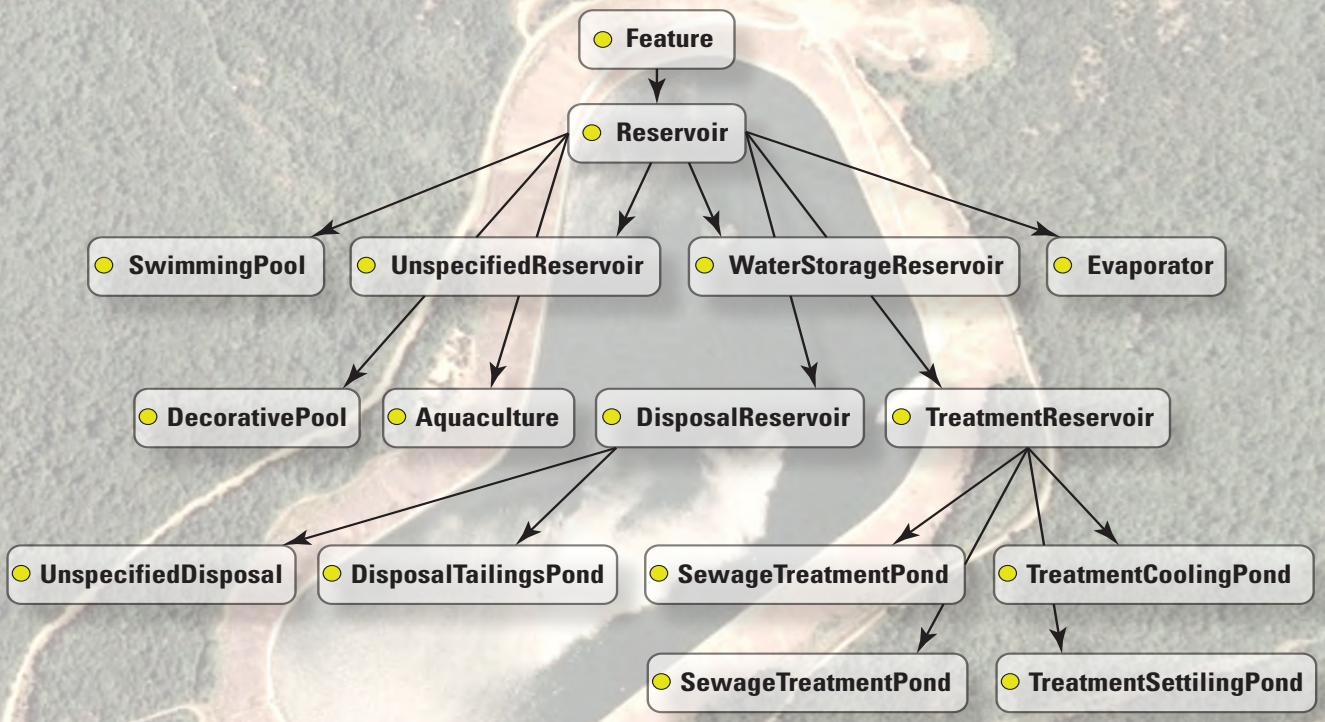
USGS
science for a changing world

A Program for the Conversion of *The National Map* Data from Proprietary Format to Resource Description Framework (RDF)



Open-File Report 2011–1142

U.S. Department of the Interior
U.S. Geological Survey

Cover. The cover design depicts a portion of *The National Map* data converted to triples.

A Program for the Conversion of *The National Map* Data from Proprietary Format to Resource Description Framework (RDF)

By Andrew Bulen, Jonathan J. Carter, and Dalia E. Varanka

Open-File Report 2011–1142

U.S. Department of the Interior
U.S. Geological Survey

U.S. Department of the Interior
KEN SALAZAR, Secretary

U.S. Geological Survey
Marcia K. McNutt, Director

U.S. Geological Survey, Reston, Virginia: 2011

Suggested citation:
Bulen, A.N., Carter, J.J., and Varanka, D.E., 2011, A program for the conversion of *The National Map* data from proprietary format to resource description framework (RDF): U.S. Geological Survey Open-File Report 2011–1142, 9 p.

Contents

Abstract....1
Introduction....1
The Semantic Web....1
The National Map....2
Objective....2
Data Conversion Approach....2
File Formats....2
Geodatabase....3
Personal GeoDatabase....3
Shapefiles....3
GML....3
N3....3
Libraries and Application Programming Interfaces (APIs)....4
GeoTools....4
Jena....4
Quantum GIS....4
GML2RDF....4
Configurations....4
Parsing GML....5
Feature Parser....5
Geometry Parser....5
GML Parser....7
Graphical User Interface....7
Data Queries....8
Results and Discussion....8
Conclusions....8
References Cited....8

Figures

1. Diagram of data conversion process....3
2. Screen shot of GML2RDF sequence diagram....4
3. Screen shot of xample of feature configuration....5
4. Screen shots showing *A*, All classes pertaining to the GUI. *B*, All classes pertaining to parsing GML files. *C*, Main classess pertaining to configurations and RDF handling....6
5. Screen shot showing Graphical User Interface for GML2RDF program....7

Abbreviations

USGS	U.S. Geological Survey
OGC	Open Geospatial Consortium
GML	Geographic Markup Language
RDF	Resource Description Framework
GIS	Geographic Information Systems
QGIS	Quantum GIS
SPARQL	SPARQL Protocol and RDF Query Language
GDAL	Geospatial Data Abstraction Library
OGR	OGR Simple Feature Library, part of GDAL library
GUI	Graphical User Interface
W3C	World Wide Web Consortium

A Program for the Conversion of *The National Map* Data from Proprietary Formats to Resource Description Framework (RDF)

By Andrew Bulen, Jonathan J. Carter, and Dalia E. Varanka

Abstract

To expand data functionality and capabilities for users of *The National Map* of the U.S. Geological Survey, data sets for six watersheds and three urban areas were converted from the Best Practices vector data model formats to Semantic Web data formats. This report describes and documents the conversion process. The report begins with an introduction to basic Semantic Web standards and the background of *The National Map*. Data were converted from a proprietary format to Geography Markup Language to capture the geometric footprint of topographic data features. Configuration files were designed to eliminate redundancy and make the conversion more efficient. A SPARQL endpoint was established for data validation and queries. The report concludes by describing the results of the conversion.

Introduction

Semantic technology holds promise for solutions to challenges in geographic information systems (GIS) data handling. Though the literature on conceptual approaches is diverse, applications of such projects are less widely communicated in geospatial semantic studies. To provide the data for tests of semantic technology potential for topographic data distribution and uses, sample data sets from *The National Map* of the U.S. Geological Survey (USGS) were converted to World Wide Web Consortium (W3C) standards for Resource Description Framework (RDF) and stored in an open-source endpoint for data queries using SPARQL, a recursive acronym meaning **S**PARQL Protocol **a**nd RDF Query Language (World Wide Web Consortium, 2008). This report describes and provides documentation for the conversion program that was used.

The data conversion draws on the broader context of the Semantic Web and *The National Map* (World Wide Web Consortium, 2010; U.S. Geological Survey, 2011a). This introductory section describes some background to Semantic Web technology, *The National Map,* and objectives for the conversion results. The conversion process and the SPARQL endpoint are described in the following sections.

The Semantic Web

The Semantic Web is a collection of data models and technologies defined by the W3C and used to more richly define and integrate data based on the meaning of those data. Data on the Semantic Web is modeled as triples, a set of nodes connected by arcs, where the nodes are data values or entities, and the arcs are relations between the data and entities (World Wide Web Consortium, 2008). In this context, entities are abstract or physical concepts, such as the Missouri River or Hydrological Unit Code (HUC) 1405000504, and a data value is the literal value of a given relation, such as a length of 4 kilometers. The languages for representing triples and modeling the relations between them are Resource Description Framework (RDF) and Web Ontology Language (OWL) (World Wide Web Consortium, 2004, 2009).

By defining relations between data and entities, the Semantic Web can integrate data from multiple sources using descriptions of how concepts are related by either explicitly defining them or reusing existing well-known definitions (Allemang and Hendler, 2008). Definitions are labeled with a Uniform Resource Identifier (URI), a universally unique identifier typically similar in construction to a Uniform Resource Locator (URL) used to access web pages. However, URIs are not necessarily dereferenceable, meaning they may not point to a particular location on the web and are instead used as a name.

Defining relations between data and entities also allows specialized software packages, called reasoners, to use the relations defined in the data to allow queries to infer new data from a given collection. For example, a river or transportation network can be modeled as a set of line segments representing the paths and the connection between the paths. For a given set of paths A, B, and C such that A is connected to B and B is connected to C, it is easy to query any database, and with no additional processing determine that A and B, and B and C are connected. However, querying to determine if B is connected

to A or if A is connected to C would require additional processing, typically by adding to the query in a traditional database. This effort would have to be replicated every time a new application is written to determine how a network of paths is connected. In semantic databases, the connection relations can be defined as having transitive and inverse properties and a reasoner can be used to allow a simple query to determine if any connection exists, as well as the ways in which the connection can be made. This greatly simplifies accessing the data by making the database "smart" and reduces the duplication of effort necessary for building "smart" queries for multiple applications.

Semantic Web data can be stored in several formats. Text files are the simplest and easiest to transport, though they are large, so they are difficult to use effectively without other Semantic Web software. In-memory graphs generally are the best way to interact with Semantic Web data, as they allow faster and more extensive timely analysis. Data in this format is volatile, and thus should be serialized regularly to avoid losing data. Dedicated triple stores operate most similarly to traditional SQL-based databases. They can be configured to work as on-disk, in-memory, or a hybrid model. Most triple stores support various types of reasoning engines that can be used to expose relations within the data that were not defined explicitly. Some examples are Virtuoso, Parliament, and Oracle (Virtuoso Universal Server, 2011) (Hebeler and others, 2009).

The National Map

The National Map is a collaborative effort built on partnerships and using standards to improve and deliver topographic information to the nation at multiple scales and resolutions. The goal of *The National Map* is to become the nation's source for trusted, current, and integrated topographic information available online for a broad range of uses. This goal and a policy of collaboration make *The National Map* cohesive with the vision for the Semantic Web, a web of broadly linked data within the Internet (World Wide Web Consortium, 2010).

The National Map includes the USGS geospatial and topographic mapping data and services with eight base data layers: transportation, structures, ortho-imagery, hydrography, land cover, geographic names, boundaries, and elevation, as well as public domain access to these and other data through a Web portal, *The National Map* Viewer. Commonly used formats for this geographic data are based on the vector and raster data models.

In current geographic information systems (GIS), table-based databases are used to store most geospatial data. Some types of table-based databases and formats that make use of table-based databases specific to geospatial applications are shapefiles and geodatabase files (ESRI, 2011). Data are stored in a series of tables from which data are extracted and relations derived by the individual programs as the data are processed. In Semantic databases, the relations are stored in the graph structure itself, requiring much less processing after extraction in instances where relations need to be determined and no additional processing in instances where relations are unimportant. Using the graph structure to store relational information allows much richer data to be stored efficiently and effectively without the increase in computational complexity that an equivalent table-based database would require.

A particular challenge for modeling the geospatial semantic web from W3C data type standards is the question of how to represent the geometric coordinate outline of topographical features. One prominent solution is the use of Geography Markup Language (GML), a standard developed by the Open Geospatial Consortium (OGC) (Portele, 2007).

Objective

The objective of this project was to convert geospatial vector data in proprietary formats to triples, while maintaining data accuracy, locational geometry, and efficient functions. The proprietary formats are the ESRI formats for shapefiles and geodatabases. The data model of the National Hydrography Database is particularly complex because of its advanced modeling capabilities (U.S. Geological Survey, 2010). The converted data must be made publicly available and useable.

Data Conversion Approach

In order to move the data from the current databases into RDF format, a data conversion process was developed to first convert the data into GML format and then to RDF files (fig. 1). The data are extracted from the database using Quantum GIS (QGIS, 2011). QGIS was chosen because it is an open source GIS software package that implements the Geospatial Data Abstraction Library (GDAL) and supports the OGR Simple Features Library formats (for example shapefile, personal geodatabase, GML) necessary for the conversion process. The graphs are read into QGIS from either shapefiles or personal geodatabase files. The separate layers of data are then written to GML version 2.1.2 format. The interoperability of GML 2.1.2 with GIS software packages was the reason it was chosen. Once in GML format, the data were then processed into RDF using a USGS developed program coded in Java called GML2RDF.

File Formats

The National Map data conversion process implements several different file types. Data are currently (2011) contained in one of a few possible ESRI formats, such as file geodatabases or shapefiles. These files are then used to create GML documents that are used for storing the geometric data in an Extensible Markup Language (XML) like format (Bray and others, 2008). Finally, the output of the conversion process is written to an N3 document, a shorthand RDF format (Berners-Lee, 2005).

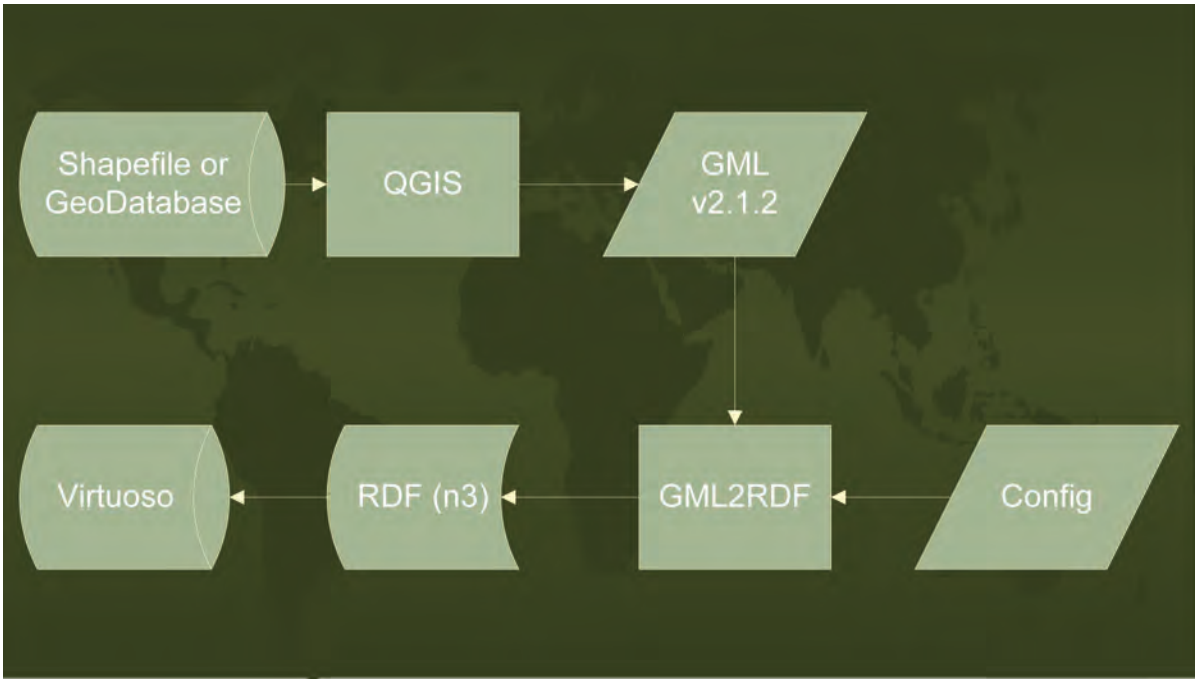


Figure 1. Diagram of data conversion process.

Geodatabase

A geodatabase is a "collection of geographic datasets of various types used in ArcGIS and managed in either a file folder or a relational database" (ESRI, 2011). Geodatabases are used by ESRI's ArcGIS software as the primary data source. Geodatabases are beneficial because of their large capacities, up to 256 TB, and their ability to be accessed concurrently by multiple editors. However, the geodatabase format is unique to ESRI's software packages and not accessible by other GIS software.

Personal GeoDatabase

Personal GeoDatabases are Microsoft Access databases with a set of tables defined by ESRI for holding geodatabase metadata along with geometries held in a column with a custom format. Unlike a file geodatabase, personal geodatabases are limited to 2 GB of data and can only be accessed by a single editor at a time, but they are interoperable with other GIS software packages.

Shapefiles

Shapefile is a geospatial vector data format for GIS software developed by ESRI. Shapefiles are used to store nontopological geometry and attribute information for spatial features. Shapefiles are easy to read and write, and usually require less disk space to store data. The shapefile specification is openly available from ESRI, allowing programs to be developed to read and write shapefiles. Support for this file format is implemented in the GDAL library allowing interoperability with other GIS software packages, as well as custom developed applications.

GML

The Geographic Markup Language is an XML based grammar for storing and expressing geographical features. Like XML, GML consists of two parts, an instance of the GML document and the schema that describes the document. This allows generic geographic data sets containing points, lines, and polygons or any combination of these to be extended to represent geographic features (for example, roads and waterways) containing both the geometric data and any attribute data associated with the feature.

N3

The N3 file format is a shorthand non-XML serialization of RDF. It is much more compact than RDF/XML and is easier to read. N3 achieves compactness and readability by incorporating features such as URI abbreviation, adding multiple objects to a single subject predicate pair by separating them with a comma and adding multiple predicates to a single subject by separating with a semicolon. This results in smaller file sizes that contain the same data as a RDF/XML file.

Libraries and Application Programming Interfaces (APIs)

The conversion program makes use of two predeveloped open source APIs for parsing GML files and handling the creation of the RDF models. The GeoTools API is used for reading the data from GML files, and the Jena API is used to contain the converted RDF data in a memory model and eventually write the data into a N3 file (Jena, 2011).

GeoTools

GeoTools is a Java API developed and maintained by the Open Source Geospatial Foundation (GeoTools, 2011; OSGeo, 2011). The API is used for parsing GML files and extracting each feature into a simple feature class containing a list of all the attributes of the feature, as well as the geometry for the feature. This simple feature class is used for parsing the GML files into RDF format.

Jena

Jena is an open source Java framework for building Semantic Web applications. Jena includes a RDF API, an OWL API, reading and writing of RDF in RDF/XML, N3, N-Triples formats, and in-memory and persistent storage, as well as a SPARQL query engine.

Quantum GIS

QGIS is a GIS program developed by OSGeo. It is used to import file formats, such as shapefiles or personal geodatabases. The graphs are loaded into QGIS and converted into feature layers. The geometries for each feature are then converted to the WGS84 spatial reference system (World Wide Web Consortium, 2003). Once the geometries are converted, each layer is then written to GML version 2.1.2. These GML files are then used as the input into the GML2RDF Java program (U.S. Geological Survey, 2011b).

GML2RDF

The GML to RDF conversion program is a Java program that uses GeoTools to parse through GML files and extract the simple features in order to process them into RDF (fig. 2). Each simple feature is then added to an RDF model using the Jena library, a semantic web framework, along with feature specific configuration files used to create the linkage between data.

Configurations

In order to take full advantage of the linked data format of the Semantic Web, configuration files were used to determine whether each data field processed from the original graphs represents a literal value or a reference to another resource and to process the attribute accordingly. Each feature type has its own configuration that contains a list of all the attributes that can be present for that feature type as well as the feature type name and an optional field for selecting which attribute will be used as the unique identifier for the feature (fig. 3). For each attribute, the configuration stores the original attribute name as it was represented in the GML files, whether or not the attribute is storing a literal value or a reference to another resource, the namespace of the resource it represents, the parent resource of the attribute, and the predicate used to store the attribute in RDF format.

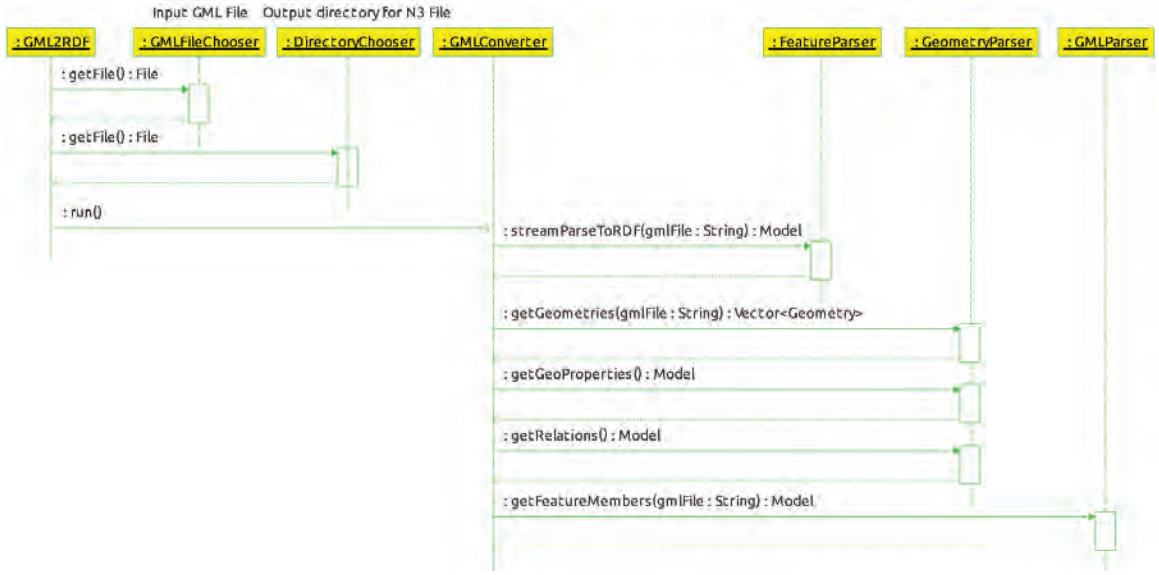


Figure 2. GML2RDF sequence diagram.

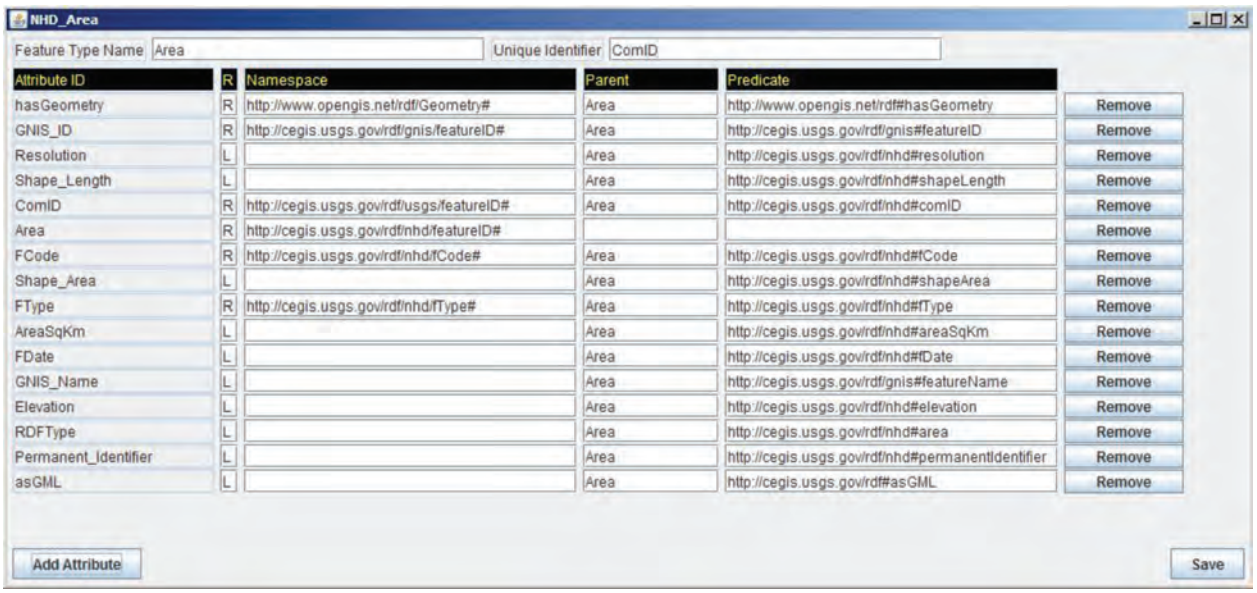


Figure 3. Example of feature configuration.

For this project, all URIs that did not come from other sources begin with http://cegis.usgs.gov/rdf/ in order to ensure that only universally unique URIs were created. The /rdf part was added to prevent a collision with any existing CEGIS pages or resources. To prevent internal collisions, each URI was then appended with the source data set, for instance / nhd. Finally, each unique concept was appended after a #. An example of a complete URI is http://cegis.usgs.gov/rdf/nhd#flowline, which was generated to represent the plain English definition taken from National Hydrography Dataset (NHD) of the term "flowline". Individual resources would be represented similarily. For instance, http://cegis.usgs.gov/rdf/nhd/featureID#_77128993 would be the resource identified with ComID 77128993 within NHD. Although this project used the above convention for generating URIs to avoid internal and external collisions, it is not necessary to do this to generate valid semantic data and not all other sources generate data using similar conventions. Therefore, URIs should always be treated as meaningless identifiers unless there is reason to believe the URI itself has significance. All data should explicitly define all relevant properties within the graph structure and not rely on URI parsing to retrieve any information.

Parsing GML

In order to limit the volume of data that is necessary to be stored in memory during the conversion process, as well as allowing the user to customize the data to be converted, the parsing of the GML files was split into separate classes (fig. 4). GML2RDF (fig. 4*A*) is the main function that implements the parser functions (fig. 4*B*) and the parser functions implement the rdfUtil classes (fig. 4*C*). The first is used to simply copy all of the attribute data from the GML file and add it to an RDF Model. The second class handles all of the features geometric data conversion. The final class is used to read in the GML and store each feature's full GML as a string in the RDF Model, making it possible to recreate the GML file from a simple SPARQL query.

Feature Parser

The feature parser class uses the GeoTools simple feature class to extract the attributes from each GML feature. The GML file is parsed a single feature at a time. Once a feature is extracted, the feature type configuration is used to copy the attributes into an RDF model. While iterating through the set of attributes, the configuration for that attribute is checked to determine if the value represents a resource or a literal. If the attribute is a literal, the value is added to its parent resource as an RDF literal value using the predicate URI from the configuration to describe what the value represents. If the attribute represents a resource a new resource is created under the namespace stored in the configuration with the value of the attribute as the identifier of the resource. A reference to the newly created resource is then added to the parent under the predicate from the configuration.

Geometry Parser

The geometry parser extracts the geometry from the features in the GML file by first extracting the simple features from the file, in a similar fashion as the feature parser; however, the feature data are not saved. Only the geometry data are extracted and added to a list of all geometries from the GML. Once all of the geometries are extracted the geometry parser can extract the geometric data and spatial relations or both depending on the user selection. Any geometric data taken from the geometry is converted to RDF format using the geometry configuration. All types of features use a single configuration file for their geometries. This is because of the fact that all geometries are stored as a separate resource in

Figure 4. *A*, All classes pertaining to the GUI. *B*, All classes pertaining to parsing GML files. *C*, Main classess pertaining to configurations and RDF handling.

the RDF model. The geometries are stored under a separate namespace as the features, but with the same identifier as the feature with which they are associated. Once each geometry has been added to the model, a resource is added to the RDF feature with which the geometry is associated, and a reference to that geometry is added to the features attributes.

Spatial relations between the features contained in the geometry parsers set also can be computed using functionality implemented in the GeoTools library. The spatial relations that can be determined are equals, touches, disjoint, contains, crosses, overlaps, covers, intersects and within from the 9-intersection model (Egenhofer and Herring, 1991). Because all of these relations are either symmetric or inverse of another relation, for example, contains and within are inverses, it is only necessary to perform a single comparison between features. This also decreases the necessary runtime for comparing large sets of geometries. The geometry located at the i^{th} position in the set is only compared to geometries in the subset $\{i+1,...,n\}$ where n is the total number of geometries. For each relation that is discovered between two geometries, a resource is added to the RDF geometry resource under the predicate representing the spatial relation.

GML Parser

The GML parser class is used for creating exact copies of the original GML data within the RDF graph. This is accomplished by reading through the GML document line by line. Each line of a feature member is added to a string until the end of the feature member is located. Once the entire feature member has been loaded, the parser searches the string for the feature type and the identification field used to represent the feature. The feature type is used to load the configuration for that feature type and find the URI of the resource. The entire string of GML is then added to the RDF feature. This makes it possible to recreate the original GML document by querying the GML field from each feature. The string of GML also is used to extract the geometric portion of the GML and add the string to the RDF feature's geometry resource under the OGC's as GML predicate. This makes it possible to query only the GML geometry of each feature.

Graphical User Interface

A graphical user interface (GUI) was also developed with the GML2RDF conversion program (fig. 5). This provides an easier interface between the user and the program. The GUI implements graphical file selections for the input GML files and the location for the output N3 formatted RDF files. The GUI also allows the user to customize the functionality of the program by selecting which conversion functions will be executed for a given set of input files. The user can select whether to convert only the feature's stored attributes, the geometric data, the spatial relations, the strings of GML, or any combination of them. This can be useful when converting large data sets that would result in output file sizes that are too large for the system to handle. By separating each section of the conversion, the resulting RDF data can be separated into smaller, more manageable files. This is possible because of the nature of a linked database. Whereas a single feature can have data stored in separate locations, each time data are loaded into a graph any data that are associated with a feature already in the graph will be added to that feature's resource rather than creating a duplicate of the resource to contain the new data. The GUI also has a console area for displaying any output information from the program. This console is used to display the files being converted, and when they finish, conversion times for individual files, total running time, and any important information, such as errors converting a file.

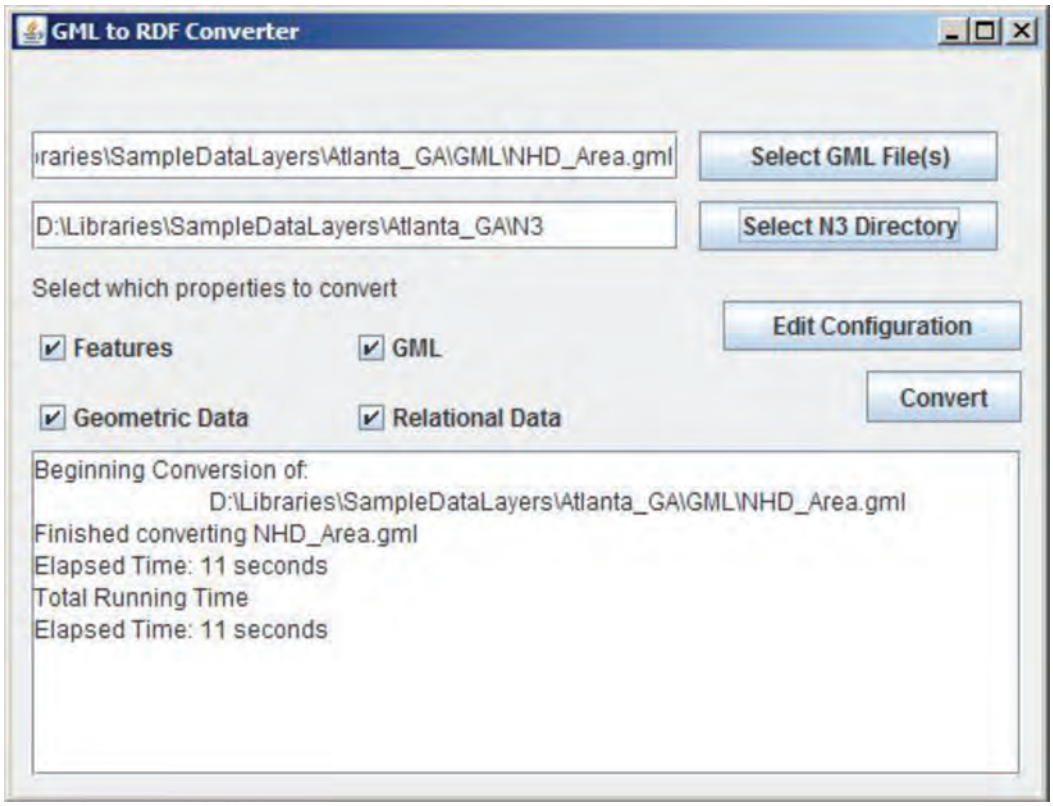


Figure 5. Graphical User Interface for GML2RDF program.

Data Queries

Semantic data primarily are queried using SPARQL. Similar to Structured Query Language (SQL), it is a specifically formatted plain-text query asking the triple store or local program for a specific piece or pieces of information from the database. A typical query looks something like "select distinct ?Concept where {[] a ?Concept}". This query would return a list of nodes in which some node (labeled as [] in the query, meaning it can be any node) is connected to another node we want to save (in this case, to store it in a variable called ?Concept, although the name of the variable is unimportant) by the relation "a", which is shorthand for "is a". So, in plain English, this query requests each distinct concept in the database where concepts are defined as the object in relation with the predicate 'a'. The result set lists each type of item listed in the database. More complex queries can invoke reasoning engines and return different types of data, including new Semantic databases. SPARQL is used by most triple stores and programs designed to work with the Semantic Web.

The SPARQL endpoint for the USGS N3 data can be accessed using any web browser. A link is located on the project web page at *http://cegis.usgs.gov/ontology.html#technology*. The endpoint is hosted using Open Virtuoso. Virtuoso was chosen because it had a proven record of stability and scability as exemplified by its use in DBpedia, having served more than 1 billion triples, as well as its ability to serve as an endpoint and triple store. Its use in this project has resulted in queries on almost 25 million triples returning results almost instantly on most queries. For queries with even moderately large result sets, more time was spent sending the data over the Internet than was spent finding and formatting the results. In instances where extensive reasoning are required, more time would be necessary to construct the result set.

Results and Discussion

The data conversion process moves the data from ESRI database files to GML files, and then from the GML files into N3 formatted RDF files. This process allows for the entirety of the data to be created in GML format and then converted to RDF format along with a copy of the GML for each feature, making it possible to create valid GML using queries on the semantic database. The resulting semantic databases also created linkages between features and entire data sets that were not present in the previous formats. Another outcome of storing the data in RDF format is the ability to add data to existing features or link new features to the data without having to change the structure of the database. New predicate object pairs can be added to a single resource without having to create a new field in all features of the same type as the resource.

The use of the graphical user interface and configurations for features also allows users to customize the process of converting data. Configuration options allow users to choose specific resource namespaces and predicate URIs, thus creating the database structure according to their specifications. The graphical user interface makes it possible for users to choose the data to be converted by specifying the parsers to run; decide which attributes are resources and which are literals through the use of the graphical configuration editor, which allows attributes to be added or removed; edit resource namespaces; edit predicate URIs; and select the attribute to be used as a unique identifier for the feature type.

Conclusions

The objectives of the conversion algorithm were to retrieve data from proprietary formats, to determine a method for representing coordinates, and to be able to expand the range of semantic properties of data by improving the usability of the data for others. The resulting approach maintains the ability to work with data in the original format, but the data can be accessed through open source libraries and programs or recompiled into proprietary GIS formats when necessary. Queries and accessing the database became much simpler, not requiring a username or password to access the server, or as much processing to validate a query. Even if semantic attributes in established GIS and geospatial data technologies are attached to the data files, these properties are not shared easily between data sets. These technical limitations have prevented the easy sharing of attributes between data. Previously accessible data remains available while adding new features easily and securely. Opportunities to enhance the data are increased. The converted data acquired increased data richness; making information available that previously had to be computed. The data allow more complex data classification and relations than are easily possible in traditional databases.

References Cited

Allemang, D. and Hendler, J., 2008, Semantic Web for the Working Ontologist, Effective Modeling in RDFS and OWL: Burlington, Mass., Morgan Kaufman Publishers.

Berners-Lee, T., 2005, Primer: Getting into RDF & Semantic Web using N3 (v. 1.61): World Wide Web Consortium, accessed March 16, 2011, at *http://www.w3.org/2000/10/swap/Primer*.

Bray, T., Paoli, J., Sperberg-McQueen, C.M., Maler, E., and Yergeau, F., 2008, Extensible Markup Language (XML) 1.0 (Fifth Edition): accessed March 16, 2011, at *http://www.w3.org/TR/2008/REC-xml-20081126/*.

Egenhofer, M.J., and Herring, J.R., 1991, Categorizing Binary Topological Relations between Regions, Lines, and Points in Geographic Databases: Technical Report, Department of Surveying Engineering, University of Maine, Orono, Maine, accessed May 6, 2011, at *http://www.spatial.maine.edu/~max/9intReport.pdf*.

ESRI, 2011, Geodatabases. ESRI, accessed March 30, 2011, at *http://resources.arcgis.com/content/geodatabases/10.0/about*.

Geospatial Data Abstraction Library, 2011, GDAL—Geospatial Data Abstraction Library: Open Source Geospatial Foundation, OSGeo Project, accessed March 16, 2011, at *http://www.gdal.org/*.

GeoTools, 2011, GeoTools, The Open Source Java GIS Toolkit: OSGeo Project, accessed March 16, 2011, at *http://www.geotools.org/*.

Hebeler, J., Fisher, M., Blace, R., and Perez-Lopez, A., 2009, Semantic Web Programming: Indianapolis, Ind., Wiley Publishing, Inc., 616 p.

Jena, 2011, Jena—A Semantic Web Framework for Java: Sourceforge, accessed March 16, 2011, at *http://www.openjena.org/*.

Open Geospatial Consortium, 2011, OGC "Making location count," accessed March 16, 2011, at *http://www.opengeospatial.org/*.

OSGeo, 2011, OSGeo, Your Open Source Compass. The Open Source Geospatial Foundation, accessed March 30, 2011, at *http://www.osgeo.org/*.

Portele, C., 2007, OpenGIS Geography Markup Language (GML) Encoding Standard, v. 3.2.1: Open Geospatial Consortium, Inc., OGC 07–036, accessed March 30, 2011, at *http://www.opengeospatial.org/standards/gml*.

Quantum GIS, 2011, Quantum GIS Version 1.6.0.: OSGeo Project, accessed March 30, 2011, at *http://www.qgis.org/*.

Uniform Resource Identifier, accessed March 30, 2011, at *http://tools.ietf.org/html/rfc3986*.

U.S. Geological Survey, 2010, National Hydrography Dataset (NHD) Model (v. 2.0), accessed January 18, 2011, at *http://nationalmap.gov*.

U.S. Geological Survey, 2011a, *The National Map*, accessed January 18, 2011, at *http://nationalmap.gov*.

U.S. Geological Survey, 2011b, Semantic Web Triples—Sample Data from *The National Map*, accessed January 18, 2011, at *http://cegis.usgs.gov/ontology.html*.

Virtuoso Universal Server, 2011, Virtuoso Universal Server. OpenLink Software, accessed March 30, 2011, at *http://virtuoso.openlinksw.com/*.

World Wide Web Consortium (W3C), 2003, Basic Geo (WGS 84 lat/long) Vocabulary. Accessed July 22, 2010, at *http://www.w3.org/2003/01/geo/*.

World Wide Web Consortium (W3C), 2004, RDF Primer, accessed July 14, 2010, at *http://www.w3.org/TR/2004/REC-rdf-primer-20040210/*.

World Wide Web Consortium (W3C), 2008, SPARQL Query Language for RDF, accessed July 14, 2010, at *http://www.w3.org/TR/2011/WD-sparql11-query-20110512/*.

World Wide Web Consortium (W3C), 2009, OWL 2 Web Ontology Language Document Overview. World Wide Web Consortium, accessed March 30, 2011, at *http://www.w3.org/TR/owl2-overview/*.

World Wide Web Consortium (W3C), 2010, Semantic Web, accessed July 14, 2010, at *http://www.w3.org/standards/semanticweb/*.

Publishing support provided by:
Rolla Publishing Service Center

For more information concerning this publication, contact:
Director, USGS Center of Excellence for Geospatial Information Science (CEGIS)
1400 Independence Road
Rolla, MO 65401
(573) 308–3837

Or visit the CEGIS Web site at:
http://cegis.usgs.gov