A Geospatial Data Catalog and Metadata Management Tools for the U.S. Environmental Protection Agency's Western Ecology Division

by

David L. Bradford

A RESEARCH PAPER

submitted to

THE GEOSCIENCES DEPARTMENT

in partial fulfillment of the
requirements for the
degree of

MASTER OF SCIENCE

GEOGRAPHY PROGRAM

November 2007

Committee in Charge:
Dawn Wright, Geosciences, Major Professor
Julia Jones, Geosciences, Committee Member
Jon Kimerling, Geosciences, Committee Member

A Geospatial Data Catalog and Metadata Management Tools for the U.S. Environmental Protection Agency's Western Ecology Division

Abstract

The EPA's Western Ecology Division (WED) had been accumulating geographic information system (GIS) data files for up to 20 years. There was no index or catalog for these files; locating data became more difficult over time. More that 4 Tb of data existed on numerous disk volumes in hundreds of directories. These GIS data files included current and historical WED projects and locally downloaded copies of various national or regional datasets. Metadata was sparse or non-existent. A GIS catalog was implemented for organizing and maintaining these data. The catalog allows advanced search capabilities for the purpose of quickly locating specific GIS data files maintained within the WED. Metadata are a key component of the system; the process of creating metadata has been streamlined using a custom ArcCatalog extension. Once a metadata record is created, it is cataloged by an automated process. The catalog was implemented largely with existing resources and requires low administrative overhead. The components have been implemented, however, widespread testing of the system has yet to occur. Without a continued commitment to the project, institutional inertia may stall it.

Introduction

The U.S. Environmental Protection Agency's Western Ecology Division (WED), in Corvallis, Oregon, had been accumulating geographic information system (GIS) data files for

over 20 years. Until recently, informal data management conventions and practices that were largely conveyed verbally within the close-knit GIS group were sufficient to maintain an implicitly understood process for organizing WED GIS data; thus, no formal index or catalog of these data existed.

However, GIS analysts who were formerly located within the same office area began to relocate to different floors, buildings, and sites. Additionally, as the total amount of GIS data increased, it became necessary to divide the data among several disk volumes due to capacity limits of individual volumes. Hence, due to the decrease in informal communication between analysts and the fragmentation of data storage areas, locating and sharing data had become increasingly difficult. Contributing to this local dilemma was the increase in the downloading of public datasets from external online sources; data has become so abundant that it challenges our ability to manage and efficiently use it (e.g., Goodchild, 2003).

More than 4 Tb of data existed on numerous disk volumes in hundreds of directories. These GIS data files included current and historical WED projects and locally downloaded copies of various national or regional datasets. The creation and maintenance of metadata was never a priority, so metadata were largely non-existent. The few metadata records which did exist did not necessarily follow national or organizational metadata standards. This tends to be a problem for many organizations, as metadata are expensive and time-consuming to create, and users typically perceive that doing so is not worth the effort (e.g., Larson et al., 1998; Goodchild, 2003).

The goal of this project was to implement a GIS catalog and begin the process of organizing and maintaining the WED's GIS data within it. The purpose of the catalog was to allow

advanced search capabilities for quickly locating specific GIS datasets (most importantly, their on-disk locations) among the large amount of GIS data maintained within the WED, including the means to search on numerous attributes such as geographical extent and metadata. However, one important constraint was that many GIS data files were interdependent and contained internal links (absolute paths) to each other. Thus it was necessary that the catalog not require relocating the data, instead providing direct access to them where they existed on disk.

Using the catalog, it was a goal that casual users who were not GIS technicians could find and view the data, and that GIS technicians and analysts could load the data into ArcMap for new analyses and generation of new datasets. As such, ideally the catalog was to be available via both a web-based portal with view capabilities, as well as via an integrated extension for direct access to the data from within the ArcGIS environment.

The benefits expected from such a catalog included the ability to quickly locate needed data, the confidence that the correct data were being used, reduction of redundant downloading of external data, reduction of duplicate data, a more systematic organization and maintenance of WED GIS data assets, improved maintenance of metadata, and an overall efficiency gain in GIS processes and communication.

Background

The project was initiated in June of 2007 as a three month summer internship by the Western Ecology Division (WED) of the United States Environmental Protection Agency (EPA). The WED is a member of the National Health and Environmental Effects Research Laboratory

(NHEERL), which is an organization within the EPA's Office of Research and Development (ORD). The WED is comprised of three branches: the Freshwater Ecology Branch (FEB), Corvallis, Oregon; the Ecological Effects Branch (EEB), Corvallis, Oregon; and the Pacific Coastal Ecology Branch (PCEB), Newport, Oregon.

The members of the project team were a core group who represent the primary GIS analysts employed directly by the WED. Additional project support was provided by the Information Technology Coordinator for the WED as well as local GIS and information technology (IT) contract staff. EPA contractors at the national level also provided vital support in the latter stages of the project.

Metadata

The project team quickly realized that the critical component of any form of GIS catalog, regardless of its eventual architecture, was going to be the consistent creation of meaningful metadata.

Metadata are commonly defined as "data about data" (e.g., Larson et al., 1998; http://jollyroger.science.oregonstate.edu/myst), and are key attributes, usually textual in nature, which describe and summarize an item of data in a useful way. They are used to describe not just geospatial data, but almost any kind of data: social and scientific datasets, enterprise applications, data warehouses, educational resources and bibliographic data. Libraries have been employing metadata standards to describe, manage and index various types of digital resources and objects (Ma, 2007).

But metadata can be expensive to create; the vast majority of metadata must be created manually (e.g., Ma, 2007). It can be very time-consuming to go back and effectively tag legacy data, which may have existed long before metadata standards were implemented, and long after its original purpose has been forgotten (Green and Bossomaier, 2002). In a survey conducted by the Association of Research Libraries regarding metadata, it was found that other challenges include reconciling metadata quality vs. metadata cost, implementing organizational changes, creating the right internal organization for providing metadata services, and developing/accommodating workflow for metadata creation (Ma, 2007). Quite a few respondents urged automating metadata creation as much as possible (Ma, 2007).

One of the best-known standards for geospatial metadata is the Federal Geographic Data Committee's Content Standard for Digital Geospatial Metadata (CSDGM). This standard goes beyond general metadata standards in that it intends to capture additional important properties of geographic datasets, such as spatial resolution, accuracy, projection and datum (Longley et al., 2005). But creating CSDGM-compliant metadata can be a skilled job. It requires not only a full understanding of the data itself, but also a detailed understanding of the content standard. Without the right tools, it can involve significant costs, in terms of personnel, to create and update the information (Green and Bossomaier, 2002).

Yet accurate metadata remain the key to finding and re-using GIS datasets. Solutions for searching for geospatial data include the promotion of metadata standards that make the datasets self-indexing (Green and Bossomaier, 2002). We also need metadata to be able to judge the fitness of the dataset for a particular use, and to know how to handle the dataset effectively (e.g., where it is located on the network, its filesize, and the software required to open it; Longley et al, 2005). Thus it was recognized that a core requirement of the project

would be to provide a means of creating metadata, automating the process as much as possible, and insuring that the metadata met the necessary standards.

Existing EPA Process

WED projects are initiated by principal investigators (PIs). These projects can require new GIS analyses and result in creation of new GIS data. During the development and execution of their projects, the PI will issue GIS service requests to GIS analysts and GIS contractors. GIS analysts may perform the work directly, or divide the work and delegate some of it to the EPA contractors.

In a large organization such as this, several projects can be running in parallel at any given time, and the potentially singular nature of the projects can often lead to duplication, as each project evolves using different data, people and procedures (Longley et al., 2005). Sharing data and experience is usually seen to be a low priority (Longley et al., 2005).

In this fashion, the WED has been accumulating GIS data files for up to 20 years. These files were stored on numerous disk volumes in hundreds of folders/subdirectories. These GIS data files represented current and historical WED projects, locally downloaded copies of various national or regional datasets, and locally shared community files. An unknown number of these files were duplicates or redundant. For many of the files, metadata records were non-existent.

Goodchild (2003) indicates that, in the absence of catalogs containing general descriptions of archive contents, searches must too often rely on personal knowledge, personal contacts, and

time-consuming trial and error. Indeed, as there was no index or catalog for these files, it was found at the WED that in order to locate a specific GIS data file, one was required to:

a) have familiarity with the current or historical use and organization of the GIS disk volumes; or
b) execute a time-consuming search through all of the folders/directories using either an ArcCatalog search or a basic Windows file search; or
c) ask someone else who might know where the data are located; or
d) download a national or regional dataset from elsewhere (e.g., publicly available on the Internet or from an offline CD/DVD library – with the potential to create a duplicate copy of something which may already exist locally but could not be located).

Several commonly-used national or regional datasets were stored either on the disk volumes or on offline media and were used for multiple projects. These included digital raster graphics (DRGs) for the entire country, base coverage files for regions of interest, National Land Cover Data (NLCD), the National Hydrography Dataset Plus (NHDPlus), and the National Elevation Dataset (NED).

Existing Infrastructure

Hardware, Network, & Operating Environment

The WED local area network consisted of numerous Windows NT-based servers performing various functions, including one operating as network file server for storage of GIS datasets (approximately 2.5 Tb of files). Additionally, an older UNIX-based RAID file server was

employed to store GIS data (approximately 1.5 Tb). GIS analysts and users also made extensive use of disk space on their local PCs to store "in-process" GIS data.

A Windows NT-based web server running Internet Information Services operated as the WED's local web server on the nationwide EPA intranet. The web server was utilized by this project to provide a "web accessible folder" (WAF) as an intermediate storage area for metadata harvesting (described in detail in the **Results** section).

Software Environment

The Environmental Systems Research Institute (ESRI) ArcGIS suite of products was utilized extensively by the EPA and constituted their primary GIS platform. During the project, the WED was in the process of transitioning from ArcGIS versiont 9.1 to version 9.2. Other pre-existing applications which became key components in the project's solution were:

EPA Metadata Editor – a custom extension or "plug-in" for ArcCatalog, developed by the EPA's GeoData Gateway group, which allows users to create and edit geospatial metadata records that meet the EPA Metadata Technical Specification v1.0 (an FGDC-compliant specification with additional EPA-specific requirements).

Second Copy – a Windows utility which allows the duplication of selected files and their directory structures from one disk storage location to another; in this project, the utility was used to copy newly created or modified metadata files from their native storage locations to the WAF, maintaining an equivalent folder hierarchy.

EPA GeoData Gateway – a geospatial portal based on ESRI's GIS Portal Toolkit that serves as a national access point for EPA geospatial assets (data, metadata, services, applications, other resources). For the purposes of this project, the GeoData Gateway was utilized as a catalog for indexing and searching the WED GIS datasets.

Microsoft's Visual Studio 2005 – an integrated software development environment (IDE), utilized as the development tool for a key custom component of the project solution: the Visual Basic language and ESRI's ArcObjects library were used to develop the EPA Synchronizer and the EPA Batch Synchronizer extensions for ArcCatalog (see **Results**).

Organizational Parameters and Constraints

Budget

The budget for the project consisted of funds for the summer employment of one GIS software developer. No additional budget was formally allocated for hardware, software or services, and none was required.

Staff

In addition to the software developer, the other core members of the project team were four GIS analysts employed by the EPA's Western Ecology Division, who provided the majority of input regarding project requirements and direction throughout the project.

Additional project team members at points during the project were the EPA's GeoData Gateway support staff, located in Research Triangle Park, North Carolina; the local IT Director for the WED; and the EPA's Office of Research and Development (ORD) computer technical support staff.

Users

Within the WED, there were fourteen GIS analysts, approximately half of them contractor staff, who were responsible for creating and modifying GIS data. These personnel required read and write access to most GIS datasets. There were roughly fifty other individuals (principal investigators and science support staff) who might view and utilize the GIS data produced. These GIS users required only read access to data files.

The majority of GIS analysts and users were located at the WED's Corvallis, Oregon campus; one additional analyst and some users were located at the Hatfield Marine Science Center in Newport, Oregon.

Data

The data consisted almost exclusively of the ESRI ArcGIS formats of coverages and shapefiles. Approximately 4 Tb of data existed on six separate disk volumes.

Standards and Policies

GIS data files published for broader use within the EPA and by the public are required meet certain Federal and EPA-specific metadata standards:

Federal Geographic Data Committee - Content Standard for Digital Geospatial Metadata (FGDC-CSDGM) – The CSDGM, Version 2 (FGDC-STD-001-1998) is the US federal metadata standard (FGDC, 1998). The Federal Geographic Data Committee originally adopted the CSDGM in 1994 and revised it in 1998. According to Executive Order 12096 all Federal agencies are ordered to use this standard to document geospatial data

created as of January, 1995. The standard is often referred to as the FGDC Metadata Standard and has been implemented beyond the federal level with State and local governments adopting the metadata standard as well.

National Geospatial Data Policy (NGDP) – Institutional arrangements underlie the availability of an organization's data (e.g., Goodchild, 2003). The ability of federal agencies to supply the rapidly increasing demand for geographic data has been severely curtailed by budget reductions. In response to this, the National Research Council proposed the National Spatial Data Infrastructure (NSDI). The NGDP represents the EPA's commitment to data sharing, promoting secondary data use, and supporting the NSDI (Goodchild, 2003). It establishes principles, responsibilities, and requirements for collecting and managing geospatial data used by Federal environmental programs and projects within the jurisdiction of the EPA (EPA, 2005).

Geospatial Metadata Technical Specification Version 1.0 (EPA internal document) – a standard for publishing geospatial metadata for data sets, applications, and services developed by the EPA (EPA internal document). This standard is a more stringent extension of the FGDC-CSDGM. The installation and proper use of the EPA Metadata Editor extension for ArcCatalog insures that all of the required EPA standards are met.

GeoData Gateway Governance Structure Report (EPA internal document) – the EPA's GeoData Gateway served as the catalog for the project; this document identifies the oversight and authority organization of the GeoData Gateway (GDG) and provides an overview of roles and responsibilities for those who contribute to the GDG. Roles and responsibilities outlined within this document align upwardly with the NGDP.

Additional Constraints

A significant percentage of the GIS data files stored on WED systems were dependent upon other GIS datasets and contain internal links (absolute paths) to them. As such, in order for ArcMap to properly display these GIS files, their dependencies could not be relocated. It was thus decided that the catalog should reference their existing physical locations rather than moving them onto a dedicated catalog server.

.

Methods

It was known that the solution ultimately developed for the project was to be an application system, i.e., primarily a software-based solution with an integrated set of support and service components. As such, a traditional system development life cycle was applied throughout the project. Following the identification of a need for a new system, the steps in this method are a requirements analysis, an architectural design or "blueprint" for the new system, software development, integrated system testing, and implementation.

Requirements Analysis

At the outset of the project, the primary objective was generally stated to be the development of a catalog, or *geolibrary*, which lists the WED's GIS data holdings and their physical locations. The term geolibrary was coined to describe digital libraries that can be searched for information about any user-defined geographic location (e.g., Longley et al., 2005).

During this phase, it is necessary to more specifically identify the types of functions that the system should perform, as well as the organizational parameters and constraints, such as

budget, project team members and skills, number of users, amount and type of data, available hardware and software, organizational standards/policies/procedures, and technical or institutional constraints. Additionally, the desired user interface is outlined.

Architectural Design

During the architectural design, the project team examined their options for implementing various components of the system, weighing the results of the analysis and selecting the most feasible strategy for a solution. Having generally identified the desired process for getting GIS datasets cataloged, a plan was established for coding the missing software components as well as developing and enhancing existing resources.

Software Development

Once requirements were fully defined, the project team determined the scope and nature of the technical components necessary to accomplish the project goals. In this case, a software application of some sort was to be necessary, as well as additional software components and background administrative processes to tie the system together.

Software components and processes newly developed must each be tested individually and in isolation to insure that, given the full range of expected inputs, they are producing the proper outputs. This type of testing is known in the software development world as “unit testing” (IEEE, 1999). Such testing is usually done by developers and not end users; as bugs are found in the individual components, they are fixed and re-tested.

Integrated System Testing

Once components were developed and successfully unit-tested, they are implemented together in a test environment and tested from end-to-end. This type of testing is known as "integration testing" (ISTQB, 2006). Select end-users, typically people involved in the project from its outset, operate the system and identify any problem areas encountered, while developers and technical support staff work to fix the problems.

User Training and Implementation

After the system has been fully tested and deemed to be functioning adequately, it is "rolled out," that is, made generally available for organizational use. Depending on the complexity of the application, end-user documentation and/or interactive user training may be required.

Results

Requirements Analysis

The results obtained during the requirements analysis phase largely describe the structure of the environment in which the project was conducted, and thus most appear in the Background section of this paper under ***Organizational Parameters and Constraints***.

As identified by the core project team, additiontional details regarding user interface requirements appear here.

User Interfaces

ArcGIS-Integrated

Ideally, GIS analysts and contractors at the WED preferred to utilize the capabilities within ArcGIS applications (ArcExplorer and ArcMap) to connect directly to publicly available GIS data services via the Internet. ArcCatalog's existing search function can connect to GIS data services and perform searches based on metadata attributes, geographical location, data type, and date.

Web Portal

For other users who did not use ArcGIS on a frequent basis, or who desired a quicker, more direct means of finding GIS data, a web-based interface to the GIS catalog (i.e., a "web portal") was preferable. This would implement the same search functionality as in ArcCatalog, but would instead do so via web forms. Ideally, the web portal was to include the ability to view the data from within the web browser.

Architectural Design

During this phase, it was quickly determined that the starting point for any form of solution would require that GIS analysts create meaningful metadata. Accurate metadata is the key to finding and re-using GIS datasets. Further, if the contents of the GIS Catalog were to be made available outside of the WED, they would need to meet EPA metadata standards (and by extension, FGDC); that is, they would need to meet the EPA Geospatial Metadata Technical Specification Version 1.0. Thus, valid metadata would also be necessary for this

system to work. The full end-to-end process developed during architectural design is as follows:

1. Create metadata for GIS datasets to be cataloged.

Metadata files are created and updated using ArcCatalog and two custom EPA tools designed for ArcCatalog: the EPA Synchronizer, and the EPA Metadata Editor.

In ArcCatalog, when the "Create/Update Metadata" button is pushed, a metadata file for the selected dataset is created in "XML" format. By default, ArcCatalog creates a metadata file that adheres to FGDC, ISO, and Geography Network metadata standards. What was needed was to make it meet the more stringent EPA standard, as well as contain certain EPA default values.

Fortunately, the code behind ArcCatalog's "Create/Update Metadata" button includes a "hook," or customizable subroutine, that allows the software developer to insert custom code. This code enables one to add or change any metadata values which are being written by ArcCatalog. Thus, the EPA Synchronizer, the primary piece of software developed during this project, allows ArcCatalog to create largely EPA-compliant metadata with the push of a button. Only the **Title**, **Abstract**, **Purpose**, and **Supplemental Info** fields are left blank, as they obviously require human intervention.

The EPA Metadata Editor (EME) is a custom editing plug-in for ArcCatalog, developed by the EPA's GeoData Gateway group. There are two main advantages of the EME: 1) it includes a customizable Microsoft Access database which allows the user to specify default values for almost any metadata element (e.g., owner, keywords, data quality); and 2) it

validates the metadata to the EPA standard; that is, before saving the metadata record, the user can press a button and receive a report indicating whether it is fully EPA-compliant or not.

However, without the EPA Synchronizer, the EME was of doubtful value to the project team; creating metadata using it alone was a cumbersome and time-consuming process. It was not written to extract information directly from the dataset itself, and users were still required to manually enter such things as geographic extent, datum, projection, and on-disk file location. Thus, the EPA Synchronizer, which can write internal dataset values to the metadata as well as default values obtained from the EME's Access database, is the perfect compliment – the "missing link"– that makes creating EPA-compliant metadata as fast and painless as possible.

2. The metadata are copied to harvest folder.

Creating valid metadata is the only step that the GIS analyst is required to do to add a dataset to the catalog. The rest of the process is automated. Once the metadata file exists, it is automatically copied by a server process which looks for new or updated metadata files. This step is informally called the "internal harvest" and works as follows:

On a weekly basis, a file copying utility called Second Copy runs on one of the WED's local servers and does a comprehensive search for all new and changed metadata files within the internal file servers hosting the WED's GIS datasets. This utility then copies all of these new and changed metadata files to a folder on the WED's intranet web server. This folder is visible to the rest the national EPA intranet, but not to the public, and is formally called a

"Web Accessible Folder" or WAF. Second Copy places the files in a folder hierarchy identical to the internal folder hierarchy of the original GIS datasets.

Control over which GIS datasets' metadata get published in the GeoData Gateway is maintained by taking care in specifying which of the WED's GIS folders are included in the internal harvest process. At the WED, certain folders on its file servers are considered to contain "common" data or "base" datasets which:

- do not change once placed there;
- may be directly linked/referenced within other, project-oriented datasets;
- are accessible to all GIS analysts and users at the WED; and
- may be local downloaded copies of commonly-used national GIS datasets.

All such GIS common folders should ideally be configured for internal harvest. Other folders which contain "finished" project data, or data which analysts and principle investigators might wish to share EPA-wide, should also be configured for internal harvest.

3. Metadata are harvested weekly into the GeoData Gateway catalog.

The EPA GeoData Gateway (GDG) is an ESRI GIS Portal Toolkit server created and maintained by the EPA's Office of Environmental Information. It functions as both an ArcIMS server accessible through ArcCatalog and a web server accessible via a standard web browser. It serves as an access point for EPA geospatial assets (data, metadata, services, applications, other resources). For the purposes of this project, the GDG was chosen to serve as the catalog for indexing and searching for the WED GIS datasets. The primary reason for

using the GDG is that the system is managed centrally by national EPA staff, requiring no local expertise or system support resources.

The person at the WED responsible for managing its GDG holdings (the "GDG Steward") has an administrative account on the GDG server through which he or she can configure the automatic harvest of the WED's metadata. This configuration simply consists of typing in the web address for the WED's WAF, and indicating how frequently to check the WAF for new metadata.

The GDG then automatically searches through the WED's WAF periodically, and imports ("harvests") any new or changed metadata it finds, updating the GDG database with the new information. However, before the metadata are posted and searchable, they must pass an automatic validation test and it must be approved by the GeoData Gateway Administrator (the EPA's national geospatial data steward). Through the administrative account, the WED's GDG Steward can also review metadata harvest reports and check for metadata records that did not validate.

4. Users can search the GeoData Gateway via ArcCatalog or via the GeoData Gateway web interface.

At this stage, the GeoData Gateway now contains all valid and approved metadata that analysts at the WED have chosen to catalog. These metadata are full-text searchable on any metadata element value, as well as searchable using the dataset's geographic extent. The user can search for datasets from within ArcCatalog, or they can search for datasets using the GDG web interface. The user can then review the matching metadata records found, and identify the dataset's location on a local file server. Since the GDG contains only the

metadata for a dataset and not the actual dataset itself, an extra manual step is required to load the dataset into ArcMap.

Some of the advantages of internet technology portals for digital catalogs are the relative ease with which they can be set up, and the convenience for users to preview summaries or simplified versions data to determine its suitability for a particular purpose (e.g., Longley et al., 2005). In this sense, the Internet simplifies the problem rather than adding value (Longley et al., 2005).

Further, making local datasets available in a centralized, online catalog can help eliminate duplication of effort. Additionally, knowing an externally developed dataset is already available locally can help organizations avoid redundant downloading of the dataset from elsewhere, saving time, network bandwidth and local disk space (Green and Bossomaier, 2002).

Software Development

The only custom software developed for the project were the EPA Synchronizer, the hook which creates a metadata file for a single dataset, and the EPA Batch Synchronizer, a custom ArcCatalog toolbar tool which simply executes the EPA Synchronizer program on multiple selected datasets.

The process of creating and updating metadata using values obtained from the dataset or from other sources is called *synchronization*. For the EPA's use, a custom synchronizer, called the EPA Synchronizer, was developed to make sure the metadata is compliant with the

EPA's metadata technical specifications. It is based on concepts and sample code originally developed by ESRI (ESRI 2002, ESRI 2006).

The EPA Synchronizer (Appendix A) was programmed in Visual Basic (VB) using the Microsoft Visual Studio 2005 IDE, and utilized ESRI ArcObjects libraries. These libraries enable full programmatic access to ArcGIS program and data classes. Using ArcObjects classes, VB code was written which directly queried the selected GIS dataset, retrieved internal values such as datum, projection, geographic extent, and physical file location, and wrote these values into the appropriate metadata elements. Additionally, the code fleshed out the rest of the metadata by opening the EME's Microsoft Access database, retrieving the user-specified default values, and writing them into their corresponding metadata elements.

The EPA Synchronizer, when compiled, exists as a "dynamic link library" or DLL file. Using ESRI-supplied utilities, this DLL is installed as a new custom component of the user's ArcGIS installation. Thereafter, when the "Create/Update Metadata" button is pushed, ArcCatalog will execute the EPA Synchronizer code when creating (or updating) a metadata file for the selected dataset.

By default, ArcCatalog is installed with three active metadata synchronizers which automatically create metadata for selected GIS datasets. These are the FGDC Synchronizer, the Geography Network (GN) Synchronizer, and the ISO Synchronizer. However, the EPA Synchronizer setup program was configured to deactivate the ESRI installed synchronizers, leaving the newly installed EPA synchronizer as the only active synchronizer. This is to insure that the other synchronizers do not write metadata values that are invalid according to the EPA metadata specification.

To facilitate the creation of metadata for multiple, related GIS datasets that should contain largely similar (or identical) metadata, an additional tool was created called the EPA Batch Synchronizer. This tool allows one to select multiple GIS datasets in ArcCatalog, and specify the same **Title**, **Abstract**, **Purpose** and **Supplemental Info** metadata fields for all of them. It also runs the EPA Synchronizer on all of the selected datasets, updating the remainder of their metadata from the defaults database and using internal dataset values.

Integrated System Testing

Funding for the project expired shortly after the software development phase. As of this writing, integrated system testing had barely begun. It appears that a renewed institutional commitment and devotion of time and effort by the remaining project team will be required to complete this phase.

A similar GIS data management system was developed and described by Vert et al. (2002). Regarding the integrated system testing phase, they eloquently state:

> Finally, in order for this prototype, thus architecture, to be turned into a production-grade design, the prototype needs to be stress-tested in a larger user environment. The prototype implementing the architecture described here was only intended as an initial proof of concept for theories developed in our previous work. What is needed next is to have a larger test population attempt to utilize the GIS Workbench for doing their work. This would help identify any

> flaws in the concepts and in the resultant architecture, and would lead to its further development for managing GIS data files.

Thus is clearly stated the goal of the integrated system testing phase. During this phase, the plan was to identify several of the major, commonly used national and regional datasets, and start the process of creating metadata for them. As metadata would be created, the automated processes for harvesting them would be triggered, thus enabling the full system test.

User Training and Implementation

A comprehensive draft of instructional user documentation for the prototype system was completed. As the primary user activity was to be simply metadata creation and catalog searching, it focused primarily on those activities. Additionally, technical instructions were written for computer support personnel which detailed the installation and configuration of the software tools, harvesting processes, and GDG administration.

As of this writing, Implementation has not yet occurred. The plan for implementation was not to catalog the WED's entire GIS holdings; rather, it was to have the project team, GIS analysts or support personnel identify all of the existing datasets which should appear in the catalog. Metadata would then be created for these, thus populating the catalog. From implementation date forward, metadata was also to be created for any new datasets to be cataloged.

Discussion

This project presented some interesting challenges. The goal was a major one: to design a system which could catalog a large amount of GIS data (having little pre-existing metadata); do so with a minimum of financial and human resources; design a system which, once operational, require a minimum of resources to maintain; and implement a solution within 14 weeks.

Fortunately, there were a number of existing resources which provided significant support for the project. Many GIS projects use the services of external consultants (Longley et al., 2005). This project benefitted from the availability of the consultants on the GeoData Gateway support team, who brainstormed technical solutions and assisted with unit testing of the GDG.

In the end, a simple form of federated approach was employed. Such an approach involves autonomy in data maintenance, non-intrusive data access, and user-transparent query capability (Evans and Ferreira, 1995). The solution avoided the need to relocate data; a characteristic of the federated approach is that it allows data dissemination to occur with minimal changes in the way an organization represents, structures, and stores its own data (Evans and Ferreira, 1995).

In spite of the positive results of this minimalist solution thus far, such a project would have had a greater chance for success with a high-level champion in the organization, a willingness to overcome institutional inertia, and an additional commitment of staff. Performing GIS service and support tasks requires both analyst-level GIS skills and

administrative skills, and in GIS operations with five or more staff, it is justifiable to identify someone who can fulfill what can become a core role (Longley et al., 2005). GIS support staff need to be able to support several GIS functions, including collection development, user training, data and metadata acquisition, database management, hardware and software maintenance, and cataloging of data and metadata (Longstreth, 1995).

Additionally, local application development support can provide significant benefits. As demonstrated by this project, sources of application development work include improvements/enhancements to existing applications. Ideally, application developers should be assigned full-time to a project and should become permanent members of the GIS group to insure continuity and cohesive involvement (but often this does not occur; Longley et al., 2005).

In projects concerned with sharing geographic information, the personalities of key staff can be important. The project manager must perceive herself as a champion of the project – and act as one – in order to maintain a high level of support for the project (Obermeyer, 1995).

Spatial information sharing is often obstructed by "organizational" issues such as turf battles, the need for reorganization, or institutional inertia (Evans and Ferreira, 1995). In many organizations attempting to adopt dynamic GIS services, institutional barriers must still be overcome (Tsou and Buttenfield, 2002). Indeed, during this project, tension developed between the GIS contract staff and the project team over the perception that the project was going to negatively impact the contractors' autonomy in managing their data. Additionally, there was a level of red tape (which seems to exists only in government agencies) which slowed project progress significantly at times. Lastly, quality control as it pertains to digital

information appeared to be a low priority; it was recognized as needed, but there were no mechanisms for its enforcement.

Conclusion

Data used in a shared environment become cleaner – more complete and correct (Craig, 1995). As a beneficial side effect of this project, it is anticipated that new datasets will be developed with the understanding that they will be shared, and as a result, closer attention will be paid to insure their "completeness" and "correctness." Useful legacy datasets will receive new metadata, identifying them as valuable and allowing users to quickly locate them in the catalog.

It is hoped that with the importance of metadata records to this catalog (they are the linchpin), creating meaningful metadata will become an organizational priority for the WED. However, when integrated system testing resumes, there will undoubtedly be some unseen hurdles to overcome in achieving the full end-to-end catalog solution. Without a champion, and without institutional commitment to see the project through, there may not be enough momentum to overcome these hurdles. Indeed, Evans and Ferreira (1995) note that research of such barriers to data sharing was needed to address the "messy" transition period when organizations aren't fully equipped or conforming to the new standards and theories for spatial data sharing (i.e., still struggling with institutional inertia).

Such institutional issues are recognized in the research community as a factor in the success of new technology. The advance of research in technology use follows a pattern that begins with technological feasibility, proceeds through financial affordability, continues with

institutional issues and culminates with societal effects (Obermeyer and Pinto, 1994; Figure 1). GIS technology has matured over the last 25 years and costs have dropped to the point where most organizations can utilize some form of GIS. Institutional and societal issues are now at the fore (Obermeyer and Pinto, 1994).

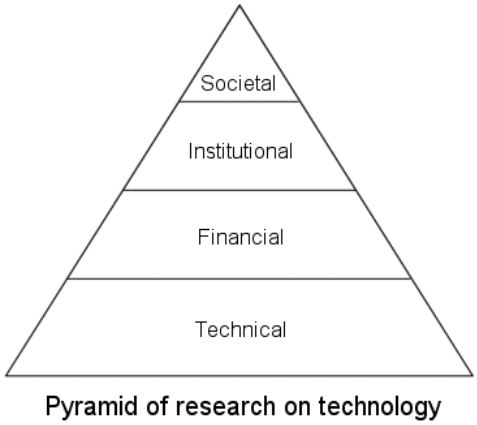

Figure 1 - Advancing levels of research in technology (Obermeyer and Pinto, 1994).

The progress seen in this project was encouraging. The EPA Synchronizer has proven to be of great use not only to the WED; the EPA's GeoData Gateway team has plans to bundle the utility with its next release of the EPA Metadata Editor. At the WED, with the GIS catalog on the verge of success, it will be interesting to see whether it has the momentum to overcome any remaining obstacles, and break free of the chains of institutional inertia.

LITERATURE CITED

Craig, William J. (1995). Why We Can't Share Data: Institutional Inertia. In: Onsrud, H.J. and G. Rushton (Eds.) *Sharing Geographic Information*. Rutgers University & the Center for Urban Policy Research, New Brunswick, New Jersey: 107-118.

EPA (2005). National Geospatial Data Policy, Environmental Protection Agency. http://www.epa.gov/nerlesd1/gqc/pdf/epa_natl_geo_data_policy.pdf, last accessed September 15, 2007.

ESRI (2002). *Creating a Custom Metadata Synchronizer*, An ESRI White Paper. July 2002. ESRI, Redlands, CA. http://www.esri.com, last accessed November 26, 2007.

ESRI (2006). Sample Metadata Synchronizer. Visual Basic source code developed by Jonathan Makin, ESRI(UK) Limited. Original coding 15th November 2000. http://www.esri.com, last accessed November 26, 2007.

Evans, John and J. Ferreira Jr. (1995). Sharing Spatial Information in an Imperfect World: Interactions Between Technical and Organizational Issues. In: Onsrud, H.J. and G. Rushton (Eds.) *Sharing Geographic Information*. Rutgers University, Center for Urban Policy Research, New Brunswick, New Jersey: 448-460a.

FGDC (1998). FGDC-STD-001-1998, Content Standard for Digital Geospatial Metadata, Federal Geographic Data Committee, June 1998.

Goodchild, Michael F. (2003). Geographic Information Science and Systems for Environmental Management. *Annual Review of Environment & Resources*, vol. 28: 493-519.

Green, David and T. Bossomaier (2002). Online GIS and Spatial Metadata. Taylor & Francis, London; New York.

IEEE Standards Board (1999) IEEE Standard for Software Unit Testing: An American National Standard, ANSI/IEEE Std 1008-1987, in *IEEE Standards: Software Engineering, Volume Two: Process Standards; 1999 Edition; published by The Institute of Electrical and Electronics Engineers, Inc.* Software Engineering Technical Committee of the IEEE Computer Society.

ISTQB (2006). Standard glossary of terms used in Software Testing Version 1.2 (June 4, 2006). Ed. van Veenendaal, Erik. Glossary Working Party, International Software Testing Qualification Board. http://www.astqb.org/documents/ISTQBGlossaryofTestingTerms1.2final.pdf, last accessed Nov 27, 2007.

Larson, K., G. Burton, P. Scarrah, and B. Snyder (1998). Don't duck metadata, *Surveying and Land Information Systems*, Vol. *59*: 169-173.

Longley, Paul A., M.F. Goodchild, D.J. Maguire, and D.W. Rhind (2005). *Geographic Information Systems and Science, 2nd Ed.* John Wiley & Sons, Ltd, Chichester, West Sussex, England.

Longstreth, Karl (1995). GIS Collection Development, Staffing, And Training. *Journal of Academic Librarianship*, vol. 21 no. 4: 267-275.

Ma, Jin (2007). *SPEC Kit 298: Metadata*. Association of Research Libraries, Washington, DC.

Obermeyer, Nancy J. (1995). Reducing Inter-Organizational Conflict To Facilitate Sharing Geographic Information. In: Onsrud, H.J. and G. Rushton (Eds.) *Sharing Geographic Information*. Rutgers University, Center for Urban Policy Research, New Brunswick, New Jersey: 138-148.

Obermeyer, Nancy J. and J.K. Pinto (1994). *Managing Geographic Information Systems*. The Guilford Press, New York.

Tsou, Ming-Hsiang and B.P. Buttenfield (2002). A Dynamic Architecture for Distributing Geographic Information Services. *Transactions in GIS*, Vol. 6, No. 4: 355-381.

Vert, Gregory, M. Stock, P. Jankowski and P. Gessler (2002). An Architecture for the Management of GIS Data Files. *Transactions in GIS*, Vol. 6, No. 3: 259-275.

Appendix A

Visual Basic Source Code for EPA Synchronizer.

```
Option Strict Off
Option Explicit On
Imports System.Globalization
Imports System.Runtime.InteropServices
Imports ESRI.ArcGIS.ADF.CATIDs
Imports ESRI.ArcGIS.Geodatabase

<System.Runtime.InteropServices.ProgId("EPAWEDSynchronizer_NET.EPAWEDSynchronizer")> Public Class EPAWEDSynchronizer
    Implements IMetadataSynchronizer

    ' Copyright 2006 ESRI
    '
    ' All rights reserved under the copyright laws of the United States
    ' and applicable international laws, treaties, and conventions.
    '
    ' You may freely redistribute and use this sample code, with or
    ' without modification, provided you include the original copyright
    ' notice and use restrictions.
    '
    ' See use restrictions at /arcgis/developerkit/userestrictions.

    ' Illustrates use of the IMetadata Synchronizer interface.

    ' Upon sychronization, several key elements in the metadata's
    ' Distribution Information section are set. Distribution Information
    ' for both ISO and FGDC are handled.

    ' Jonathan Makin, ESRI(UK) Limited.
    ' Original coding 15th November 2000

    ' Edited by John Banning, ESRI Redlands
    '
    ' September, 28, 2007:
    ' Modified extensively for EPA purposes by:
    '       David L. Bradford
    '       Environmental Protection Agency
    '       ORD-NHEERL-Western Ecology Division
    '       200 SW 35th Street
    '       Corvallis, OR 97333
    '       Permanent email: bradford@lifetime.oregonstate.edu

    ' The ESRI White Paper "Creating a Custom Metadata Synchronizer" provides
    ' helpful (but outdated) information on some of the techniques used in
    ' this module.

    Public Const MAXKEYS As Integer = 100
    Dim fs As Object
    <ComRegisterFunction()> _
    Public Shared Sub Reg(ByVal regKey As [String])
```

```
        MetadataSynchronizers.Register(regKey)
    End Sub

    <ComUnregisterFunction()> _
    Public Shared Sub Unreg(ByVal regKey As [String])
        MetadataSynchronizers.Unregister(regKey)
    End Sub

    Private ReadOnly Property IMetadataSynchronizer_ClassID() As ESRI.ArcGIS.esriSystem.UID Implements IMetadataSynchronizer.ClassID
        Get
            ' This property is used to uniquely identify the synchronizer.  The easiest way to do
            ' this is to use the UID of this VB Class module, since we know that this will be unique
            ' to the application.

            Dim myUID As New ESRI.ArcGIS.esriSystem.UID
            myUID.Value = "EPACustomSync.EPAWEDSynchronizer"
            IMetadataSynchronizer_ClassID = myUID.Value

        End Get
    End Property

    Private ReadOnly Property IMetadataSynchronizer_Name() As String Implements IMetadataSynchronizer.Name
        Get
            ' Set the name to a string which describes the metadata synchronizer.

            IMetadataSynchronizer_Name = "EPACustomSync"

        End Get
    End Property

    Private Sub IMetadataSynchronizer_Update(ByVal pPropertySet As IXmlPropertySet, ByVal itemDesc As String, ByVal Value As Object) Implements
IMetadataSynchronizer.Update
        ' This method does the work of writing/rewriting the metadata.  Each time the metadata is
        ' synchronized, ArcCatalog calls this method and passes itemDesc and Value to it. There are
        ' multiple calls to this method per synch, depending on how many itemDesc/Value pairs need to be
        ' processed. itemDesc is always a string; Value can be any object
        '
        ' Each itemDesc/Value is either a property of the dataset (e.g. the dataset's spatial
        ' reference system) or a property of the computing environment (e.g., operating system or
        ' software version). See the white paper, Table 1, for a list of possible itemDesc strings and
        ' their corresponding Value object types
        '
        ' itemDesc is tested using a Case statement to determine which type of item is being processed.
        ' A separate Case statement handles the processing of each item type by querying the
        ' Value object and extracting the appropriate values to be written into the metadata file.

        Dim onlink1 As String           'will hold primary linkage value
        Dim onlink2 As String           'will hold secondary linkage alue

        Dim metd As DateTime            'metadata date object
        Dim metfrd As DateTime          'metadata future review date object
        Dim smetd As String             'string version of metadata date
        Dim smetfrd As String           'string version of metadata future review date
        Dim cultinfo As CultureInfo     'language/culture object needed to properly parse string dates

        Dim spubdate As String          'date when the data set is published or made available for release
        Dim sprocdate As String         'processing step date
```

```
' Create a simple property set by QI the XML property set, which we can use for GetProperty tests
Dim pPropSet As ESRI.ArcGIS.esriSystem.IPropertySet
pPropSet = pPropertySet

Dim BracketPos As Long
Dim tmp_itemDesc As String
Dim NumPlus As String
Dim itemCount As Long

'Some itemDescs have bracketed counts, e.g. Entity[0]. Strip the count off into a variable
itemCount = 0

BracketPos = InStr(1, itemDesc, "[")

If BracketPos > 0 Then
    NumPlus = Right(itemDesc, Len(itemDesc) - BracketPos)
    itemCount = CLng(Left(NumPlus, Len(NumPlus) - 1)) + 1
    tmp_itemDesc = Left(itemDesc, BracketPos - 1)
Else
    tmp_itemDesc = itemDesc
End If

itemDesc = tmp_itemDesc

'Default values for metadata are obtained from the EPA Metadata Editor (EME)
'Access database. Values used are (table: field[s]):
'
'1a_Citation: origin
'1b_Publisher: pubplace, publish
'1c_OnlineLinkage: primary_onlink, secondary_onlink
'1e_Currentness: current
'1f_Update: update
'1h_KeywordsEPA: themekt, themekey
'1i_KeywordsISO: themekt, themekey
'1j_KeywordsPlace: placekt, placekey
'1k_Constraints: accconst, useconst, secsys, secclass, sechandl
'2a_Completeness: complete
'3a_ResourceType: resdesc
'3b_DistributionLiability: distliab
'3c_MetadataStandard: metstdn, metstdv

'Check for metadata database
Dim MetadataMdbFile As String
fs = CreateObject("Scripting.FileSystemObject")
MetadataMdbFile = "C:\Program Files\Innovate! Inc\EPA Metadata Editor\metadata.mdb"
If Not fs.FileExists(MetadataMdbFile) Then
    MetadataMdbFile = "C:\Program Files\Innovate! Inc\EPA Metadata Editor v2\metadata.mdb"
    If Not fs.FileExists(MetadataMdbFile) Then
        MsgBox("Can't find metadata defaults database." & Chr(13) & "Some metadata values may not be created/updated.")
        'Exit Sub
    End If
End If

'Main statement for executing synchronizer code - itemDesc is the "section" of metadata being synchronizers
Select Case itemDesc
```

```
            Case "Boilerplate"
                ' "Value" variable type: Nothing
                '
                ' Boilerplate is a special case that is called only the first time synchronization occurs.
                ' This may be used to add boilerplate text, such as documentation hints or fixed contact
                ' information for your organization, that should not be changed again by synchronization.
                ' If there are things in here that could be changed by synchronization, you could move the
                ' code down to another section that always gets called, e.g. "DatasetName"

                Dim htCitation As Hashtable = New Hashtable

                'initialize a value in case GetDefaultRecord fails
                htCitation!origin = "US EPA Western Ecology Division"

                'get the Citation record from the metadata database
                GetDefaultRecord(htCitation, MetadataMdbFile, "1a_Citation")

                'set the metadata element to the obtained record value(s)
                pPropertySet.SetPropertyX("idinfo/citation/citeinfo/origin", htCitation!origin, esriXmlPropertyType.esriXPTText,
esriXmlSetPropertyAction.esriXSPAAddOrReplace, False)

                Dim htPublisher As Hashtable = New Hashtable

                'initialize a value in case GetDefaultRecord fails
                htPublisher!publish = "U.S. EPA Office of Research & Development (ORD) - National Health and Environmental Effects Research Laboratory
(NHEERL)"
                htPublisher!pubplace = "Corvallis, OR"

                GetDefaultRecord(htPublisher, MetadataMdbFile, "1b_Publisher")

                'set the metadata element to the obtained record value(s)
                pPropertySet.SetPropertyX("idinfo/citation/citeinfo/pubinfo/publish", htPublisher!publish, esriXmlPropertyType.esriXPTText,
esriXmlSetPropertyAction.esriXSPAAddOrReplace, False)
                pPropertySet.SetPropertyX("idinfo/citation/citeinfo/pubinfo/pubplace", htPublisher!pubplace, esriXmlPropertyType.esriXPTText,
esriXmlSetPropertyAction.esriXSPAAddOrReplace, False)

                'retrieve today's date for pubdate
                spubdate = Format(Today(), "yyyyMMdd")

                pPropertySet.SetPropertyX("idinfo/citation/citeinfo/pubdate", spubdate, esriXmlPropertyType.esriXPTText,
esriXmlSetPropertyAction.esriXSPAAddOrReplace, False)

                pPropertySet.SetPropertyX("idinfo/descript/abstract", "", esriXmlPropertyType.esriXPTText,
esriXmlSetPropertyAction.esriXSPAAddOrReplace, False)
                pPropertySet.SetPropertyX("idinfo/descript/purpose", "", esriXmlPropertyType.esriXPTText,
esriXmlSetPropertyAction.esriXSPAAddOrReplace, False)
                pPropertySet.SetPropertyX("idinfo/descript/supplinf", "This dataset is a local copy for Western Ecology Division (WED) use only.",
esriXmlPropertyType.esriXPTText, esriXmlSetPropertyAction.esriXSPAAddOrReplace, False)

                'TO-DO: retrieve today's date (YYYY, YYYYMM, or YYYYMMDD) for caldate or leave "Unknown"?
                pPropertySet.SetPropertyX("idinfo/timeperd/timeinfo/sngdate/caldate", "Unknown", esriXmlPropertyType.esriXPTText,
esriXmlSetPropertyAction.esriXSPAAddOrReplace, False)

                '1e_Currentness: current
                Dim htCurrentness As Hashtable = New Hashtable

                'initialize a value in case GetDefaultRecord fails
```

```
                htCurrentness!current = "Unknown"

                GetDefaultRecord(htCurrentness, MetadataMdbFile, "1e_Currentness")

                'set the metadata element to the obtained record value(s)
                pPropertySet.SetPropertyX("idinfo/timeperd/current", htCurrentness!current, esriXmlPropertyType.esriXPTText,
esriXmlSetPropertyAction.esriXSPAAddOrReplace, False)

                '1d_Progress: progress
                Dim htProgress As Hashtable = New Hashtable

                'initialize a value in case GetDefaultRecord fails
                htProgress!progress = "In work"

                GetDefaultRecord(htProgress, MetadataMdbFile, "1d_Progress")

                'set the metadata element to the obtained record value(s)
                pPropertySet.SetPropertyX("idinfo/status/progress", htProgress!progress, esriXmlPropertyType.esriXPTText,
esriXmlSetPropertyAction.esriXSPAAddOrReplace, False)

                '1f_Update: update
                Dim htUpdate As Hashtable = New Hashtable

                'initialize a value in case GetDefaultRecord fails
                htUpdate!update = "Unknown"

                GetDefaultRecord(htUpdate, MetadataMdbFile, "1f_Update")

                'set the metadata element to the obtained record value(s)
                pPropertySet.SetPropertyX("idinfo/status/update", htUpdate!update, esriXmlPropertyType.esriXPTText,
esriXmlSetPropertyAction.esriXSPAAddOrReplace, False)

                '1i_KeywordsISO: themekey
                Dim KeywordsISO(MAXKEYS) As String
                Dim numKeywordsISO As Integer = 0
                Dim i As Integer

                'initialize a value in case GetDefaultKeywords fails
                KeywordsISO(0) = "environment"

                GetDefaultKeywords(KeywordsISO, numKeywordsISO, MetadataMdbFile, "1i_KeywordsISO", 1)

                'set the metadata element to the obtained record value(s)
                pPropertySet.SetPropertyX("idinfo/keywords/theme[0]/themekt", "ISO 19115 Topic Category", esriXmlPropertyType.esriXPTText,
esriXmlSetPropertyAction.esriXSPAAddOrReplace, False)

                For i = 0 To numKeywordsISO - 1
                    pPropertySet.SetPropertyX("idinfo/keywords/theme[0]/themekey[" & CStr(i) & "]", KeywordsISO(i), esriXmlPropertyType.esriXPTText,
esriXmlSetPropertyAction.esriXSPAAddOrReplace, False)
                Next

                '1i_KeywordsEPA: themekey
                Dim KeywordsEPA(MAXKEYS) As String
                Dim numKeywordsEPA As Integer = 0

                'initialize a value in case GetDefaultKeywords fails
                KeywordsEPA(0) = "Ecology"
```

```
                GetDefaultKeywords(KeywordsEPA, numKeywordsEPA, MetadataMdbFile, "1h_KeywordsEPA", 1)

                'set the metadata element to the obtained record value(s)
                pPropertySet.SetPropertyX("idinfo/keywords/theme[1]/themekt", "EPA GIS Keyword Thesaurus", esriXmlPropertyType.esriXPTText,
esriXmlSetPropertyAction.esriXSPAAddOrReplace, False)

                For i = 0 To numKeywordsEPA - 1
                    'MsgBox("Name: " & "idinfo/keywords/theme[1]/themekey[" & CStr(i) & "]" & vbNewLine & _
                    '"Value: " & KeywordsEPA(i))
                    pPropertySet.SetPropertyX("idinfo/keywords/theme[1]/themekey[" & CStr(i) & "]", KeywordsEPA(i), esriXmlPropertyType.esriXPTText,
esriXmlSetPropertyAction.esriXSPAAddOrReplace, False)
                Next

                '1i_KeywordsPlace: placekey
                Dim KeywordsPlace(MAXKEYS) As String
                Dim numKeywordsPlace As Integer = 0

                'initialize a value in case GetDefaultKeywords fails
                KeywordsPlace(0) = "United States"

                GetDefaultKeywords(KeywordsPlace, numKeywordsPlace, MetadataMdbFile, "1j_KeywordsPlace", 1)

                'set the metadata element to the obtained record value(s)
                pPropertySet.SetPropertyX("idinfo/keywords/place[0]/placekt", "None", esriXmlPropertyType.esriXPTText,
esriXmlSetPropertyAction.esriXSPAAddOrReplace, False)

                For i = 0 To numKeywordsPlace - 1
                    pPropertySet.SetPropertyX("idinfo/keywords/place[0]/placekey[" & CStr(i) & "]", KeywordsPlace(i), esriXmlPropertyType.esriXPTText,
esriXmlSetPropertyAction.esriXSPAAddOrReplace, False)
                Next

                '1k_Constraints: accconst, useconst, secsys, secclass, sechandl
                Dim htConstraints As Hashtable = New Hashtable

                'initialize a value in case GetDefaultRecord fails
                htConstraints!accconst = "Data Are Restricted to Internal EPA Personnel Only"
                htConstraints!useconst = "These data should not be distributed to users unless distribution is explicitly granted.  Please check
sources, scale, accuracy, currentness and other available information. Please confirm that you are using the most recent copy of both data and
metadata.  Acknowledgement of the EPA would be appreciated."
                htConstraints!secsys = "FIPS Pub 199"
                htConstraints!secclass = "Medium Confidentiality"
                htConstraints!sechandl = "Standard Technical Controls"

                GetDefaultRecord(htConstraints, MetadataMdbFile, "1k_Constraints")

                'set the metadata element to the obtained record value(s)
                pPropertySet.SetPropertyX("idinfo/accconst", htConstraints!accconst, esriXmlPropertyType.esriXPTText,
esriXmlSetPropertyAction.esriXSPAAddOrReplace, False)
                pPropertySet.SetPropertyX("idinfo/useconst", htConstraints!useconst, esriXmlPropertyType.esriXPTText,
esriXmlSetPropertyAction.esriXSPAAddOrReplace, False)
                pPropertySet.SetPropertyX("idinfo/secinfo/secsys", htConstraints!secsys, esriXmlPropertyType.esriXPTText,
esriXmlSetPropertyAction.esriXSPAAddOrReplace, False)
                pPropertySet.SetPropertyX("idinfo/secinfo/secclass", htConstraints!secclass, esriXmlPropertyType.esriXPTText,
esriXmlSetPropertyAction.esriXSPAAddOrReplace, False)
                pPropertySet.SetPropertyX("idinfo/secinfo/sechandl", htConstraints!sechandl, esriXmlPropertyType.esriXPTText,
esriXmlSetPropertyAction.esriXSPAAddOrReplace, False)
```

```
                ' Contact_Information
                Dim htContact As Hashtable = New Hashtable

                'initialize a value in case GetDefaultRecord fails
                htContact!cntorg = "U.S. Environmental Protection Agency, Western Ecology Division"
                htContact!cntper = ""
                htContact!cntpos = ""
                htContact!addrtype = "mailing and physical address"
                htContact!address1 = "200 SW 35th St"
                htContact!address2 = ""
                htContact!city = "Corvallis"
                htContact!state = "OR"
                htContact!postal = "97333"
                htContact!cntvoice = "541-754-4600"
                htContact!cntemail = ""
                htContact!cntinst = "http://www.epa.gov/wed"

                GetDefaultRecord(htContact, MetadataMdbFile, "Contact_Information")

                'set the metadata element to the obtained record value(s)
                pPropertySet.SetPropertyX("idinfo/ptcontac/cntinfo/cntorgp/cntorg", htContact!cntorg, esriXmlPropertyType.esriXPTText,
esriXmlSetPropertyAction.esriXSPAAddOrReplace, False)
                pPropertySet.SetPropertyX("idinfo/ptcontac/cntinfo/cntorgp/cntper", htContact!cntper, esriXmlPropertyType.esriXPTText,
esriXmlSetPropertyAction.esriXSPAAddOrReplace, False)
                pPropertySet.SetPropertyX("idinfo/ptcontac/cntinfo/cntpos", htContact!cntpos, esriXmlPropertyType.esriXPTText,
esriXmlSetPropertyAction.esriXSPAAddOrReplace, False)
                pPropertySet.SetPropertyX("idinfo/ptcontac/cntinfo/cntaddr/addrtype", htContact!addrtype, esriXmlPropertyType.esriXPTText,
esriXmlSetPropertyAction.esriXSPAAddOrReplace, False)
                pPropertySet.SetPropertyX("idinfo/ptcontac/cntinfo/cntaddr/address", htContact!address1 & vbNewLine & htContact!address2,
esriXmlPropertyType.esriXPTText, esriXmlSetPropertyAction.esriXSPAAddOrReplace, False)
                pPropertySet.SetPropertyX("idinfo/ptcontac/cntinfo/cntaddr/city", htContact!city, esriXmlPropertyType.esriXPTText,
esriXmlSetPropertyAction.esriXSPAAddOrReplace, False)
                pPropertySet.SetPropertyX("idinfo/ptcontac/cntinfo/cntaddr/state", htContact!state, esriXmlPropertyType.esriXPTText,
esriXmlSetPropertyAction.esriXSPAAddOrReplace, False)
                pPropertySet.SetPropertyX("idinfo/ptcontac/cntinfo/cntaddr/postal", htContact!postal, esriXmlPropertyType.esriXPTText,
esriXmlSetPropertyAction.esriXSPAAddOrReplace, False)
                pPropertySet.SetPropertyX("idinfo/ptcontac/cntinfo/cntvoice", htContact!cntvoice, esriXmlPropertyType.esriXPTText,
esriXmlSetPropertyAction.esriXSPAAddOrReplace, False)
                pPropertySet.SetPropertyX("idinfo/ptcontac/cntinfo/cntemail", htContact!cntemail, esriXmlPropertyType.esriXPTText,
esriXmlSetPropertyAction.esriXSPAAddOrReplace, False)
                pPropertySet.SetPropertyX("idinfo/ptcontac/cntinfo/cntinst", htContact!cntinst, esriXmlPropertyType.esriXPTText,
esriXmlSetPropertyAction.esriXSPAAddOrReplace, False)

                pPropertySet.SetPropertyX("dataqual/logic", "Unknown", esriXmlPropertyType.esriXPTText, esriXmlSetPropertyAction.esriXSPAAddOrReplace,
False)

                '2a_Completeness: complete
                Dim htCompleteness As Hashtable = New Hashtable

                'initialize a value in case GetDefaultRecord fails
                htCompleteness!complete = "Unknown"

                GetDefaultRecord(htCompleteness, MetadataMdbFile, "2a_Completeness")

                'set the metadata element to the obtained record value(s)
```

```
                pPropertySet.SetPropertyX("dataqual/complete", htCompleteness!complete, esriXmlPropertyType.esriXPTText,
esriXmlSetPropertyAction.esriXSPAAddOrReplace, False)

                pPropertySet.SetPropertyX("dataqual/posacc/horizpa/horizpar", "Unknown", esriXmlPropertyType.esriXPTText,
esriXmlSetPropertyAction.esriXSPAAddOrReplace, False)

                pPropertySet.SetPropertyX("dataqual/lineage/procstep/procdesc", "Default EPA metadata created", esriXmlPropertyType.esriXPTText,
esriXmlSetPropertyAction.esriXSPAAddOrReplace, False)

                'retrieve today's date for procdate
                sprocdate = Format(Today(), "yyyyMMdd")

                pPropertySet.SetPropertyX("dataqual/lineage/procstep/procdate", sprocdate, esriXmlPropertyType.esriXPTText,
esriXmlSetPropertyAction.esriXSPAAddOrReplace, False)

                'set the metadata element to the obtained record value(s)
                pPropertySet.SetPropertyX("distinfo/distrib/cntinfo/cntorgp/cntorg", htContact!cntorg, esriXmlPropertyType.esriXPTText,
esriXmlSetPropertyAction.esriXSPAAddOrReplace, False)
                pPropertySet.SetPropertyX("distinfo/distrib/cntinfo/cntorgp/cntper", htContact!cntper, esriXmlPropertyType.esriXPTText,
esriXmlSetPropertyAction.esriXSPAAddOrReplace, False)
                pPropertySet.SetPropertyX("distinfo/distrib/cntinfo/cntpos", htContact!cntpos, esriXmlPropertyType.esriXPTText,
esriXmlSetPropertyAction.esriXSPAAddOrReplace, False)
                pPropertySet.SetPropertyX("distinfo/distrib/cntinfo/cntaddr/addrtype", htContact!addrtype, esriXmlPropertyType.esriXPTText,
esriXmlSetPropertyAction.esriXSPAAddOrReplace, False)
                pPropertySet.SetPropertyX("distinfo/distrib/cntinfo/cntaddr/address", htContact!address1 & vbNewLine & htContact!address2,
esriXmlPropertyType.esriXPTText, esriXmlSetPropertyAction.esriXSPAAddOrReplace, False)
                pPropertySet.SetPropertyX("distinfo/distrib/cntinfo/cntaddr/city", htContact!city, esriXmlPropertyType.esriXPTText,
esriXmlSetPropertyAction.esriXSPAAddOrReplace, False)
                pPropertySet.SetPropertyX("distinfo/distrib/cntinfo/cntaddr/state", htContact!state, esriXmlPropertyType.esriXPTText,
esriXmlSetPropertyAction.esriXSPAAddOrReplace, False)
                pPropertySet.SetPropertyX("distinfo/distrib/cntinfo/cntaddr/postal", htContact!postal, esriXmlPropertyType.esriXPTText,
esriXmlSetPropertyAction.esriXSPAAddOrReplace, False)
                pPropertySet.SetPropertyX("distinfo/distrib/cntinfo/cntvoice", htContact!cntvoice, esriXmlPropertyType.esriXPTText,
esriXmlSetPropertyAction.esriXSPAAddOrReplace, False)
                pPropertySet.SetPropertyX("distinfo/distrib/cntinfo/cntemail", htContact!cntemail, esriXmlPropertyType.esriXPTText,
esriXmlSetPropertyAction.esriXSPAAddOrReplace, False)
                pPropertySet.SetPropertyX("distinfo/distrib/cntinfo/cntinst", htContact!cntinst, esriXmlPropertyType.esriXPTText,
esriXmlSetPropertyAction.esriXSPAAddOrReplace, False)

                '3a_ResourceType: resdesc
                Dim htResourceType As Hashtable = New Hashtable

                'initialize a value in case GetDefaultRecord fails
                htResourceType!resdesc = "Offline Data"

                GetDefaultRecord(htResourceType, MetadataMdbFile, "3a_ResourceType")

                'set the metadata element to the obtained record value(s)
                pPropertySet.SetPropertyX("distinfo/resdesc", htResourceType!resdesc, esriXmlPropertyType.esriXPTText,
esriXmlSetPropertyAction.esriXSPAAddOrReplace, False)

                '3b_DistributionLiability: distliab
                Dim htDistributionLiability As Hashtable = New Hashtable

                'initialize a value in case GetDefaultRecord fails
                htDistributionLiability!distliab = "Although these data have been processed successfully on a computer system at the Environmental
Protection Agency, no warranty expressed or implied is made regarding the accuracy or utility of the data on any other system or for general or
```

```
scientific purposes, nor shall the act of distribution constitute any such warranty. It is also strongly recommended that careful attention be paid to
the contents of the metadata file associated with these data to evaluate data set limitations, restrictions or intended use. The U.S. Environmental
Protection Agency shall not be held liable for improper or incorrect use of the data described and/or contained herein."

                GetDefaultRecord(htDistributionLiability, MetadataMdbFile, "3b_DistributionLiability")

                'set the metadata element to the obtained record value(s)
                pPropertySet.SetPropertyX("distinfo/distliab", htDistributionLiability!distliab, esriXmlPropertyType.esriXPTText,
esriXmlSetPropertyAction.esriXSPAAddOrReplace, False)

                pPropertySet.SetPropertyX("metainfo/metc/cntinfo/cntorgp/cntorg", htContact!cntorg, esriXmlPropertyType.esriXPTText,
esriXmlSetPropertyAction.esriXSPAAddOrReplace, False)
                pPropertySet.SetPropertyX("metainfo/metc/cntinfo/cntorgp/cntper", htContact!cntper, esriXmlPropertyType.esriXPTText,
esriXmlSetPropertyAction.esriXSPAAddOrReplace, False)
                pPropertySet.SetPropertyX("metainfo/metc/cntinfo/cntpos", htContact!cntpos, esriXmlPropertyType.esriXPTText,
esriXmlSetPropertyAction.esriXSPAAddOrReplace, False)
                pPropertySet.SetPropertyX("metainfo/metc/cntinfo/cntaddr/addrtype", htContact!addrtype, esriXmlPropertyType.esriXPTText,
esriXmlSetPropertyAction.esriXSPAAddOrReplace, False)
                pPropertySet.SetPropertyX("metainfo/metc/cntinfo/cntaddr/address", htContact!address1 & vbNewLine & htContact!address2,
esriXmlPropertyType.esriXPTText, esriXmlSetPropertyAction.esriXSPAAddOrReplace, False)
                pPropertySet.SetPropertyX("metainfo/metc/cntinfo/cntaddr/city", htContact!city, esriXmlPropertyType.esriXPTText,
esriXmlSetPropertyAction.esriXSPAAddOrReplace, False)
                pPropertySet.SetPropertyX("metainfo/metc/cntinfo/cntaddr/state", htContact!state, esriXmlPropertyType.esriXPTText,
esriXmlSetPropertyAction.esriXSPAAddOrReplace, False)
                pPropertySet.SetPropertyX("metainfo/metc/cntinfo/cntaddr/postal", htContact!postal, esriXmlPropertyType.esriXPTText,
esriXmlSetPropertyAction.esriXSPAAddOrReplace, False)
                pPropertySet.SetPropertyX("metainfo/metc/cntinfo/cntvoice", htContact!cntvoice, esriXmlPropertyType.esriXPTText,
esriXmlSetPropertyAction.esriXSPAAddOrReplace, False)
                pPropertySet.SetPropertyX("metainfo/metc/cntinfo/cntemail", htContact!cntemail, esriXmlPropertyType.esriXPTText,
esriXmlSetPropertyAction.esriXSPAAddOrReplace, False)
                pPropertySet.SetPropertyX("metainfo/metc/cntinfo/cntinst", htContact!cntinst, esriXmlPropertyType.esriXPTText,
esriXmlSetPropertyAction.esriXSPAAddOrReplace, False)

                '3c_MetadataStandard: metstdn, metstdv
                Dim htMetadataStandard As Hashtable = New Hashtable

                'initialize a value in case GetDefaultRecord fails
                htMetadataStandard!metstdn = "FGDC Content Standard for Digital Geospatial Metadata"
                htMetadataStandard!metstdv = "FGDC-STD-001-1998"

                GetDefaultRecord(htMetadataStandard, MetadataMdbFile, "3c_MetadataStandard")

                'set the metadata element to the obtained record value(s)
                pPropertySet.SetPropertyX("metainfo/metstdn", htMetadataStandard!metstdn, esriXmlPropertyType.esriXPTText,
esriXmlSetPropertyAction.esriXSPAAddOrReplace, False)
                pPropertySet.SetPropertyX("metainfo/metstdv", htMetadataStandard!metstdv, esriXmlPropertyType.esriXPTText,
esriXmlSetPropertyAction.esriXSPAAddOrReplace, False)

            Case "CoverageEntity"
                ' "Value" variable type: IArcInfoTable
                '
                ' Provides access to INFO tables and a coverage feature class's feature
                ' attribute table. Used to record attribute information and to count the
                ' number of records or features. Can be multiple (CoverageEntity[i])

            Case "CoverageFeatureClass"
                ' "Value" variable type: ICoverageFeatureClass
```

```
                '
                ' Provides information about a coverage feature class including the type
                ' of feature class and whether it has an attribute table or topology. Used
                ' to record feature information. Can be multiple (CoverageFeatureClass[i])

            Case "DatasetLocation"
                ' "Value" variable type: String
                '
                ' Location of data set on disk, or for ArcSDE data sets, connection
                ' information for accessing the geodatabase.

                'get on-disk file location (is path obtained appropriate, or must it be reworked?)
                'Argh: EPA validation pukes if it is not preceded with http://, ftp://, or file://
                onlink1 = "file:// " & Value

                pPropertySet.SetPropertyX("idinfo/citation/citeinfo/onlink[0]", onlink1, esriXmlPropertyType.esriXPTText,
esriXmlSetPropertyAction.esriXSPAAddOrReplace, True)

                onlink2 = "http://wedcor.cor.epa.gov/pages/gisCatalog"

                'TO-DO: determine if data to be made available via web URL (currently it is not)
                pPropertySet.SetPropertyX("idinfo/citation/citeinfo/onlink[1]", onlink2, esriXmlPropertyType.esriXPTText,
esriXmlSetPropertyAction.esriXSPAAddOrReplace, False)

            Case "DatasetName"
                ' "Value" variable type: String
                '
                ' Name of the data set derived from either the file name or the table name.
                ' This section is always executed

                'retrieve GIS dataset filename to insert as default title
                pPropertySet.SetPropertyX("idinfo/citation/citeinfo/title", Value, esriXmlPropertyType.esriXPTText,
esriXmlSetPropertyAction.esriXSPAAddIfNotExists, False)

            Case "DatasetSize"
                ' "Value" variable type: String
                '
                ' Size of the data set on disk. (Not used for objects stored in a geodatabase.)

            Case "DDExtent"
                ' "Value" variable type: IEnvelope
                '
                ' Envelope containing the data set's geographic data. Used to record
                ' its extent in decimal degrees.

                'Must set interface to be an object of correct type, otherwise properties on the Value
                'object will return Null
                Dim pBoundBox As ESRI.ArcGIS.Geometry.IEnvelope
                pBoundBox = Value

                'Check that the envelope is not empty so we don't write zero tags
                If Not pBoundBox.IsEmpty Then
                    'obtain bounding coordinates from GIS dataset
                    pPropertySet.SetPropertyX("idinfo/spdom/bounding/northbc", CStr(Math.Round(pBoundBox.YMax, 6, MidpointRounding.AwayFromZero)),
esriXmlPropertyType.esriXPTText, esriXmlSetPropertyAction.esriXSPAAddOrReplace, True)
                    pPropertySet.SetPropertyX("idinfo/spdom/bounding/southbc", CStr(Math.Round(pBoundBox.YMin, 6, MidpointRounding.AwayFromZero)),
esriXmlPropertyType.esriXPTText, esriXmlSetPropertyAction.esriXSPAAddOrReplace, True)
```

```
                    pPropertySet.SetPropertyX("idinfo/spdom/bounding/eastbc", CStr(Math.Round(pBoundBox.XMax, 6, MidpointRounding.AwayFromZero)),
esriXmlPropertyType.esriXPTText, esriXmlSetPropertyAction.esriXSPAAddOrReplace, True)
                    pPropertySet.SetPropertyX("idinfo/spdom/bounding/westbc", CStr(Math.Round(pBoundBox.XMin, 6, MidpointRounding.AwayFromZero)),
esriXmlPropertyType.esriXPTText, esriXmlSetPropertyAction.esriXSPAAddOrReplace, True)

                    'Add property explaining what spatial ref system the coords are in
                    pPropertySet.SetAttribute("idinfo/spdom/bounding", "SpatRefSystem", "LatLong", esriXmlSetPropertyAction.esriXSPAAddOrReplace)
                End If

            Case "Entity"
                ' "Value" variable type: IClass
                '
                ' Provides access to an object class such as a table or feature class.
                ' Used to record full attribute information. Can be multiple (Entity[i])

            Case "EntityBrief"
                ' "Value" variable type: IClass
                '
                ' Provides access to an object class such as a table or feature class.
                ' Used to record brief entity information for the feature classes contained
                ' in a feature data set. Can be multiple (EntityBrief[i])

            Case "Environment"
                ' "Value" variable type: String
                '
                ' Operating system, software name, and version of the computer.

                ' write to which field?
                ' MsgBox("Environment is " & Value)

            Case "FeatureClass"
                ' "Value" variable type: IFeatureClass
                '
                ' Provides access to a feature class. Used to record feature information
                ' such as feature and geometry type. Can be multiple (FeatureClass[i])

            Case "GeoForm"
                ' "Value" variable type: String.
                '
                ' One of: "raster digital data", "remote sensing image", "tabular digital data", "vector digital data"
                ' Mode in which the spatial data is represented.

                ' write to which field?
                ' MsgBox("GeoForm is " & Value)

            Case "GeometryType"
                ' "Value" variable type: String
                '
                ' One of: "Vector", "Raster"
                ' Type of geometry stored in the data set.

                ' write to which field?
                ' MsgBox("GeometryType is " & Value)

            Case "Language"
                ' "Value" variable type: String
                '
```

```
            ' Language of the data and the metadata. Derived from the operating
            ' system's default input locale.

        Case "MetadataDate"
            ' "Value" variable type: String
            '
            ' The current date. Used to record when the metadata was last updated.

            smetd = Value

            pPropertySet.SetPropertyX("metainfo/metd", smetd, esriXmlPropertyType.esriXPTText, esriXmlSetPropertyAction.esriXSPAAddOrReplace, True)

            'today's date plus 4 years for metfrd
            cultinfo = New CultureInfo("en-US")

            metd = DateTime.ParseExact(smetd, "yyyyMMdd", cultinfo)
            metfrd = metd.AddYears(4)
            smetfrd = metfrd.ToString("yyyyMMdd")

            pPropertySet.SetPropertyX("metainfo/metfrd", smetfrd, esriXmlPropertyType.esriXPTText, esriXmlSetPropertyAction.esriXSPAAddOrReplace,
True)

        Case "MetadataStandard"
            ' "Value" variable type: String
            '
            ' The name of the metadata standard supported by ArcCatalog: version 2
            ' of the CSDGM. Used to record information about the standard to which
            ' the metadata was created.

        Case "NativeExtent"
            ' "Value" variable type: IEnvelope
            '
            ' Envelope containing the data set's geographic data. Used to record its
            ' actual extent, either in projected or decimal degree coordinates.

            'Must QI into an object of correct type as properties on the Value
            'object will return Null

            Dim pBoundBox As ESRI.ArcGIS.Geometry.IEnvelope
            pBoundBox = Value

            'Check that the envelope is not empty so we don't write zero tags

            If Not pBoundBox.IsEmpty Then
                pPropertySet.SetPropertyX("idinfo/spdom/bounding/northbc", CStr(Math.Round(pBoundBox.YMax, 6, MidpointRounding.AwayFromZero)),
esriXmlPropertyType.esriXPTText, esriXmlSetPropertyAction.esriXSPAAddOrReplace, True)
                pPropertySet.SetPropertyX("idinfo/spdom/bounding/southbc", CStr(Math.Round(pBoundBox.YMin, 6, MidpointRounding.AwayFromZero)),
esriXmlPropertyType.esriXPTText, esriXmlSetPropertyAction.esriXSPAAddOrReplace, True)
                pPropertySet.SetPropertyX("idinfo/spdom/bounding/eastbc", CStr(Math.Round(pBoundBox.XMax, 6, MidpointRounding.AwayFromZero)),
esriXmlPropertyType.esriXPTText, esriXmlSetPropertyAction.esriXSPAAddOrReplace, True)
                pPropertySet.SetPropertyX("idinfo/spdom/bounding/westbc", CStr(Math.Round(pBoundBox.XMin, 6, MidpointRounding.AwayFromZero)),
esriXmlPropertyType.esriXPTText, esriXmlSetPropertyAction.esriXSPAAddOrReplace, True)

                'Add property explaining what spatial ref system the coords are in
                pPropertySet.SetAttribute("idinfo/spdom/bounding", "SpatRefSystem", "LatLong", esriXmlSetPropertyAction.esriXSPAAddOrReplace)
            End If
```

```
Case "NativeForm"
    ' "Value" variable type: String
    '
    ' Type of the data set, for example, "Shapefile", "Personal Geodatabase Table",
    ' or "Raster Dataset". This string is the same as that displayed in the Type
    ' column in ArcCatalog contents view.

    ' write to which field?
    ' MsgBox("NativeForm is " & Value)

Case "NetworkRule"
    ' "Value" variable type: IRule
    '
    ' Provides information about the connectivity rules in a geometric network.
    ' Can be multiple (NetworkRule[i])

Case "NetworkSchema"
    ' "Value" variable type: INetSchema
    '
    ' Provides information about the schema of a geometric network such
    ' as element classes, ancillary roles, and weights.

Case "OperatingSystem"
    ' "Value" variable type: String
    '
    ' The name of the operating system on the computer used to create or update
    ' the metadata (duplicated in the itemDesc Environment).

Case "RasterBand"
    ' "Value" variable type: IRasterBand
    '
    ' Provides access to information about a raster band including its
    ' attribute table and color map.

Case "RasterDataset"
    ' "Value" variable type: IRasterDataset (may also support IRasterBandCollection)
    '
    ' Provides information about a raster data set such as its format and compression type.

Case "Relationship"
    ' "Value" variable type: IRelationshipClass
    '
    ' Used to record detailed information about a relationship. Information such
    ' as the relationship's origin, destination, and cardinality is included.
    ' Can be multiple (Relationship[i])

Case "RelationshipBrief"
    ' "Value" variable type: IRelationshipClass
    '
    ' Used to record brief relationship information for objects that
    ' participate in a relationship. Can be multiple (RelationshipBrief[i])

Case "Software"
    ' "Value" variable type: String
    '
    ' The name and version of the software used to create or update
    ' the metadata (duplicated in the itemDesc Environment).
```

```
            Case "SpatialReference"
                ' "Value" variable type: ISpatialReference
                '
                ' Provides access to the data set's spatial reference.

                Dim pSpatRef As ESRI.ArcGIS.Geometry.ISpatialReference
                Dim pSpatRefRez As ESRI.ArcGIS.Geometry.ISpatialReferenceResolution

                pSpatRef = Value
                pSpatRefRez = Value

                'Dim dXmax As Double
                'Dim dYmax As Double
                'Dim dXmin As Double
                'Dim dYmin As Double
                '
                'get the xy domain extent of the dataset
                'pSpatRef.GetDomain(dXmin, dXmax, dYmin, dYmax)
                'MsgBox("dXmin: " & CStr(dXmin) & ", " & "dXmax: " & CStr(dXmax) & ", " & "dYmin: " & CStr(dYmin) & ", " & "dYmax: " & CStr(dYmax))

                'Dim dFalseX As Double
                'Dim dFalseY As Double
                'Dim dXYUnits As Double
                '
                'get the false origin and units of the dataset
                'pSpatRef.GetFalseOriginAndUnits(dFalseX, dFalseY, dXYUnits)
                'MsgBox("dFalseX: " & CStr(dFalseX) & ", " & "dFalseY: " & CStr(dFalseY) & ", " & "dXYUnits: " & CStr(dXYUnits))

                If TypeOf pSpatRef Is ESRI.ArcGIS.Geometry.IUnknownCoordinateSystem Then
                    'MsgBox("Unknown CS")
                ElseIf TypeOf pSpatRef Is ESRI.ArcGIS.Geometry.IProjectedCoordinateSystem Then
                    'MsgBox("Projected CS")
                    Dim pPCS As ESRI.ArcGIS.Geometry.IProjectedCoordinateSystem
                    pPCS = pSpatRef

                    If pSpatRef.HasXYPrecision Then
                        pPropertySet.SetPropertyX("spref/horizsys/geograph/latres", CStr(Math.Round(pSpatRefRez.XYResolution(True), 6,
MidpointRounding.AwayFromZero)), esriXmlPropertyType.esriXPTText, esriXmlSetPropertyAction.esriXSPAAddOrReplace, True)
                        pPropertySet.SetPropertyX("spref/horizsys/geograph/longres", CStr(Math.Round(pSpatRefRez.XYResolution(True), 6,
MidpointRounding.AwayFromZero)), esriXmlPropertyType.esriXPTText, esriXmlSetPropertyAction.esriXSPAAddOrReplace, True)
                    Else
                        pPropertySet.SetPropertyX("spref/horizsys/geograph/latres", "0.00", esriXmlPropertyType.esriXPTText,
esriXmlSetPropertyAction.esriXSPAAddOrReplace, True)
                        pPropertySet.SetPropertyX("spref/horizsys/geograph/longres", "0.00", esriXmlPropertyType.esriXPTText,
esriXmlSetPropertyAction.esriXSPAAddOrReplace, True)
                    End If

                    pPropertySet.SetPropertyX("spref/horizsys/geograph/geogunit", ESRItoFGDCname("geogunit", pPCS.CoordinateUnit.Name),
esriXmlPropertyType.esriXPTText, esriXmlSetPropertyAction.esriXSPAAddOrReplace, True)

                    pPropertySet.SetPropertyX("spref/horizsys/geodetic/horizdn", ESRItoFGDCname("horizdn", pPCS.GeographicCoordinateSystem.Datum.Name),
esriXmlPropertyType.esriXPTText, esriXmlSetPropertyAction.esriXSPAAddOrReplace, True)
                    pPropertySet.SetPropertyX("spref/horizsys/geodetic/ellips", ESRItoFGDCname("ellips",
pPCS.GeographicCoordinateSystem.Datum.Spheroid.Name), esriXmlPropertyType.esriXPTText, esriXmlSetPropertyAction.esriXSPAAddOrReplace, True)
```

```
                    pPropertySet.SetPropertyX("spref/horizsys/geodetic/semiaxis",
CStr(Math.Round(pPCS.GeographicCoordinateSystem.Datum.Spheroid.SemiMajorAxis, 6, MidpointRounding.AwayFromZero)), esriXmlPropertyType.esriXPTText,
esriXmlSetPropertyAction.esriXSPAAddOrReplace, True)
                    pPropertySet.SetPropertyX("spref/horizsys/geodetic/denflat", CStr(Math.Round(1 /
pPCS.GeographicCoordinateSystem.Datum.Spheroid.Flattening, 6, MidpointRounding.AwayFromZero)), esriXmlPropertyType.esriXPTText,
esriXmlSetPropertyAction.esriXSPAAddOrReplace, True)


                ElseIf TypeOf pSpatRef Is ESRI.ArcGIS.Geometry.IGeographicCoordinateSystem Then
                    Dim pGCS As ESRI.ArcGIS.Geometry.IGeographicCoordinateSystem
                    pGCS = pSpatRef

                    If pSpatRef.HasXYPrecision Then
                        pPropertySet.SetPropertyX("spref/horizsys/geograph/latres", CStr(Math.Round(pSpatRefRez.XYResolution(True), 6,
MidpointRounding.AwayFromZero)), esriXmlPropertyType.esriXPTText, esriXmlSetPropertyAction.esriXSPAAddOrReplace, True)
                        pPropertySet.SetPropertyX("spref/horizsys/geograph/longres", CStr(Math.Round(pSpatRefRez.XYResolution(True), 6,
MidpointRounding.AwayFromZero)), esriXmlPropertyType.esriXPTText, esriXmlSetPropertyAction.esriXSPAAddOrReplace, True)
                    Else
                        pPropertySet.SetPropertyX("spref/horizsys/geograph/latres", "0", esriXmlPropertyType.esriXPTText,
esriXmlSetPropertyAction.esriXSPAAddOrReplace, True)
                        pPropertySet.SetPropertyX("spref/horizsys/geograph/longres", "0", esriXmlPropertyType.esriXPTText,
esriXmlSetPropertyAction.esriXSPAAddOrReplace, True)
                    End If

                    pPropertySet.SetPropertyX("spref/horizsys/geograph/geogunit", ESRItoFGDCname("geogunit", pGCS.CoordinateUnit.Name),
esriXmlPropertyType.esriXPTText, esriXmlSetPropertyAction.esriXSPAAddOrReplace, True)

                    pPropertySet.SetPropertyX("spref/horizsys/geodetic/horizdn", ESRItoFGDCname("horizdn", pGCS.Datum.Name),
esriXmlPropertyType.esriXPTText, esriXmlSetPropertyAction.esriXSPAAddOrReplace, True)
                    pPropertySet.SetPropertyX("spref/horizsys/geodetic/ellips", ESRItoFGDCname("ellips", pGCS.Datum.Spheroid.Name),
esriXmlPropertyType.esriXPTText, esriXmlSetPropertyAction.esriXSPAAddOrReplace, True)
                    pPropertySet.SetPropertyX("spref/horizsys/geodetic/semiaxis", CStr(Math.Round(pGCS.Datum.Spheroid.SemiMajorAxis, 6,
MidpointRounding.AwayFromZero)), esriXmlPropertyType.esriXPTText, esriXmlSetPropertyAction.esriXSPAAddOrReplace, True)
                    pPropertySet.SetPropertyX("spref/horizsys/geodetic/denflat", CStr(Math.Round(1 / pGCS.Datum.Spheroid.Flattening, 6,
MidpointRounding.AwayFromZero)), esriXmlPropertyType.esriXPTText, esriXmlSetPropertyAction.esriXSPAAddOrReplace, True)

                Else
                    'MsgBox("Unknown CoClass")
                End If


            Case "Tin"
                ' "Value" variable type: ITin
                '
                ' Provides access to information about a TIN data set.

        End Select

    End Sub
    Private Function ESRItoFGDCname(ByVal sElementShortname As String, ByVal sESRIname As String) As String

        Select Case sElementShortname
            Case "geogunit"
                Select Case sESRIname
                    Case "Degree" : Return "Decimal degrees"
                End Select
            Case "horizdn"
```

```
            Select Case sESRIname
                Case "D_North_American_1983" : Return "North American Datum of 1983"
                Case "D_North_American_1927" : Return "North American Datum of 1927"
                Case "D_WGS_1984" : Return "World Geodetic System 1984"
            End Select
        Case "ellips"
            Select Case sESRIname
                Case "GRS_1980" : Return "Geodetic Reference System 1980"
                Case "Clarke_1866" : Return "Clarke 1866"
                Case "WGS_1984" : Return "WGS_1984"
            End Select
    End Select

    Return sESRIname

End Function
Private Sub GetDefaultRecord(ByVal DefaultRecord As Hashtable, ByVal MetadataMdbFile As String, ByVal DefaultTable As String)

    If fs.FileExists(MetadataMdbFile) Then

        Dim pAccessFact As IWorkspaceFactory
        Dim pAccessWorkspace As IWorkspace
        Dim pFeatworkspace As IFeatureWorkspace
        Dim pCursor As ICursor
        Dim pFields As IFields
        Dim pField As IField
        Dim pRow As IRow
        Dim i As Integer
        Dim pTable As ITable

        pAccessFact = New ESRI.ArcGIS.DataSourcesGDB.AccessWorkspaceFactory

        'open the metadata defaults database
        pAccessWorkspace = pAccessFact.OpenFromFile(MetadataMdbFile, 0)

        pFeatworkspace = pAccessWorkspace

        'open the table (table name passed to the subroutine as DefaultTable)
        pTable = pFeatworkspace.OpenTable(DefaultTable)

        If Not pTable Is Nothing Then
            Dim pQueryFilter As IQueryFilter
            pQueryFilter = New QueryFilter

            ' Set the where clause - the field name in the EME database to indicate
            ' the default contact is 'default' - a boolean field. So you can just use the
            ' fieldname as the WhereClause instead of fieldname = 'value'
            ' (Note: 'default' is a poor choice of field name since the word is also an
            ' SQL keyword. So the WhereClause below should not be confused with the
            ' keyword... if the field name changes to something like 'default_contact'
            ' you would put that in the WhereClause)

            pQueryFilter.WhereClause = "default"

            ' Execute the query filter
            pCursor = pTable.Search(pQueryFilter, True)
```

```
            pRow = pCursor.NextRow

            'loop through the table records
            If Not pRow Is Nothing Then
                pFields = pCursor.Fields
                'loop through the fields in the current record
                For i = 0 To (pFields.FieldCount - 1)
                    pField = pFields.Field(i)
                    'if the current field value is not empty, assign it to the hash
                    'table using the field name as the key
                    If Not IsDBNull(pRow.Value(i)) Then
                        DefaultRecord(pField.Name) = CStr(pRow.Value(i))
                    End If
                Next i
            Else
                MsgBox("No default contact was found." & vbNewLine _
                & "Proper default metadata values could not be written." & vbNewLine _
                & "Please specify a default contact in the EPA Metadata Editor database.")
            End If
        Else
            MsgBox("The EPA Metadata Editor contacts table was not found." & vbNewLine _
            & "Proper default metadata values could not be written." & vbNewLine _
            & "Please check for proper installation of the EPA Metadata Editor software.")
        End If
    End If

End Sub
Private Sub GetDefaultKeywords(ByVal DefaultKeywords() As String, ByRef numKeywords As Integer, ByVal MetadataMdbFile As String, ByVal DefaultTable
As String, ByVal FieldNumber As Integer)

    If fs.FileExists(MetadataMdbFile) Then

        Dim pAccessFact As IWorkspaceFactory
        Dim pAccessWorkspace As IWorkspace
        Dim pFeatworkspace As IFeatureWorkspace
        Dim pCursor As ICursor
        Dim pRow As IRow
        Dim r As Integer
        Dim pTable As ITable

        pAccessFact = New ESRI.ArcGIS.DataSourcesGDB.AccessWorkspaceFactory

        'open the metadata defaults file
        pAccessWorkspace = pAccessFact.OpenFromFile(MetadataMdbFile, 0)

        pFeatworkspace = pAccessWorkspace

        'open the table (table name passed to the subroutine as DefaultTable)
        pTable = pFeatworkspace.OpenTable(DefaultTable)

        If Not pTable Is Nothing Then
            Dim pQueryFilter As IQueryFilter
            pQueryFilter = New QueryFilter

            ' Set the where clause - the field name in the EME database to indicate
            ' the default contact is 'default' - a boolean field. So you can just use the
            ' fieldname as the WhereClause instead of fieldname = 'value'
```

```
            ' (Note: 'default' is a poor choice of field name since the word is also an
            ' SQL keyword. So the WhereClause below should not be confused with the
            ' keyword... if the field name changes to something like 'default_contact'
            ' you would put that in the WhereClause)

            pQueryFilter.WhereClause = "default"

            ' Use the PostfixClause to order the result set.
            Dim queryFilterDef As IQueryFilterDefinition
            queryFilterDef = CType(pQueryFilter, IQueryFilterDefinition)
            queryFilterDef.PostfixClause = "ORDER BY " & pTable.Fields.Field(FieldNumber).Name

            ' Execute the query filter
            pCursor = pTable.Search(pQueryFilter, True)

            pRow = pCursor.NextRow

            r = 0

            'loop through the table rows
            While Not pRow Is Nothing
                'if the value in the selected field number exists
                If Not IsDBNull(pRow.Value(FieldNumber)) Then
                    'add it to the list of keywords
                    DefaultKeywords(r) = CStr(pRow.Value(FieldNumber))
                    r = r + 1
                End If

                pRow = pCursor.NextRow
            End While

            numKeywords = r
        Else
            MsgBox("The EPA Metadata Editor contacts table was not found." & vbNewLine _
            & "Proper default metadata values could not be written." & vbNewLine _
            & "Please check for proper installation of the EPA Metadata Editor software.")
        End If
    End If

End Sub


End Class
```