Efficient application programming interface for multi-dimensional modeling data

Norman L. Jones, Robert M. Wallace, Russell Jones, Cary Butler and Alan Zundel

ABSTRACT

This paper describes an Application Programming Interface (API) for managing multi-dimensional data produced for water resource computational modeling that is being developed by the US Army Engineer Research and Development Center (ERDC), in conjunction with Brigham Young University. This API, along with a corresponding data standard, is being implemented within ERDC computational models to facilitate rapid data access, enhanced data compression and data sharing, and cross-platform independence. The API and data standard are known as the eXtensible Model Data Format (XMDF), and version 1.3 is available for free download. This API is designed to manage geometric data associated with grids, meshes, riverine and coastal cross sections, and both static and transient array-based datasets. The inclusion of coordinate system data makes it possible to share data between models developed in different coordinate systems. XMDF is used to store the data-intensive components of a modeling study in a compressed binary format that is platform-independent. It also provides a standardized file format that enhances modeling linking and data sharing between models.

Key words | data standards, finite difference method, finite element method, 3D models, 2D models

Norman L. Jones (corresponding author)
Environmental Modeling Research Laboratory,
242C Clyde Building,
Brigham Young University,
Provo UT 84602,
USA
E-mail: *njones@byu.edu*

Robert M. Wallace
Cary Butler
US Army Corps of Engineers Engineer Research and Development Laboratory,
3909 Halls Ferry Road,
Vicksburg MS 39180,
USA

Russell Jones
Alan Zundel
Aquaveo, LLC,
75 South 200 East, Suite 201,
Provo UT 84606,
USA

INTRODUCTION

One of the more costly aspects of any computational modeling effort is the management of data. A conservative estimate is that more than 50% of an entire modeling effort is involved with obtaining, cleaning, transferring and manipulating data files. The problem is exacerbated during large, multi-dimensional projects where multiple investigators, multiple data sources and long project durations can create complicated and expensive data management problems. The US Army Corps of Engineers (USACE) is particularly sensitive to data management issues because it is a large organization that hires multiple contractors to obtain and manipulate data for modeling projects. Reducing the effort required to work with data by adopting common data standards can significantly reduce the overall costs of a modeling project.

The Corps of Engineers has a long history of creating data standards for modeling and other engineering purposes. A format developed by the Hydrologic Engineering Center (HEC) in the 1980s was designed to facilitate data similarity for models utilizing paired data (HEC-DSS 2010). This format, called the Data Storage System (DSS), has become an industry standard for sharing zero-dimensional (0D) time-variant data. The Spatial Data Standards for Facilities, Infrastructure and the Environment (SDSFIE 2010) were developed for standardizing the attribution of spatial Geographic Information Systems (GIS) and Computer-Aided Design and Drafting (CADD) data. In addition to these efforts sponsored by USACE, numerous additional data formats and specifications have proven useful at alleviating many data management concerns. Despite these earlier efforts, there is currently no standard method to support the data associated with complex, multi-dimensional, water-resource-related computational modeling data.

doi: 10.2166/hydro.2011.013

The problem is not unique to the water resources community and alternative solutions have been presented, especially within the ocean/atmospheric modeling community. For instance, the NetCDF (Rew & Davis 1990; NetCDF 2010) and GRIB (GRIB 2010) data formats have been developed to handle large gridded datasets common to ocean and atmospheric modeling. The Hydro-NEXRAD system described by Krajewski *et al.* (2010) is designed for managing multi-terabyte datasets related to rainfall data. Gourbesville (2010) discusses challenges related to integrating high-resolution bathymetry and terrain data with simulation models. However, because none of these systems support boundary-fitted grids or unstructured meshes, they do not provide the necessary capability to support the requirements designated in this research.

One of the primary limitations of previous work is that few have been adapted to support unstructured meshes. Unstructured meshes are particularly well adapted to non-uniform topography and undulating riverine and coastal shorelines and have widespread use within the water resources community. Unstructured meshes must therefore be supported in any new data model.

To address these issues, the eXtensible Model Data Format (XMDF) was jointly developed by the US Army Engineer Research and Development Center (ERDC) and the Environmental Modeling Research Laboratory (EMRL) at Brigham Young University (BYU), Provo, UT. XMDF is designed to be open architecture, platform-independent, memory-efficient and computationally inexpensive to use. It provides support for a wide variety of modeling frameworks including meshes, grids and cross sections. New data formats are only as useful as they are simple to implement. With this in mind, XMDF is designed to be relatively easy to use and require minimal effort to incorporate into existing computational models.

This paper will review previous efforts to develop modeling data formats and will then document the development of the XMDF Application Programming Interface (API) and data architecture. Finally it will show a sample implementation and document the performance of XMDF with regards to data compression and access performance. The research described in this paper is relevant to both researchers and industry practitioners because it illustrates the design and implementation of an API using state-of-the-art data input/output (i/o) libraries and techniques. The API can be freely downloaded for use by both researchers and practitioners from a public domain web server (XMDF 2010a).

PREVIOUS WORK

Other efforts have been conducted to produce a common data standard for water resources modeling. A few of the more recent of these efforts are discussed in the following subsections.

ArcHydro

ArcHydro was developed by a consortium of industry, government and academia researchers as a GIS-based data structure that links hydrologic data to water resources models and decision-making methods. Water resources data are described using standard GIS methodology for consistency and to more easily share data between models and decision tools. ArcHydro provides the attribute structure so that the GIS database for a particular location contains most of the information necessary for a computational model to analyze the hydraulics or hydrology of that same location. The strength of ArcHydro is that it fuses the geospatial and temporal water resource data to support hydrologic analysis and modeling (Maidment 2002). ArcHydro is based on the ESRI GeoDatabase design, which prevents its use on computational platforms other than MS Windows, such as on Linux clusters or larger supercomputing resources. Furthermore, ArcHydro does not directly support multi-dimensional computational domains such as unstructured meshes and non-rectilinear, orthogonal grids.

HEC-DSS

The US Army Engineer Hydrologic Engineering Center (HEC) Data Storage System, or DSS, is a database designed to efficiently store and retrieve scientific data that are typically sequential. Such data types include, but are not limited to, time series data, curve data or spatially oriented gridded data. DSS provides both the database structure and an API for efficient incorporation within computational

models. The system was designed to make it easy for users and application programs to retrieve and store data. DSS is incorporated into most of HEC's major application programs (USACE 1991; HEC-DSS 2010). Newer versions include spatial geo-referencing information and support for grids, but because DSS was designed for single-dimensional data, it is not well-suited for multi-dimensional data.

NetCDF

NetCDF (Network Common Data Form) is a set of interfaces for array-oriented data access and a freely distributed collection of data access libraries for C, Fortran, C++, Java and other languages. The NetCDF libraries support a machine-independent format for representing scientific data. Together, the interfaces, libraries and format support the creation, access and sharing of scientific data. NetCDF data are self-describing, portable, directly accessed, expandable and sharable. UniData, the organization that created and administers NetCDF, is undergoing the process of transforming NetCDF to utilize the Hierarchical Data Format version 5 (HDF5) developed by the National Center for Supercomputing Applications (Sinha *et al*. 2005; HDF5 2010). This effectively turns NetCDF into an API for HDF5 that is specific to structured, matrix data. All computer software that already utilizes the NetCDF programming structure can utilize the HDF5-based version with only minor programming modifications. While NetCDF is exceptionally well-suited for data that are organized in array format, like that associated with grids, it is not easily adapted for unstructured data associated with unstructured meshes or triangulated irregular networks (TIN).

SEDRIS

SEDRIS was initiated in 1994 as a co-sponsored activity by the Simulation Training and Instrumentation Command (STRICOM, now PEO STRI) and the Defense Advanced Research Projects Agency (DARPA) to provide solutions to the complex problem of environmental data representation and interchange for networked heterogeneous applications. Distributed simulation requires a common representation of 'place' relative to the application view of the common world. A common representation of place is a precondition to interoperability. SEDRIS was initiated to provide a capability for solving the environmental data representation and interchange problems for heterogeneous systems that would be required to interoperate in networked and distributed applications (Harris 2008; SEDRIS 2010). Because SEDRIS is focused on the representation of environmental data required for distributed simulation, it does not address the unique data requirements of computational modeling. For instance, there is no way within SEDRIS to represent the node configuration and element topology used in finite element modeling.

XMSF

The Extensible Modeling and Simulation Framework (XMSF) was developed jointly by a number of Department of Defense (DoD) organizations. It is defined as a set of web-based technologies, applied within an extensible framework, that enables a new generation of modeling and simulation (M&S) applications to emerge, develop and interoperate (Brutzman *et al*. 2002; XMSF 2010). XMSF augments SEDRIS by providing the framework by which new M&S systems can be built. SEDRIS provides the mechanism by which simulation worlds can be defined while XMSF provides the mechanism by which the defined worlds can communicate and be linked together in a simulation environment. Because XMSF is a framework for web-based protocols to communicate, it does not solve the fundamental data requirements of computational modeling of extremely large datasets.

XDMF

Of particular interest (aside from the similarity in names) for this paper is the eXtensible Data Model and Format (XDMF). XDMF is part of the Interdisciplinary Computing Environment (ICE) developed by the Army Research Laboratory (Clarke & Namburu 2002; Mehrotra & Zubair 2002; XMDF 2010a). XDMF provides a common method to pass values and metadata in a standard fashion between application modules. XDMF views data as consisting of two basic types: light data and heavy data. Light data are both metadata and small amounts of values. Heavy data

typically consist of large arrays of values. All light data in XDMF are stored using the eXtensible Markup Language (XML). Heavy data are currently confined to HDF5 but will eventually include formats like Oracle and Plot3D. XDMF is both a data model and format. That means that the size and shape of the data are described, as is the intended use (i.e. XYZ position versus three-dimensional vectors). While not required, a C++ API is provided to read and write XDMF data. This API has been 'wrapped' so it is also available in both Tcl and Java. The API contains a built-in XML parser to handle the light data. Heavy data are not handled explicitly by XDMF. It is the responsibility of each application model to provide the mechanism to parse the HDF5 file. XDMF describes how the HDF5 data are organized and presents a pointer to where the HDF5 file is located. XDMF is probably the most similar effort to that described in this paper. However, the critical difference between XDMF and the effort described here is that XDMF does not free the computational program from inserting low-level HDF5 data calls within their code. As a result, there is not a complete abstraction between the data and the computational program, which was one of the design requirements for the current research.

GeoVRML

The GeoVRML consortium has developed a data standard to bridge the gap of standardized, geographic representation of multi-dimensional data (Reddy *et al*. 2001; GeoVRML 2010). GeoVRML is built upon the Virtual Reality Markup Language (VRML), which provides a common method for distributing 3D data models over the Internet. GeoVRML provides both geo-referencing and time-referencing schemas to standard VRML. GeoVRML also provides a method for providing data at multiple levels of detail, depending on the extents of the viewing area. While GeoVRML is well-suited for geographic data, it was not designed for computational modeling.

While extremely useful, none of these existing data formats completely fulfills the criteria established for future multi-dimensional computational modeling within the Corps of Engineers. For instance, ArcHydro and DSS do not address the multi-dimensional issue, and NetCDF does not readily adapt to handle unstructured meshes. Furthermore, neither of these systems can handle files larger than 2 GB. While workarounds are being examined for this limitation, this is a critical limitation that allowed for the development of a completely new data specification. XMSF and SEDRIS were designed for use in military modeling and simulation and do not translate well into computational modeling. GeoVRML is designed for distributing 3D topology over the web and is also not well-suited for handling computational data. XDMF provides the most compelling methodology for handling the data needs of computational hydrodynamic modeling. However, the lack of a complete abstraction between the data and the computational engine necessitates a new and separate system for data handling.

DESIGN OBJECTIVES

In the design of XMDF, it was determined that the following features would be essential to the success of the project.

Ease of use/implementation. The success or failure of any attempt at standardization will ultimately be judged by how widespread within the targeted organization the protocol is adapted. To overcome reluctance on behalf of model developers, the file format must be easy to implement. A complicated structure requiring an extensive overhaul of legacy modeling codes is doomed to failure from the beginning.

Efficiency. Perhaps the most important factor in ensuring widespread usage of XMDF is to provide numerous performance benefits beyond the data sharing benefits to be derived from the usage of a common file format. If the XMDF tools result in more efficient modeling code, model developers will be further motivated to adopt the standard.

Platform independence. The files written to the XMDF standard must be compatible with both the UNIX and PC platforms. Data written to one platform must be readable from the other platform.

Support of multiple languages. The tools associated with the file formats must be accessible from multiple programming languages. At a minimum, the C/C++ and FORTRAN languages must be supported.

Extensibility. The format should be developed in a fashion that will allow future modifications and enhancements to the format to be implemented without rendering obsolete files written according to previous versions of the standard.

Data abstraction. In order to ensure ease of implementation and extensibility, the file formats should support data abstraction. This ensures that the model developer/user does not need to be concerned with low-level implementation details. Rather, a simple interface to the data is provided and the implementation details are hidden behind this interface. This makes it possible to modify the file formats as new technologies are developed without affecting the code written by model developers to store and retrieve data.

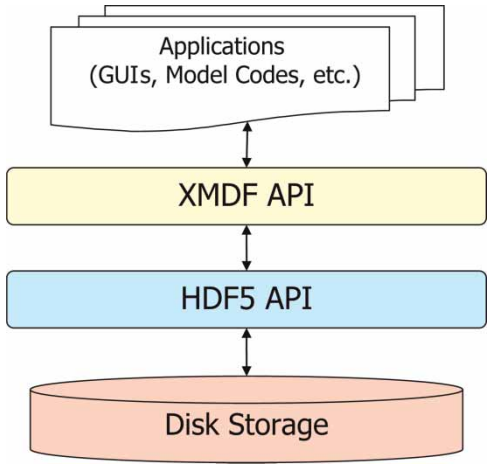


Figure 1 | Interface layering between applications and disk storage using XMDF and HDF5 APIs.

APPLICATION PROGRAMMING INTERFACE

After careful consideration of these design goals, it was concluded that XMDF should be delivered as an API rather than a prescribed file format. The API approach satisfies many of the design goals listed previously. An API is easy to implement since the model developer can focus on a simple set of subroutines and functions to store and retrieve the data rather than writing the file I/O code from scratch. The API allows for performance enhancements since complex functionality such as data compression and bit-swapping for binary file input/output (I/O) can be hidden behind the API. The API also allows for data abstraction since the API is, by definition, an interface designed to hide implementation details.

The XMDF API is built as an extension of the HDF5 library. HDF5 is a general-purpose library and file format for storing scientific data developed by the National Center for Supercomputing Applications (NCSA) at the University of Illinois. HDF5 was created to address the data storage and retrieval needs of scientists and engineers involved with high-performance computing. It consists of a low-level API for storing and retrieving data objects to a binary database. HDF5 is cross-platform-compatible because the 'little-endian/big-endian' byte swapping is handled automatically by the API. It supports multiple types and levels of automatic data compression, depending on the type of data being stored and the desired tradeoff between file size and speed of storage/retrieval. HDF5 does not impose data size limits, other than the available storage space on the target media.

The XMDF API is built on top of the HDF5 library, as illustrated in Figure 1. The model data are stored on disk using the HDF5 format. The low-level file I/O is handled with the HDF5 API. The XMDF API is built on top of the HDF5 API and provides a simpler interface to the data. For example, the XMDF API includes a set of simple subroutines for saving a finite element mesh. These XMDF subroutines receive the finite element data and then use the native HDF5 API and subroutine calls to store the data into the low-level hierarchal structure utilized by HDF5. The XMDF API provides a buffer between the model codes and the low-level HDF5 library. This buffer makes the file format easier to implement and maintain.

XMDF has been implemented within a number of computational codes developed and maintained by the US Army Corps of Engineers. These models include the Adaptive Hydraulics Model (Tate *et al*. 2006; Gambucci 2009; ADH 2010) and the Gridded Surface Subsurface Hydrologic Model (Downer & Ogden 2004; GSSHA 2010). The advantages for this data model are most realized when operating on a shared memory architecture or symmetric multi-processing machine. Results on parallel or distributed architecture computers has been mixed because the overhead of re-constructing the data into a single volume can negate the advantages of speed and compressibility. Further research in this area is warranted.

DATA TYPES SUPPORTED

Theoretically, all data associated with a computational model could be stored in XMDF/HDF5 format. However, converting the entire set of source code related to file I/O to the XMDF format would require a substantial amount of work for each model and would not be necessary in order to achieve the benefits associated with XMDF. Rather, XMDF is used to store the subset of the model data that is the bulkiest and requires the most disk storage. This subset includes the model geometry, array-based properties and solution data (datasets). Model geometry includes meshes, grids and cross-sectional data.

Meshes

XMDF supports 1D, 2D and 3D finite element meshes. Both the element topology and nodal coordinates are saved to the file. Since some models utilize elements of different dimensions in a single simulation, any combination of element types can be combined in a single file. Each element type is identified by a code. The element types currently supported in XMDF are shown in Figure 2. These represent the types most commonly used in water resources modeling. It is anticipated that additional types may be added in the future based on feedback from users.

Grids

The types of grids supported in XMDF are illustrated in Table 1. Both 2D and 3D grids are supported. The computational points can coincide with the cell corners, centers or faces. Grids can be rotated with respect to the global *XYZ* axes, and the relative orientation of the rows, columns and layers (*IJK* axes) can be user-defined. 2D grids can be either Cartesian or curvilinear.

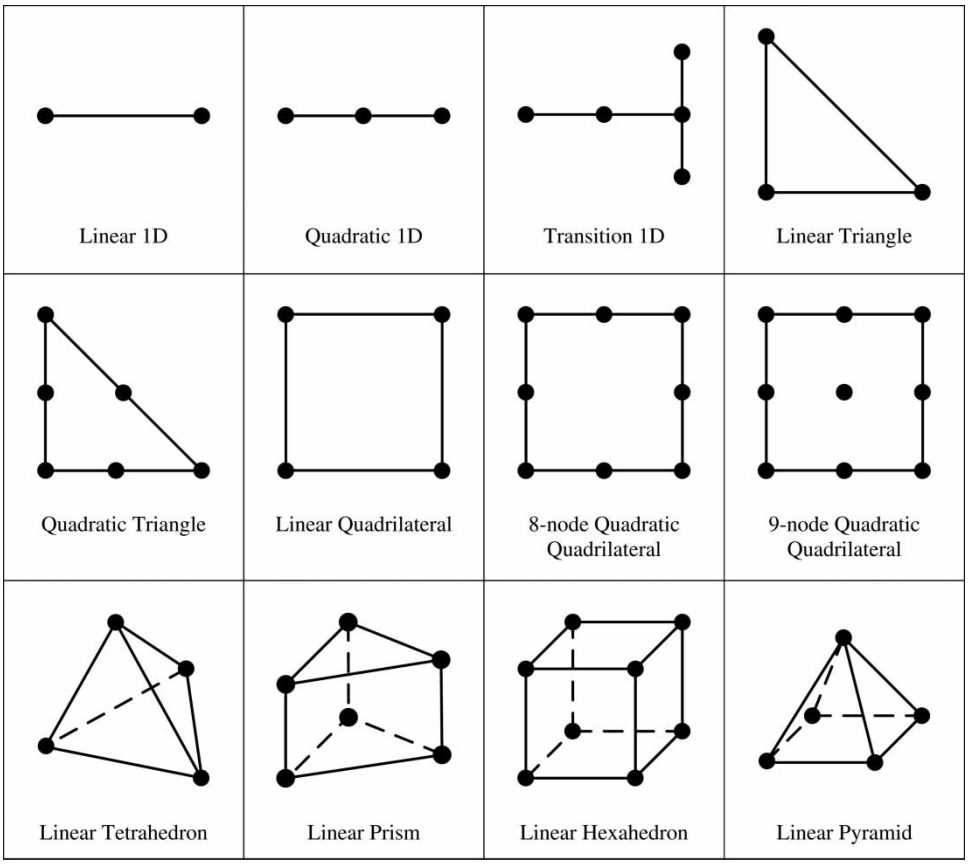
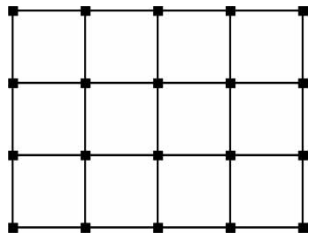
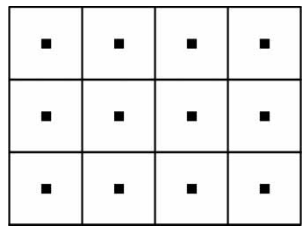
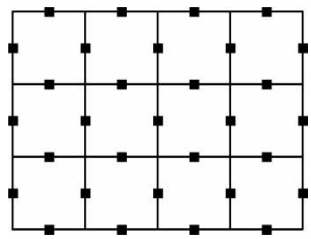
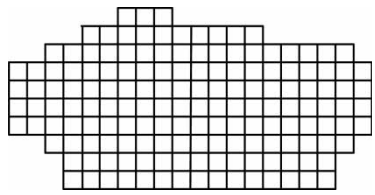
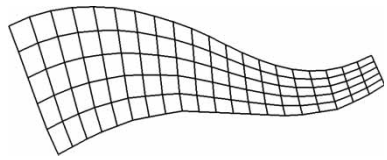
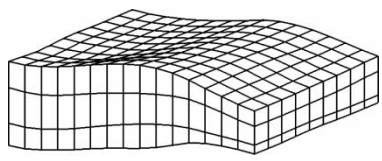


Figure 2 | Element types supported in XMDF.

Table 1 | Grid types supported in XMDF

Type	Description	Sample
Mesh-centered	Computational points are located at the corners of the grid cells (2D and 3D)	
Cell-centered	Computational points are located at the centers of the grid cells (2D and 3D)	
Face-centered	Computational points are at the centers of the faces of the grid cells (2D and 3D)	
Cartesian	Row, column and layer boundaries are orthogonal (2D and 3D)	
Curvilinear	Row, column and layer boundaries are **not** required to be orthogonal. A set of coordinates is assigned to the corner of each grid cell (2D and 3D)	
Extruded	A special type of 3D grid that where each layer represents an 'extrusion' of a 2D grid. The extruded 2D grid can be orthogonal or curvilinear	

3D grids can be Cartesian, curvilinear or extruded 2D grids. Options to specify 3D layers on extruded 2D grids include sigma stretch, Cartesian, corner-specified z value for every layer (curvilinear at corners) and column-specified z value for every layer (curvilinear at mid-sides/cell centers). Sigma stretch grids have varying z values for the top and bottom of each column. Every layer in a sigma stretch grid has a constant percent

thickness of the column thickness. As an example, the MODFLOW groundwater model (Harbaugh *et al*. 2000) uses a 2D Cartesian grid extruded to a 3D grid using the curvilinear-at-cell-centers option.

Cross sections

Cross-sectional data are associated with commonly used 1D river and coastal models such as HEC-RAS, and WSPRO (Shearman 1990; HEC-RAS 2001). Cross-sectional data define channel bathymetry and profile (longitudinal) lines that can be used to represent centerline, bank line or other 'stream' paths within a stream channel or coastline (Figure 3). Line (material properties) and point (thalweg, bank) properties associated with cross sections are stored as well as other attributes.

A 1D cross section consists of a set of distance (station) and elevation values. If the cross section is to be used in conjunction with terrain data, it must also be geo-referenced. This means that the distance or station value can be converted into X and Y values, and the elevation is assumed to be a Z value. The geo-referencing can be provided on a point-by-point basis, in which case each point of a cross section has an X and Y coordinate defining its Cartesian location. Geo-referencing can also be established using one or two points on the cross section. Single-point geo-referencing provides the X and Y values associated with a distance along the section as well as an azimuth. Two-point geo-referencing requires the specification of two XY pairs as well as two distances along the section.

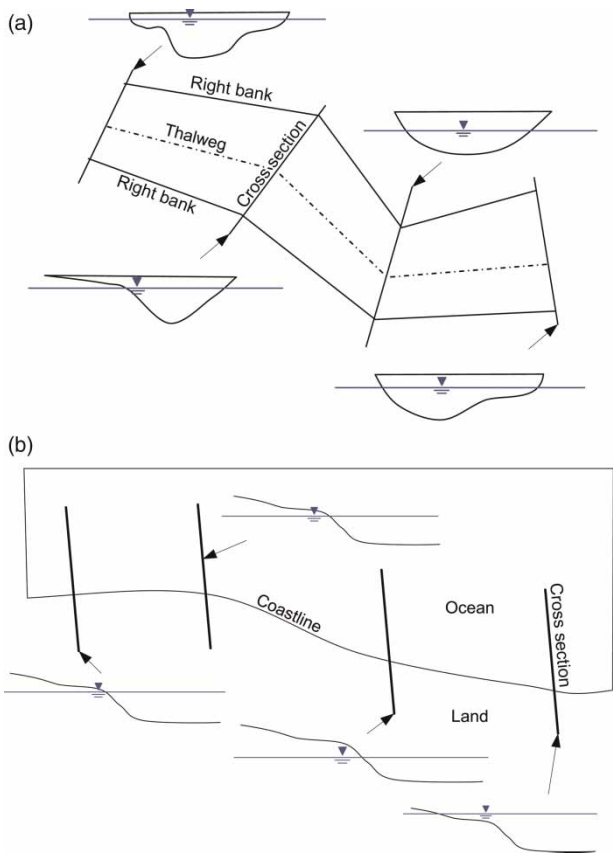


Figure 3 | Cross-sectional data. (a) Riverine cross sections and (b) coastline cross sections.

Array-based properties

In addition to model geometry, XMDF provides a simple mechanism for storing array-based model properties such as hydraulic conductivity or roughness coefficients. These arrays can be floats, double precision floats, integers or strings.

Datasets

A dataset is similar to an array-based property except that each item can be either a scalar or a vector and datasets can be either steady state or transient (one array per time step). Datasets are generally used for model solutions. Scalar datasets have one value for each entity in a mesh or grid. Vector datasets may have either two (x, y) or three components (x, y and z), depending on whether the data are 2D or 3D.

Time-varying datasets may begin at a specific reference time or be relative times from an arbitrary zero hour. Reference times are specified in Julian days. Time values for specific time steps are given as time offsets from either zero or from the reference time. The units for the offset times can be given in days, hours, minutes or seconds.

Coordinate systems

XMDF also supports storage of coordinate system data with grids, meshes or sets of cross sections. This makes it easier to import the data into other systems such as geographic information systems or to combine model data defined in two different coordinate systems. XMDF supports most of the coordinate systems used within the United States and a smaller number of those used internationally.

METADATA

XMDF also includes an option for associating metadata with each of the object types described above. The metadata fields in XMDF were designed to be compatible with the Federal Geographic Data Committee (FGDC) standards (FGDC 2010). The XMDF metadata template is not fully 'compliant' with the standard, since we did not adopt all of the FGDC fields; but the fields in the XMDF metadata are a subset of the FGDC standard. The XMDF metadata fields include: title, abstract, purpose, creation date, last modified date, contact information and coordinate system. A metadata group can be added to any of the supported object types (meshes, grids, etc.) using the XMDF API.

ORGANIZATION

Data are organized in an XMDF file in a hierarchical fashion using 'groups'. A group is similar in concept to a folder or directory on a file system. Each group represents an unstructured mesh (or set of scattered data points), a structured grid (either Cartesian or curvilinear) or a set of cross sections. Each of these groups may include one or more subgroups with property arrays or datasets. A sample mesh group is shown in Figure 4. The ability to organize data in a hierarchical fashion is one of the basic features of the HDF5 library, upon which XMDF is built. However, the file structure is automatically organized by the XMDF API. The user simply needs to pass the data to the XMDF API using the FORTRAN/C interface.

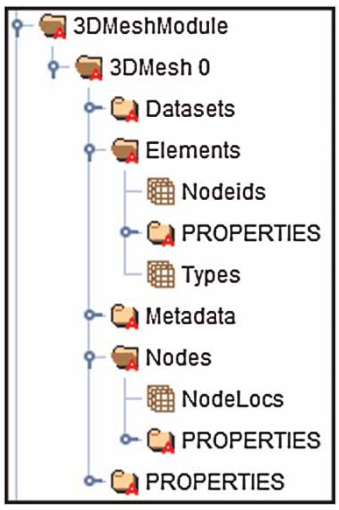


Figure 4 | Mesh group layout.

TYPICAL USAGE

Implementing the XMDF file format and API can be accomplished by a model developer in a few simple steps. For a typical scenario, a model code will read the model geometry and write out solution datasets. For simple models, the geometry can be entered into an HDF5 file using the HDF5 Viewer. For more complex models, a graphical user interface can be used to build the model geometry. The Groundwater Modeling System (GMS), Surface Water Modeling System (SMS) and Watershed Modeling System (WMS) interfaces have been modified to support the XMDF formats (Aquaveo 2010a–c). Any of the grid/mesh types supported by XMDF can be built and exported from GMS/SMS/WMS. To support the XMDF format, the file format utilized by the model code would need to be modified so that the model geometry portion of the input could be defined in a separate file. The rest of the model input (boundary conditions, material properties, analysis options, etc.) could remain in the original format in an ASCII file. Typically, the largest part of the model input is the model geometry. This is especially true for finite element models. The portion of the model code that reads the model geometry would then be altered to read in the model geometry using the functions/subroutines provided in XMDF. The output portion of the model code would also be modified to write out solution datasets to an XMDF/HDF5 binary file. Once again, a few simple functions/subroutines can be used to output time-step data as the solution is computed. Several sets of sample source code are shown in the XMDF documentation (XMDF 2010b).

PERFORMANCE

One of the primary design objectives for the XMDF API is performance. Model developers are more likely to adopt the XMDF format if there are substantial improvements in file I/O efficiency. Initial applications of XMDF have confirmed the performance gains. Part of this is due simply to

the fact that the XMDF format is binary. Converting from a traditional ASCII file format to a binary version alone will result in smaller file sizes and faster file I/O. However, additional gains are realized due to the compression algorithms included in HDF5 (Yeh *et al.* 2002). HDF5 supports two built-in compression algorithms, GZIP and SZIP. For scientific data, SZIP has a better compression ratio and compression time than GZIP compression and two other commonly used algorithms: Run Length Encoding (RLE) and Adaptive Huffman. A comparison of these algorithms is shown in Table 2. The ratio listed in the last column represents the size of the original data divided by the size of the compressed data. It should be noted that SZIP encoding is proprietary and is subject to a licensing fee. Because of this, XMDF does not use SZIP as a compression option but can read files written using SZIP compression. GZIP compression can have a compression level from 1 to 9. Preliminary tests using GZIP indicate that, for most types of data, compression levels greater than 1 take significantly longer to process with little reduction in size. The performance of HDF5 has also been tested on a series of models. A subset of the results is shown in Tables 3 and 4. The numbers shown in parentheses represent the compression efficiency relative to the ASCII format. The rightmost column in each table represents the resulting file size using the GZIP algorithm with a compression level of 1 on a scale of 1–9. Higher compression levels result in even smaller file sizes but with longer read/write times. Read/write times are not shown in the table, but all objects listed are read/saved using XMDF in just a few seconds, compared to times as long as several minutes for the ASCII formats.

Table 2 | Relative performance of compression algorithms using a Pentium II 300 MHz processor on a 343 MB block of data (Yeh *et al.* 2002)

Type	Compression time (s)	Decompression time (s)	Ratio
RLE	85.7	41.6	1.6
Adaptive Huffman	558.4	574.9	2.28
Gzip	273.1	38.3	2.37
Szip	71.6	63.6	2.8

Table 3 | XMDF performance – finite element meshes

2D/3D	No. nodes	No. element	Size (MB) ASCII	XMDF	XMDF C1*
2D	8060	15,786	0.9	0.6 (33%)	0.3 (67%)
2D	1,002,001	1,000,000	74.7	58.7 (21%)	11.9 (84%)
3D	1,69,260	315,720	25.4	16.4 (35%)	5.4 (79%)
3D	4,000,080	7,582,640	453.1	376 (17%)	126 (72%).

*Compression level = 1 (out of nine available levels)

Table 4 | XMDF performance – datasets

No. pts	Transient	Size (MB) ASCII	Binary	XMDF	XMDF C1
10,000	No	0.09	0.04 (56%)	0.05 (44%)	0.04 (56%)
250,000	No	0.7	0.9 (−29%)	1 (−43%)	0.1 (86%)
6160	Yes	2.2	1 (55%)	1 (55%)	0.5 (77%)

The structured binary format used by XMDF also results in performance gains due to random access. This is particularly useful for model linking where one model needs to extract data from the output file of another model. The API can be used to extract dataset values for specific entities over a range of time steps as well as a range of entities within a specific time step. With an ASCII-formatted file, the file must be read sequentially until the desired data are located. With the XMDF API, it is possible to jump directly to the proper location in the file, resulting in substantial time savings.

CONCLUSIONS

This paper presents a new API for storing data associated with water resources modeling studies. This API is built upon HDF5 and is a generic way to describe multi-dimensional numerical model data and associated datasets and properties. The XMDF format/API provides a number of benefits.

A common data format makes it easy to share data between models and pre- and post-processing tools. Prior to this effort, an expensive burden was placed upon pre- and post-processors to support multiple, model-specific file formats.

Due to the use of HDF5, the API automatically performs conversions for numerical and string formats due to

platform, precision and language inconsistencies. Big/Little endian conversions are performed for platform independence. Floats can be automatically converted from between different orders of precision (i.e. 32-bit to 64-bit float). Strings are automatically converted based upon how they are stored in C versus FORTRAN.

The API is designed to maintain backward compatibility. The library performs versioning automatically so data files do not become unusable in the future. The API interface makes it possible to adopt the format with minimal effort and the HDF5-based format results in substantially faster file I/O and much smaller file sizes. The XMDF API and documentation can be downloaded free of charge (XMDF 2010a).

ACKNOWLEDGEMENTS

This work was funded by the US Army Engineer Research and Development Center in Vicksburg, MS. Permission to publish this paper was granted by the Chief of Engineers.

REFERENCES

ADH 2010 *Adaptive Hydraulics Model.* Coastal and Hydraulics Laboratory, US Army Corps of Engineers Research and Development Center, Vicksburg, Mississippi. Available from: http://chl.erdc.usace.army.mil/adh (accessed 1 September 2010).

Aquaveo 2010a *Groundwater Modeling System (GMS),* Version 7.1, Aquaveo LLC, Provo, Utah.

Aquaveo 2010b *Surface Water Modeling System (SMS),* Version 10.1, Aquaveo LLC, Provo, Utah.

Aquaveo 2010c *Watershed Modeling System (WMS),* Version 8.3, Aquaveo LLC, Provo, Utah.

Brutzman, D., Zyda, M., Pullen, J. M. & Morse, K. L. 2002 Extensible Modeling and Simulation Framework (XMSF): challenges for web-based modeling and simulation. In *Findings and Recommendations Report, Technical Challenges Workshop, Strategic Opportunities Symposium, 22 October, George Mason University, Fairfax, Virginia*.

Clarke, J. A. & Namburu, R. R. 2002 A distributed computing environment for interdisciplinary applications. *Concurrency Comput. Practice Experience* **14** (13–15), 1161–1174 (*special issue: Grid Computing Environments*).

Downer, C. W. & Ogden, F. L. 2004 GSSHA: a model for simulating diverse streamflow generating processes. *J. Hydrol. Eng.* **9** (3), 161–174.

FGDC 2010 *The Federal Geographic Data Committee.* Available from: http://www.fgdc.gov (accessed 1 September 2010).

Gambucci, T. 2009 ADH=fast and stable 2D finite element model. In *Proceedings of the World Environmental & Water Resources Congress, 17–21 May, Kansas City, Missouri*, ASCE, Reston, Virginia, pp. 2816–2822.

GeoVRML 2010 GeoVRML.org. Web3D Consortium. Available from: http://www.ai.sri.com/geovrml/ (accessed 1 September 2010).

Gourbesville, P. 2010 Data and hydroinformatics: new possibilities and challenges. *J. Hydroinf.* **11** (3–4), 330–343.

GRIB 2010 *Office Note* 388 GRIB. National Centers for Environmental Prediction. Available from: http://www.nco.ncep.noaa.gov/pmb/docs/on388/ (accessed 1 September 2010).

GSSHA 2010 *Gridded Surface Subsurface Hydrologic Analysis.* Coastal and Hydraulics Laboratory, US Army Corps of Engineers Research and Development Center, Vicksburg, Mississippi. Available from: http://chl.erdc.usace.army.mil/gssha (accessed 1 September 2010).

Harbaugh, A. W., Banta, E. R., Hill, M. C. & McDonald, M. G. 2000 *MODFLOW-2000, The U.S. Geological Survey Modular Ground-water Model – User Guide to Modularization Concepts and the Ground-Water Flow Process.* U.S. Geological Survey Open-File Report 00-92, United States Geological Survey, Reston, Virginia.

Harris, R. 2008 Interchanging simulation databases with third parties using SEDRIS. *Aeronaut. J.* **112** (1130), 207–212.

HDF5 2010 *HDF5 Home Page.* The HDF Group. Available from: http://www.hdfgroup.org/HDF5/ (accessed 1 September 2010).

HEC-DSS 2010 *Hydrologic Engineering Center Data Storage System.* US Army Corps of Engineers Hydrologic Engineering Center, Davis, California. Available from: http://www.hec.usace.army.mil/software/hec-dss/hecdss-dss.html (accessed 1 September 2010).

HEC-RAS 2001 *HEC-RAS River Analysis System Hydraulic Reference Manual Version 3.0*. US Army Corps of Engineers, Institute for Water Resources Hydrologic Engineering Center, Davis, California.

Krajewski, W. F., Kruger, A., Smith, J. A., Lawrence, R., Gunyon, C., Goska, R., Seo, B. C., Domaszczynski, P., Baeck, M. L., Ramamurthy, M. K., Weber, J., Bradley, A. A., DelGreco, S. A. & Steiner, M. 2010 Towards better utilization of NEXRAD data in hydrology: an overview of hydro-NEXRAD. *J. Hydroinf.* **13** (2), 255–266.

Maidment, D. R. 2002 *ArcHydro: GIS for Water Resources.* ESRI Press, Redlands, California.

Mehrotra, P. & Zubair, M. 2002 *XML Based Scientific Data Management Facility.* ICASE-2002-45; NAS 1.26:211968; NASA/CR-2002-211968. ICASE.

NASA Langley Research Center, Hampton, Virginia. Available from: http://www.cs.odu.edu/~mln/ltrs-pdfs/icase-2002-45.pdf.

NetCDF 2010 *NetCDF (Network Common Data Form)*. Unidata. Available from: http://www.unidata.ucar.edu/software/netcdf/ (accessed 1 September 2010).

Reddy, M., Iverson, L., Leclerc, Y. & Heller, A. 2001 GeoVRML: open web-based 3D cartography. In *Proceedings of the International Cartographic Conference (ICC2001), 6–10 August, Beijing*. http://icaci.org/documents/ICC_proceedings/ICC2001/icc2001.htm).

Rew, R. K. & Davis, G. P. 1990 NetCDF: an interface for scientific data access. *Comput. Graphics Appl.* **10** (4), 76–82.

SDSFIE 2010 *Spatial Data Standards for Facilities, Infrastructure and Environment*. Defense Installations Spatial Data Infrastructure (DISDI) Group. Available from: http://www.sdsfie.org/ (accessed 1 September 2010).

SEDRIS 2010 *SEDRIS Technologies: The Source for Environmental Representation and Interchange. Synthetic Environment Data Representation and Interchange Specification*. DoD Modeling and Simulation Coordination Office. Available from: http://www.sedris.org/ (accessed 1 September 2010).

Shearman, J. O. 1990 *Users Manual for WSPRO – A Computer Model for Water Surface Profile Computations*. Report No. FHWA-IP-89-027, Federal Highway Administration, Denver, Colorado.

Sinha, R. R, Mitra, S. & Winslett, M. 2005 Format independent data management system for scientific data. In *Proceedings of the 3rd International Workshop on Storage Network Architecture and Parallel I/O (SNAPI05). September 18, 2005, Saint Louis, USA*.

Tate, J. N., Berger, R. C. & Stockstill, R. L. 2006 Refinement indicator for mesh adaption in shallow-water modeling. *ASCE J. Hydraul. Eng.* **132** (8), 854–857.

USACE 1991 *HECLIB, Volume 2: HEC-DSS Subroutines, Programmer's Manual*. CPD-57. US Army Corps of Engineers Hydrologic Engineering Center, Davis, California.

Wallace, R., Pathak, K., Holland, J. P., Stuart, D., Butler, C., Richards, D. R., Fife, M., Jones, N. L. & Harris, J. 2006 Information infrastructure for integrated ecohydraulic and water resources modeling and assessment. *J. Hydroinf.* **8** (4), 317–333.

XMDF 2010a *eXtensible Data Model and Format*. Available from: http://www.xdmf.org (accessed 1 September 2010).

XMDF 2010b *XMDF on XMS WIKI*. Available from: http://www.xmswiki.com/xms/XMDF (accessed 1 September 2010).

XMSF 2010 *Extensible Modeling and Simulation Framework*. Networking and Simulation Lab, George Mason University. Available from: http://netlab.gmu.edu/XMSF/ (accessed 1 September 2010).

Yeh, P. S., Wei, X. S., Miles, L., Kobler, B. & Menasce, D. 2002 Implementation of CCSDS lossless data compression in HDF. In *Proceedings of the Earth Science Technology Conference 2002, 11–13 June, Pasadena, CA*. Available from: http://esto.nasa.gov/conferences/estc-2002/Papers/A3P2(Yeh).pdf (accessed 1 September 2010).

First received 29 February 2008; accepted in revised form 11 September 2010. Available online 7 June 2011