Editing XML metadata files with the aid of the open-source editor MEE

Tomasz Kubik[1], Bartosz Kopańczyk[2]

[1]Institute of Computer Engineering, Control and Robotics, Wrocław University of Technology, Poland, tomasz.kubik@pwr.wroc.pl
[2]GeoScope, Wrocław, Poland, bartek.kopanczyk@gmail.com

Abstract

Catalog services are essential components of a spatial data infrastructure. They support the ability to publish and search collections of geographic information metadata. The metadata describe the content, quality, condition, authorship, and any other characteristics of the geographic datasets and services in a standardized way. A common specification used for describing, validating and exchanging metadata is the XML Schema encoding.

The objective of this paper is to present some remarks on the role of metadata editors in relation to the catalog services and the requirements these editors are expected to meet. Thsee remarks are accompanied by the description of METadata Editor (MEE). This open-source XML metadata editor was designed as a desktop application offering a GUI, by which user can edit metadata elements in a user friendly way. The metadata elements are presented to the user in an organized manner, collecting all logically dependent metadata elements in one place. The editor supports the ISO 19139 and 19115 metadata profiles and can communicate with deegree catalog service. The design of the editor follows the object oriented programming paradigm and the MVC design pattern. Thanks to this approach the information model of metadata has been mapped onto the classes of the C# language. The code of the editor was tested on Windows and Linux/Wine systems, and the tests against the Mono platform are under way.

1. Introduction

For many years different public agencies and organizations have worked on collecting technologies, policies and institutional arrangements that facilitate the availability of, and access to spatial data. This work has resulted in establishing and operating of spatial data infrastructures that provide the basis for spatial data discovery, evaluation, and application for users and providers of various interests and responsibilities at a regional, national or global level (Maguire et al, 2005). Nowadays a big challenge is to assure that the infrastructures built are interoperable at the semantic, technological and organizational level. Thus, several initiatives and actions have been started in order to build cross-country infrastructures for spatial information. One of them is the INSPIRE directive, establishing the Infrastructure for Spatial Information in the European Community, accepted on the 15th of May 2007 (for the official INSPIRE documents please refer to

http://inspire.jrc.ec.europa.eu/). The directive specifies that the Member States shall establish and operate a network of services for the spatial data sets and services, for which metadata have been created, including: discovery services, view services, download services, transformation services, and services allowing spatial data services to be invoked. These services shall take into account relevant user requirements and shall be easy to use, available to the public and accessible via the Internet or any other appropriate means of telecommunication.

One of the essential components of the infrastructure, enabling outside users to find datasets or services relevant to their needs, is the catalog service (Nogueras-Iso et al, 2005). The catalog service collects metadata of datasets and services. Metadata describe the content, quality, condition, authorship, and any other characteristics of the datasets and services. Based on this information catalog service allows resource searching and evaluating, and provides the description of resource binding. The interoperability of catalog services is achieved by the requirements on their standardization (Nogueras-Iso et al, 2004). The standardization addresses different aspects of using metadata in the context of catalog services: the implementation aspect (metadata information model inside database or storage system), service interface aspect (encoding and incorporating of metadata inside a communication protocol), and the presentation aspect (metadata viewing on a user friendly interface). Typically the functionalities of catalog services are exposed to the user trough WEB applications. By these applications the user can access catalog service's storage in order to perform metadata searching, viewing, and editing. In many cases these applications are supported with thesauruses and dictionaries (Tindall et al, 2006). But WEB applications are not the only tools that can be used for metadata processing. This task can be done by XML metadata editors as well.

The XML metadata editors are desktop applications allowing metadata creation and modification. Metadata produced are stored in XML files which are compliant with a chosen metadata profile (an XML implementation of the metadata information model). Sometimes XML metadata editors can communicate with catalog services. In that case metadata created can be retrieved from or inserted into a catalog.

The above functionalities are not the only requirements such editors are expected to meet. More details on that subject will be provided throughout the paper. The paper is organized as follows. Remarks on metadata infrastructure and application profile for catalog services are provided in section 2. The description of the functionalities of XML metadata editors and the details of MEE editor implementation are discussed in section 3. The concluding remarks are given in the last section of the paper.

2. Metadata infrastructure

The implementation of interoperable metadata infrastructure is based on two kinds of regulations: a) metadata model description with XML encoding rules and associated XML schemas as specified in the series of ISO 191xx standards (ISO 19115/Cor.1:2006, ISO/TS 19139:2007); b) catalog model with behavior and interfaces description together with corresponding XML schemas as specified in OGC Catalog Services Web related documents (OGC 07-006r1, OGC 07-045). These two kinds of regulations are allied – catalogs allow registering metadata that adhere to

metadata schema standards. Thus both kinds of regulations should be considered in order to assure interoperability of the metadata systems. However the development of metadata systems in the heterogeneous environments of different communities is not an easy task. Due to the specific user requirements the model of metadata can be adapted by means of profiles. That is, metadata model can consist of a choice from the clauses, classes, options and parameters of base standards, or other profiles, and can include some extensions and/or restrictions on the metadata elements used. The similar arrangements apply to the catalog model – the model can be adjusted to the particular needs as well. Consequently, a catalog service implementation must be adjusted for every new profile – from a storage implementation, trough business logic, up to metadata editor design. More over, the implementation of metadata services is strongly influenced by the infrastructure architecture. For the XML metadata editor described later these are the main facts influencing its functionality and the reason for its existence. The following sections provide more on this.

2.1 Metadata profile

Community profiles might be created on different organizational level, from national to the local level, including branches and others. The national profile should be the common denominator of all profiles used by the components of national metadata infrastructure. By this requirement all interested parties can cooperate using at least this part of metadata model specification.

Metadata should be well structured and properly filled with data in order to eliminate service errors and/or data inconsistency. A list of software solutions that should use conformance testing includes an online validator and catalog services with an associated WEB application, as well as XML metadata editors. Following ISO 19106 there might be two classes of metadata conformance classes defined: class 1 and class 2. Class 1 is satisfied when metadata includes ONLY elements prescribed for the profile (pure subset of one or more ISO standards). Conformance class 2 assumes that there can be elements from a general standard as well as from the profile. All profile conditionals should be valid.

An XML metadata editor should support both kinds of conformance testing.

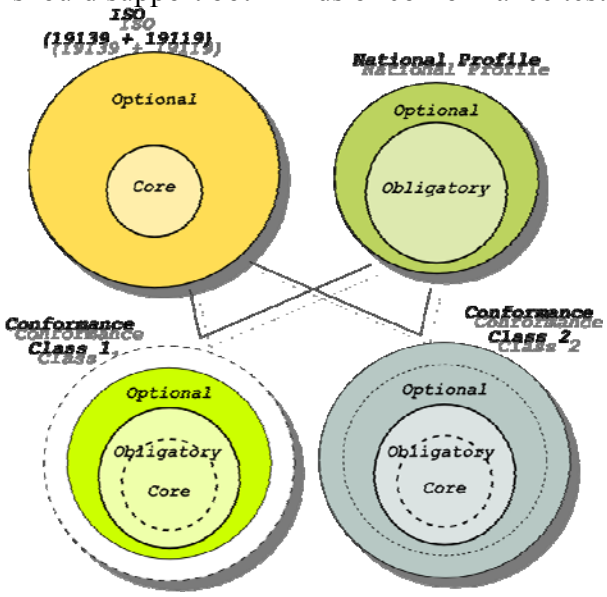

Figure 1. Metadata profile and conformance classes

2.2 Catalog service profile

A catalog service profile might be created on the same basis as metadata profiles, targeting specific implementation communities. By assumption the general interface model of the catalog service is a platform-neutral description of catalog operations while each application profile is platform-specific. It means that an application profile makes use of one of the protocol bindings defined in the catalog specification.

An XML metadata editor supporting a client interface to the catalog service must implement proper service profile.

2.3 System architecture

Catalog services expose their interfaces to the service bus. The specification of the CSW interface includes the harvestResource method and therefore the catalog services can be logically connected in various ways – some services may serve as sources of metadata for other services. Resulting topologies theoretically could be quite complicated (with deep nesting and loops), but practically they are in most cases of the tree form.

Harvesting can be done against a catalog service or against a single metadata file prepared with the aid of an XML metadata editor and published in the Internet. To address the specific metadata resource, Uniform Resource Locators (URLs) are used.

3. Metadata editor MEE

The metadata editor MEE was designed as a desktop application offering a GUI, by which user can edit metadata elements in a user friendly way. At first the editor was only a simple application, without any communication abilities, but later was augmented with a partial implementation of a catalog service client. The screen shot of the MEE interface is shown in Figure 2.

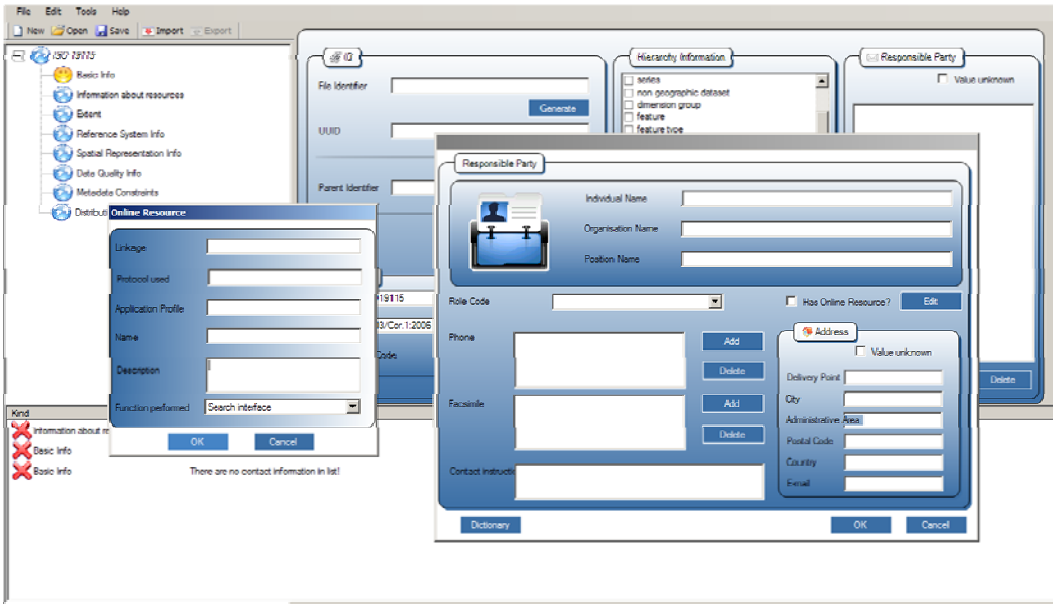

Figure 2. Screenshot of MEE user interface

3.1 Functionalities

The list of functionalities offered by the MEE editor consists of (see Figure 3):

- XML metadata files edition (supported by the use of dictionaries; concerning definitions of metadata elements according to ISO 19119 and 19139 standards and national profiles; supporting such file operations as: create, load, save, and import from a catalog service – export to catalog services is not implemented yet);
- XML files validation on XML metadata file loading and while editing (which includes: XML Schemas based validation – for syntactic file checking; and implementation based validation – where XML file structure and content are checked together with restrictions and conditional statements of the profile used);
- batch file processing (possibility of doing editing operations on multiple files at once);
- metadata series edition (possibility of filling in metadata elements with data taken from a series);
- templates use (the possibility of using metadata files templates – it is not yet implemented).

Metadata elements are presented to the user in an organized manner, collecting all logically dependent elements in one place. The editor provides the ability of quick and easy operating on many files, even hundreds, with a selected profile. The most interesting functionality from the point of view of the editor's use within a metadata infrastructure context is the ability of communication with catalog services (see Figure 4). At the current stage the editor can get the data from a service using CSW ISO AP, but not only one file, but also multiple files being part of defined series. This functionality was tested on deegree CSW server implementation.

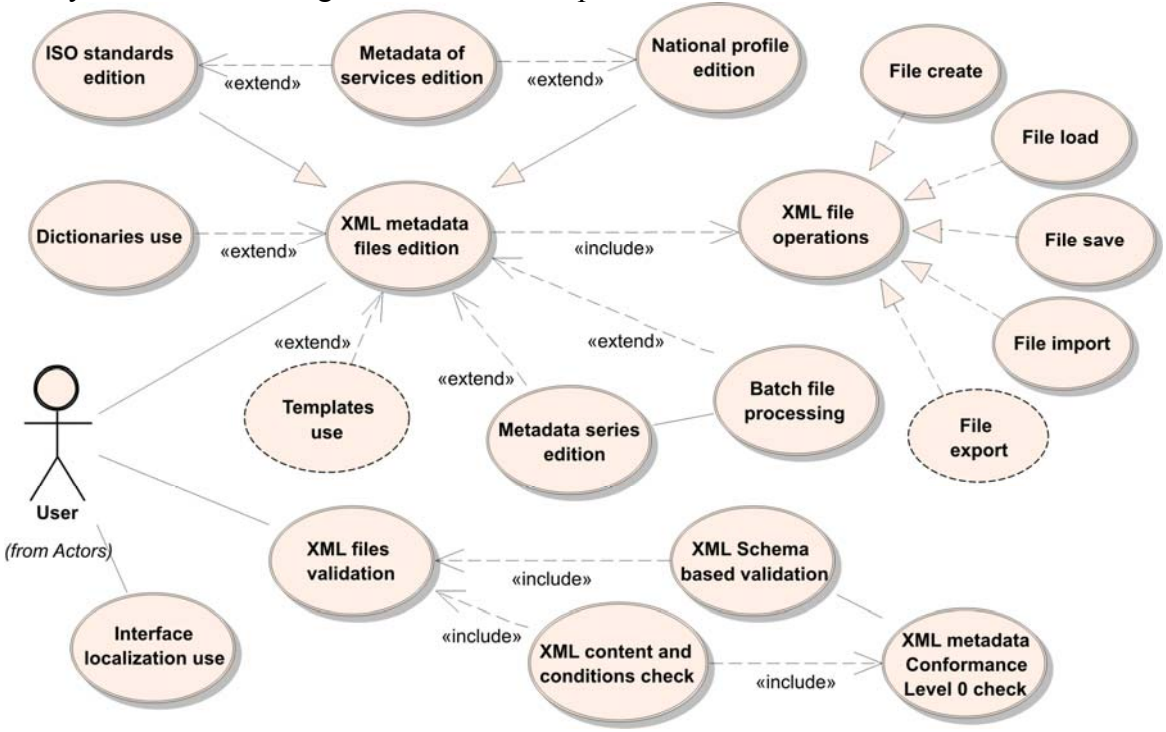

Figure 3. Functionalities of MEE

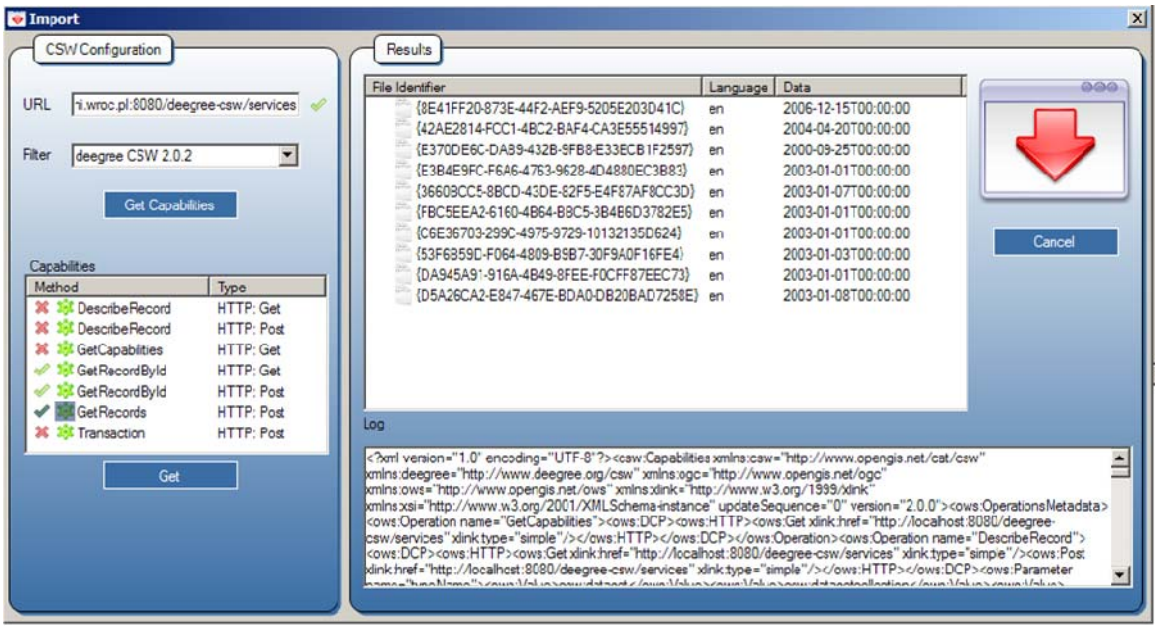

Figure 4. Screenshot of the import from catalog service form of MEE

3.2 Implementation details

The priority was to achieve high efficiency, simplicity of use, and well structured code, open for extensions, flexible and easy to manage. This goal was reached by applying object oriented design programming paradigm and using a few design patterns. The main pattern of MEE application was Model-View-Controller (MVC). The MVC pattern is a widely used sample in GUI frameworks and envisages three main components: the model, the controller and the views. It is used to separate an application's data model, user interface and control logic into three distinct components. In the case of MEE the user interface manages data by using controller functions and events. The controller communicates with the model and sends back information to the view. Not significant, but very helpful in memory economization was the "Singleton" pattern. Essentially, a singleton is a class which only allows a single instance of itself to be created, and gives simple access to that instance.

The information model of metadata was mapped onto the classes of the C# language. Because of the ISO19139 standard extensive data structure implementation, only one container of data is created on a controller's initialization. It is the same data container either for the ISO standard or the national profile (the only national profile already implemented is the Polish profile). An application does not allow new memory allocation while editing data. At startup the application must only know what metadata standard or profile is to be used, and whether service metadata (as described in the ISO 19119 standard) should be taken into account.

The code of the editor was tested on Windows system, but the tests against the Mono platform on other operating systems are under way. All functionalities were encapsulated in dynamic libraries. For accessing and manipulating XML documents the XML DOM (Document Object Model) was used.

4. Concluding remarks

The metadata editor potentially might not only be a tool for XML file editing. It can be a part of a broader solution, offering user interface to catalog services. It can hide behind its forms some crucial business processes of the spatial information infrastructure, as for example, metadata registration, validation, etc. But the implementation of these functionalities requires, among others, communication protocol implementation and harmonization of different standards. That is why the development of the import/export functionality is a complex task. Developers must consider, for example, the existence of different namespaces that describe input and output data used by different catalog service vendors.

The main functionality of the editor presented was XML metadata files creation, editing and modification in accordance to the ISO standards and national profile specifications. During the implementation of the editor several problems were experienced. These included: imperfection of standards, incompatibilities between existing catalog service implementations from different vendors, and differing interpretations of the metadata model (especially sets and series).

The choice of the implementation platform seems to have been a good one. The use of C# allowed the creation of a solution with a support from a design framework by simplification and automation of the common work with code management. The MVC design pattern applied granted clear code partitioning into functional parts. The technology chosen offers a chance to take advantage of the capabilities of the new Microsoft systems (future systems are planned to be implemented in C#) and provides good memory management. Thanks to Wine and MONO it can also be seen as a potentially multiplatform solution.

The most desired functionalities of a metadata editor are those which can increase the effectiveness and efficiency of metadata creation. The use of series has been implemented in MEE, and the use of templates is on the agenda. In many practical cases, on a customer site, metadata sets already exist but are hidden inside resources encoded with different GIS formats. Thus the information extraction or format translation should be in focus of the XML metadata editor when extending its functionalities.

The XML metadata editor described in the paper is accessible thorough the site: http://sourceforge.net/projects/metadataeditor.

This work was supported in part by a grant from the Polish Ministry of Science and Higher Education number 4T12E 010 30 realized at the Institute of Geodesy and Geoinformatics, Wroclaw University of Environmental and Life Sciences, Poland.

References

Bernard, L., Kanellopoulos, I., Annoni, A. & Smits, P. (2005), 'The european geoportal - one step towards the establishment of a european spatial data infrastructure', Computers, Environment and Urban Systems vol. 29, no. 1, pp. 15–31.

Maguire, D. J. & Longley, P. A. (2005), 'The emergence of geoportals and their role in spatial data infrastructures', Computers, Environment and Urban Systems, vol. 29, no. 1, pp. 3–14.

Nogueras-Iso, J., Zarazaga-Soria, F., Béjar, R., Álvarez, P.&Muro-Medrano, P. (2005), 'Ogc catalog services: key element for the development of spatial data infrastructures', Computers and Geosciences, vol. 31, no. 2, pp. 199–209.

Nogueras-Iso, J., Zarazaga-Soria, F., Lacasta, J., Béjar, R. & Muro-Medrano, P. (2004), 'Metadata standard interoperability: application in the geographic information domain', Computers, Environment and Urban Systems, vol. 28, no. 6, pp. 611–634.

Tindall, C. I., Moore, R. V., Bosley, J. D., Swetnam, R. D., Bowie, R. & De Rudder, A. (2006), 'Creating and using the urgent metadata catalogue and thesaurus', Science of the Total Environment, vol. 360, no. 1-3, pp. 223–232.

OGC 07-006r1, OpenGIS® Catalog Services Specification, version 2.0.2

OGC 07-045, OpenGIS® Catalog Services Specification 2.0.2 - ISO Metadata Application Profile

ISO 19115/Cor.1:2006, Geographic information – Metadata, Technical Corrigendum1

ISO/TS 19139:2007, Geographic information - Metadata – XML Schema