures and Actions.

Fig. 11. Diagram showing the general structure of the DataQuality extension.

Fig. 12. Diagram showing the general structure of the ExtensionProperties extension.
Fig. 11. Entity-relationship diagram showing entities in the DataQuality extension.

Fig. 12. Entity-relationship diagram showing a subset of the ExtensionProperties schema for SamplingFeatures and Actions.
types stored in the database (e.g., the Sites at which the Specimens were collected). We have currently modeled extension properties for SamplingFeatures, Actions, Methods, Results, Variables, and Citations as these are the entities that are most likely to be extended. However, a similar construct could be used to create extension properties for any entity in ODM2.

4.8. Controlled vocabularies extension

Like ODM 1.1.1, ODM2 was designed with controlled vocabularies (CVs) that define the terms that can be used to populate many of the metadata attributes within the entities of the information model in efforts to reduce the semantic heterogeneity across multiple instances of ODM2. Names of attributes within ODM2 for which CVs have been developed end with “CV” (e.g., ActionTypeCV, ResultTypeCV, VariableTypeCV, etc.), and entities for encoding CV terms have been added to an optional CV extension. Several of the ODM2 CVs are “Type” CVs that specify the specialized type of a particular entity instance and participate in the business logic of the information model. For example, a Result’s type is specified in the ResultTypeCV attribute, and the value of the ResultTypeCV attribute would indicate which linked entities contain the full metadata description and data values associated with the result (e.g., a Result with ResultTypeCV of “Time series coverage” would have related metadata and data values in the TimeSeriesResults and TimeSeriesResultValues entities).

We developed an initial version for all of the ODM2 CVs and exposed them in a community CV registry and moderation website (http://vocabulary.odm2.org). This website is similar to the shared vocabulary moderation system we built for ODM 1.1.1 (Horsburgh et al., 2014). In fact, several of the ODM2 CVs were inherited directly from those used in ODM 1.1. Others were developed for new attributes that have been added to ODM2. Many of these describe Specimens and their attributes and adopted terms from PetDB. The ODM2 CV registry provides the capability to view the existing CVs and terms, request edits to existing terms, and request new terms. We also enhanced this website with a Representational State Transfer (REST)ful web service application programming interface (API) to expose each of the vocabularies and individual terms using Simple Knowledge Organization System (SKOS) encoding (Isaac and Summers, 2009) so they could be automatically/programmatically retrieved in a format that is compatible with other vocabulary systems. Although we have suggested terms for all of the CVs within ODM2 via the ODM2 CV registry website and encourage users to adopt and contribute to the community moderation of these terms, ODM2 does also allow for using terms from any formally published vocabularies.

5. Physical implementations for ODM2

Physical implementations of the ODM2 information model serve a particular purpose or function, in this case major roles and functionalities within SOAs for managing and publishing observational data. At the outset of our work, four primary physical implementations of ODM2 were identified, including data storage, data exchange, metadata cataloging, and data archival. It should be noted that because physical implementations serve different purposes (e.g., storage on disk versus data exchange over the Internet), they may use different syntax, file formats, software technologies, and data types. Our choices for the physical implementations described in the following paragraphs were driven by the needs of our existing cyberinfrastructure systems (e.g., RDBMSs are commonly used in geoscience cyberinfrastructure and the systems we have developed). However, our needs may be different than what others may choose for physical implementations using different technologies. For example, some may wish to implement concepts from the ODM2 information model within object-relational or object oriented databases, NoSQL databases, JavaScript Object Notation (JSON) for data exchange, etc. In each case described below, a mapping was performed to ensure that the information model content could be expressed using the file formats, data types, and organizational structures available within the chosen software technology. Similar mappings would have to be performed for any specific physical implementations of ODM2.

To support data storage, we developed relational database implementations of ODM2. Structured query language (SQL) scripts are available for generating blank ODM2 databases within Microsoft SQL Server, MySQL, PostgreSQL, and SQLite. Users can choose the appropriate script for their platform to implement an ODM2 database for use as an operational data store. Although not yet available, modified relational database implementations that take advantage of data warehousing techniques could be implemented to facilitate efficient querying and analysis of data stored in an ODM2 database. We also have developed a Python-based object model representation of ODM2 with an associated Python API that we are using as the basis of new software tools for loading data into and managing data within ODM2 databases. The Python object model and API can be used on Windows, Mac, and Linux platforms and with the four relational database management systems listed above. These tools are available via the ODM2 Python API source code repository (https://github.com/ODM2/ODM2PythonAPI).

Our work to date on a data exchange format for ODM2 has two facets. First, we have defined a YAML (YAML Ain’t Markup Language)-based text file format called YODA (YAML Observations Data Archive) for encoding ODM2 datasets for exchange over the Internet (https://github.com/ODM2/YODA-File). Similar to CZO Display Files used to encode time series of observational data for ingestion into ODM 1.1.1 databases, the primary objective for the YODA format is to encode more diverse observational datasets (i.e., time series of sensor data and/or geochemical data derived from physical samples, whereas CZO Display files are limited to time series) in a human and machine readable format for transfer over the Internet. One of the driving use cases for YODA files is to enable data managers from individual CZO sites to transfer observations datasets encoded in a YODA file to the CZOData system where they can be loaded into an ODM2 database and cataloged for discovery. Standardization on the YODA file format helps to overcome the heterogeneity in data management systems used by the different CZOs and supports integration of a broader set of observational data types into the existing CZOData system. At the time of this writing, we have developed Microsoft Excel-based data entry templates into which data managers can paste data and then export a valid YODA file. YODA files contain fully specified ODM2 datasets, and we are now investigating YODA as a potential long-term data archival format for submitted datasets. Work is ongoing within the CZOData project to integrate YODA files into the existing CZOData system.

In addition to the YODA format, we are experimenting with XML and JSON encodings of ODM2 datasets for use with web services that allow retrieval of datasets from an ODM2 database. This includes mapping ODM2 Results to existing XML schemas (e.g., mapping ODM2 time series coverage Results to WaterML 1.1, see https://github.com/ODM2/WOFpyp for implementation and services) and extending beyond existing XML schemas to support additional ODM2 result types (see https://github.com/ODM2/ODM2RESTfullWebServices for implementation and services). When complete, these efforts will enable multiple opportunities for ODM2 users. First, users will be able to host their own ODM2 databases and compatible web services that can be integrated with cyberinfrastructure systems like those hosted at the CUAHSI Water
Data Center (which currently supports web services using WaterML). Second, the ODM2 web services under development have an extensibility mechanism that would enable adaptation of the service to an existing data store for cases where users may want to become interoperable at the service interface and exchange format level while keeping their existing data store.

Although we have not yet implemented a formal metadata catalog based on ODM2, we structured ODM2 with data values outside of the core schema to better support an eventual metadata catalog implementation. We also included globally unique identifiers (GUIDs) in ODM2 for SamplingFeatures, Results, and Datasets to aid in disambiguating these entities across metadata compiled from multiple ODM2 instances. A primary application area for this functionality will be in future iterations of the CZOData system, where CZO data managers are routinely working with both sensor and sample data and where we are working to develop a catalog that provides access to both.

Finally, we have adopted ODM2 as the storage model for hydrologic time series datasets in HydroShare (http://www.hydroshare.org) (Tarboton et al., 2014; Horsburgh et al., 2015). In HydroShare, users can upload time series datasets and share them with other HydroShare users, providing scientists with a convenient way to store and share data without setting up any data management infrastructure, while at the same time taking advantage of the rich metadata capabilities of ODM2. New functionality under development in HydroShare includes the ability to upload a YODA file that is then parsed into an ODM2 SQLite database to support value-added functionality such as visualization. We also plan to extend the use of ODM2 for storing Specimen-based datasets in HydroShare.

6. Discussion and concluding remarks

Our ultimate goal was to develop an information model that is integrative and extensible, that accommodates a wide range of observational data, and that serves both research scientists and software developers by providing the information model, initial physical implementations, and software tools that can be used to promote greater interoperability across multiple disciplines and systems that support publication of Earth observations. Identifying the information requirements was a necessary first step that was greatly facilitated by the community of people who participated. We anticipate that the designs and approaches presented in this paper will continue to stimulate further discussion and advances within the geosciences community with respect to data storage, exchange, metadata cataloging, and archival in support of geoscience research. Indeed, we anticipate that some adopters will implement ODM2 “as-is” using the software tools we have developed. Others will examine what we have developed and modify it to suit their needs within their own system development as was observed with earlier versions of ODM. We expect that continuing work within our group and the work of others will explore the performance of the various ODM2 physical implementations we have developed. These tests may provide feedback into the system development lifecycle that will enable us to continue advancing ODM2 and related software tools. The open source code repository for ODM2 (http://www.github.com/ODM2/ODM2) and the active software development community we have established provide mechanisms for new functionality, performance enhancements, and new types of physical implementations to be integrated into subsequent versions of ODM2.

ODM2 significantly improves the ability of scientists to capture metadata describing their observations. The representation of Actions and relationships between Actions, the People that perform them, the Methods and Equipment that are used, and the Results that are generated is a powerful new way to describe sampling, sensing, and analysis workflows that up to this point have been captured only in field and/or laboratory notebooks that are not accessible to potential data users. Separation of the metadata description of Results from the actual data values enabled us to flexibly model Results of many different types, provided a way to support additional Result types in the future, and enabled a catalog implementation of ODM2 that contains metadata about Results, but no data values. Representation of Citations and specific semantic relationships among important entities within ODM2 is another important advancement that aids with provenance tracking and properly crediting original authors/creators. When considered as a whole, the ODM2 core schema and the extensions described here contain a fair degree of complexity. However, ODM2 implementations do not have to be “complete” in a sense that they implement all developed extensions. An implementation that meets any specific application needs (within the scope of ODM2) can be constructed using the core schema and one or more of the described extensions or other extensions built using the same pattern.

ODM2 enhances the domain-specific information models and encodings used by the cyberinfrastructure systems we have developed over the past several years and standardizes descriptions of common characteristics of spatially discrete Earth observations. It is already being used in production implementations within the IEDA and EarthChem systems. It is now being used to represent time series and will eventually be used to represent Specimen-based datasets within the CUAHSI HydroShare system. We are also now working to integrate it with the CZOData cyber-infrastructure and software systems supported by the CUAHSI Water Data Center. The use of ODM2 by these systems standardizes the representation of many different aspects of the metadata describing observations. Indeed, ODM2 is key to realizing opportunities for more reliable data and information integration across these and other domain cyberinfrastructures. The emerging software ecosystem to support the physical implementations of ODM2, including support for implementation across multiple desktop and server platforms, will make it usable both within large-scale cyberinfrastructure development efforts as well as in more localized use cases for individual researchers or research groups. Software tools for implementing and working with ODM2 are being developed in several open source code repositories within the ODM2 GitHub organization (https://github.com/ODM2), and contributions from the community are welcomed.

Although we have made progress with ODM2 in supporting a suite of new functionality, we are still working to extend it. The extensibility mechanisms we have included and the modular core plus extensions design make this possible. Future extension schemas for ODM2 that we are currently working on include one for storing the metadata, Results, and intermediate data products of model simulations and one for capturing metadata that describe observations made under experimental or treatment conditions (e.g., tracer studies where ambient conditions within an aquatic environment are altered by release of an artificial tracer chemical). These use cases have grown out of related research projects for which ODM2 already meets a high percentage of the project’s information needs, but for which specific extensions are needed to fully support the additional metadata.

Acknowledgments

The information model reported here is the work of many people. Community design workshops were held in 2012 and 2013, at which members of the geoscience community identified requirements related to both data use cases and physical
implementations. The authors gratefully acknowledge the contributions from participants at these workshops. This work was supported by the National Science Foundation under grants EAR 1242638, EAR 1332257, and ACI 1319384. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References


Dataset URI: DataCite Metadata Schema for the Publication and Citation of Research Data, Version 3.0.http://dx.doi.org/10.5438/008.


Sinton, D.F., 1978. The inherent structure of information as a constraint to analysis:


