

BIOGRAPHICAL INFORMATION

John G. Hedstrom
Programming Specialist
United Services Group / Great River Energy

Specific Responsibilities

Joined United Services Group in 2003. Responsible for providing a leadership role in evaluating, acquiring, integrating and designing software for the needs of USG and USG clients. In my role I develop and direct software testing procedures, programming and documentation. Consult with customers concerning the development of vertical market software.

Past Experience

Gemplus Corporation - Gemplus is the world's leading provider of solutions based on smart card technology. My role was as a Java software developer and network administrator.

Siemens Power Systems Control - Siemens Power Systems Control is part of one of the world's largest companies and serves the needs of the electric power industry. My role was as a software engineer for SCADA and Oracle systems.

Grocery Shopping Network – GSN develops database driven e-commerce internet solutions for grocery stores across the US. My role was as a software engineer developing websites in Coldfusion and MS SQL Server.

Educational Information

B.S. – Electrical Engineering, Saint Cloud State University, MN

Professional Memberships

GITA

Developing GIS data transfer with XML
John G. Hedstrom

United Services Group
17845 Highway 10 P.O. Box 341, Elk River, MN. 55330-0341

An overview of industry standards being developed in XML (Extensible Markup Language) like MultiSpeak will be discussed along with what considerations a potential user must evaluate before making a decision for ones industry. The NRECA (National Rural Electric Cooperative Association) has an initiative called MultiSpeak which utilizes XML for data transfer in a defined format for the electrical industry. A condensed history of XML will be discussed to its current state. Users need to understand XML in order to handle it for data. Several software applications use XML to process data in a way that is easy and convenient to use. I will show you additional ways to take advantage of this data. The discussion takes one developers perspective in designing a GIS data transfer application for rural electric cooperatives using the GIS MultiSpeak format for import and export of information related to the construction or line design of electrical facilities more commonly called staking sheets. Because XML is proliferating applications both on the Internet and the workstation, there are many initiatives under way to apply security technologies to XML data. The discussion would go through a few methods of tightening up the security of XML data.

What is XML? XML stands for eXtensible Markup Language. Developed by the World Wide Web Consortium (W3C) founded in 1994. The W3C in an international consortium of companies involved with the internet and the web with the premise to allow the web to evolve in a direction with open standards rather than in total chaos. XML was designed specifically for web documents. It allows developers to create their own custom tags enabling the definition, transmission, validation, and interpretation of data between applications and between organizations. XML tags are not predefined. The developer must define their own tags, unlike HTML where the tags are defined. XML uses a Document Type Definition (DTD) or an XML Schema to describe the data. XML with a DTD or XML Schema is designed to be self-descriptive. XML is a markup language much like HTML. XML is not a replacement for HTML. XML and HTML were designed with different goals. XML does not do anything. It was created to structure, store and to send information. A developer must write a piece of software to send, receive or display it

Here is an example of HTML: <HTML><HEAD> (enter here what document is about) <BODY></BODY></HTML>. All the information in the Web page fits in between <BODY> and </BODY> tags. XML is very similar to this. A simple example of XML is : <NOTE><TO>JOHN</TO><FROM>ERIN</FROM><HEADING>Good morning</HEADING><BODY>HELLO</BODY></NOTE>. Note that both use tags, but that the XML will do nothing without software to utilize it. The HTML will display a web page when placed on a web server. There are applications that are almost entirely XML. One such application is ArcPad by ESRI for GIS mapping. All of the layers, forms, coordinates, and images are stored in XML. Now that there is an understanding of what XML is, let's discuss how it is used in GIS systems and how the NRECA has developed an initiative to utilize it.

The National Rural Electric Cooperative Association (NRECA) is the national service organization dedicated to representing the national interests of cooperative electric utilities and the consumers they serve. The NRECA Board of Directors oversees the association's activities and consists of 47 members, one from each state in which there is an electric distribution cooperative.

Founded in 1942, NRECA was organized specifically to overcome World War II shortages of electric construction materials, to obtain insurance coverage for newly constructed rural electric cooperatives, and to mitigate wholesale power problems. Since those early days, NRECA has been an advocate for consumer-owned cooperatives on energy and operational issues as well as rural community and economic development.

The NRECA's more than 900 member cooperatives serve 37 million people in 47 states. Most of the 865 distribution systems are consumer-owned cooperatives; some are public power districts. NRECA membership includes other organizations formed by these local utilities: generation and transmission cooperatives for power supply, statewide and regional trade and service associations, supply and manufacturing cooperatives, data processing cooperatives and employee credit unions. Associate membership is open to equipment manufacturers and distributors, wholesalers, consultants and other entities that do business with members of the electric cooperative network.

The NRECA has an initiative called MultiSpeak. MultiSpeak is an effort between software vendors serving the electric utility industry and the NRECA. Its purpose is to define standard data interfaces to help make available cost-effective integrated software applications to meet the needs of small electrical utilities. The specification is intended to aid interface development so that software products from a variety of vendors can interoperate without the need for extensive custom development. The initiative supports a variety of hardware and software platforms, database programs, and programming languages so that data objects can be sent between two MultiSpeak-compliant software applications.

The MultiSpeak specification consists of three parts : the specification document, the XML schema, and the schema document. The MultiSpeak version 2.2 specification document includes a description of the structure of the data interfaces. The XML schema includes data objects, interface definitions, and message structures (Multispeak.xsd, MultispeakMessaging.xsd, and mspGeometry.xsd). The schema documentation is described in hypertext markup language.

The MultiSpeak-formatted messages are file-bases transfer implementations and the only message that applies is the Batch message. Each message includes a MultiSpeakMessageHeader element that carries a set of sub-elements and attributes. The specific documents being sent, time stamp, and authenticate the application sending the message. Every file-bases data transfer has a Batch message element contained in the MultiSpeakMessageHeader. Within each Batch message element is a MultiSpeak container element. Each MultiSpeak container element has attributes that identify the document type, type of connectivity, coordinate system, datum, and coordinate system

unit of measure. Every top level data element contained in the MultiSpeak container carries an object that identifies the specific instance of data and may contain other attributes. The only optional attribute is the Verb attribute. It indicates what action should be taken. The values permitted are : New, Change, Delete, Replace, Link, and Unlink. Although file-based transfers are less desirable than real time transfers, if sent small and frequently enough, they can approach performance of real time data exchanges.

MultiSpeak does not specify a file naming or network location convention which a client should search for new file-based data exchanges. These are left up to the vendors. These should be handled prior to development between the two vendors or vendor / client relationships.

An example of XML data transfer used for GIS Export with MultiSpeak is as follows:

The screenshot displays an XML data transfer interface. The main structure is as follows:

- XML**
 - version**: 1.0
 - MultiSpeakMessageHeader**
 - Version**: 2.2
 - xmlns**: http://www.multispeak.org/Schema/Version2.2
 - xmlns:xsi**: http://www.w3.org/2001/XMLSchema-instance
 - TimeStamp**: 10/28/2004 12:49:59 PM
 - DocumentID**
 - VendorApp**
 - AppName**: ADELINE_Multispeak_Export
 - AppVersion**: 0.1
 - Company**: United Services Group
 - Function**: STAKING-GIS
 - Batch**
 - MultiSpeak**
 - documentType**: dump
 - ohPrimaryLine** (4)
 - ugPrimaryLine** objectID= utility=
 - capacitorBank** objectID=932 utility=Anoka
 - overcurrentDeviceBank** (2)

objectID	utility	mapLocation	gridLocation	rotation
1 929	Anoka	mapLocation		
2 930	Anoka	mapLocation		
 - switchDeviceBank** objectID= utility=
 - regulatorBank** objectID=933 utility=Anoka
 - transformerBank** (3)
 - serviceLocation** objectID=40 utility=Anoka
 - riser** objectID=40 utility=Anoka
 - pole** (4)
 - streetLight** (2)
 - surfaceStructure** objectID=40 utility=Anoka
 - ohSecondaryLine** objectID=20 utility=Anoka
 - backgroundGraphics** objectID=40 utility=Anoka
 - workOrder** objectID=40 utility=Anoka

The example above is from United Services Group GIS XML MultiSpeak export for our Staking sheet software ADELIN. The information stored in the database can be exported to electrical cooperative via the XML export. Once received it can be imported into their system with ease because of the predetermined format specified by MultiSpeak. The program above was written in Visual Basic 6.0 with an Access 2000 database. The

XML can be viewed from an XML viewer. A free piece of software is XML Viewer, but prefer XML Spy.

Although the file is cumbersome – it is structured in a way that is readable and neatly organized. It expands rapidly and can be hard to read just like some comma delimited data files. The data is easily extracted into a usable format for GIS. The MultiSpeak initiative seems very repetitious for much of the data, but because so many vendors have such a variety of data to transmit it seems that the process covers as much of the unknown possibilities as possible. With that in mind it seems to work well for a system that would otherwise be very arduous to overcome for each new utility to transfer to. A lot of time would be wasted in developing the new data exports/imports. Once developed to meet the MultiSpeak specification the process is complete with only minor modifications if in fact none at all. In the developers opinion the MultiSpeak specification relieves a lot of the development cycle and increases productivity and lowers costs. Any number of programming languages, databases, hardware, and transport protocols can be utilized to each utilities skill sets. Once again lowering costs for investing in education, contractors, and hardware/software.

Security has always been vitally important in the business world to ensure the integrity of content and transactions, to maintain privacy and confidentiality, and to make sure information is used appropriately. However, in today's web-based business environment, the means for providing that security have changed. Using physical security no longer works as well as it did in the past when all the computing resources were locked in a central computing room with all jobs submitted locally. Efforts to create a single pervasive security infrastructure do not scale effectively to the Internet, due to the heterogeneous nature of hardware and software systems and to conflicting administrative, application and security requirements. There is too much to administer, too many applications, too many variations, and too rapid a pace of technology change to design a single infrastructure to meet all requirements effectively. Extensible standards are required that can adapt to changing requirements, that can incorporate new technologies while continuing to work with legacy technologies, and that can be deployed modularly as needed without requiring use of unnecessary portions. These standards should work well together, not replicate functionality, and should fit with new technologies to enable open distributed systems, application integration and content management.

The main XML security standards are : XML Digital Signature for integrity and signatures, XML Encryption for confidentiality, XML Key Management (XKMS) for key management, Security Assertion Markup Language (SAML) for making authentication and authorization assertions, and XML Access Control Markup Language (XACML) for stating authorization rules.

An essential requirement of new security standards is that they work naturally with content created using XML. XML is being adopted widely for a growing variety of applications and types of content. It is also forming the basis for distributed system

protocols to integrate applications across the Internet, such as Web Services protocols. XML languages are text based and designed to be extended and combined. It should be natural to provide integrity, confidentiality and other security benefits to entire XML documents or portions of these documents in a way that does not prevent further processing by standard XML tools. XML Security therefore must be integrated with XML in such a way as to maintain the advantages and capabilities of XML while adding necessary security capabilities. This is especially important in XML-based protocols, such as XML Protocol (XMLProt, Simple Object Access Protocol, SOAP), that are explicitly designed to allow intermediary processing and modification of messages.

Older security technologies provide a set of core security algorithms and technologies that can be used in XML Security, but the actual formats used to implement security requirements are inappropriate for most XML Security applications. One reason is that these standards use binary formats that require specialized software for interpretation and use, even for extracting portions of the security information. A second reason is that these standards are not designed for use with XML and do not support common XML technical approaches for managing content, such as specifying content with uniform resource identifier strings (URIs) or using other XML standard definitions for locating portions of XML content (like XPath). In addition, some existing security technologies assume that security-specific software will be integrated with applications to enable security. In practice, this is not always the case due to the details of custom integration.

XML Security addresses these issues by defining a common framework and processing rules that can be shared across applications using common tools, avoiding the need for extensive customization of applications to add security. XML Security reuses the concepts, algorithms and core technologies of legacy security systems while introducing changes necessary to support extensible integration with XML. This allows interoperability with a wide range of existing infrastructures and across deployments.

XML Security reduces barriers to adoption by defining the minimum modular mechanisms to obtain powerful results. By employing existing technologies and enabling use of XML paradigms and tools, XML Security minimizes the need to modify applications to meet security requirements.

References:

Hirsch, F., 2002, Getting started with XML security

MultiSpeak Joint Implementation Guidelines Working Group, 10/01/2003, MultiSpeak
Version 2 File-Based Transfer Implementation Guidelines