Dynamic Server Map System

Martin Mráz

Katedra informatiky, Fakulta elektrotechniky a informatiky, Vysoká škola báňská - Technická univerzita Ostrava, 17.listopadu 15,
708 33, Ostrava-Poruba, Česká republika,
martin.mraz.sk@gmail.com

Abstrakt Cílem diplomové práce je návrh a realizace manažmentu dynamického mapového servra v technologii ASP.NET (C#) s použitím datového úložiště MS SQL Server. Na samotném začátku je nevyhnutné vykonat analýzu současného stavu technologií a standardů používaných v GIS sféře a publikování prostorových dat. Porovnání a výběr vhodné enterprise technologie je jeden ze základních kroků pro další fáze vývoje. Nejdůležitější části a logika systému jsou representována diagramy UML. Po samotné implementaci by měla následovat poslední část cyklu a to testování, ověření stability a nasazení prototypové aplikace.

Klov slova: dynamický mapový systém, ASP.NET, GIS, prostorová data, UML

Abstract. The aim of the work is design and execution of dynamic map server system management in the ASP.NET (C#) technology using the MS SQL Server as a data store. At the beginning of the work it is necessary to provide current situation analysis of the state of technologies and standards used in the GIS sphere and spatial data publishing. Comparison and choice of appropriate enterprise technology is one of the basic steps for the further developing phases. The most important parts and the main logic of the system are represented by UML diagrams. After the whole implementation part is done, the stability verification and testing of created prototype application should follow as the last portion of the software cycle before its deployment.

Keywords: dynamic map server system, ASP.NET, GIS, spatial data, UML

1 Introduction

The project is focused on the concept of design and system implementation for the complex GIS application management which should be used in a company or an organization. Today, there are only a few solutions (that can be found) at the market. Most of them are technologically outdated or too specialised in

concrete functionality. Suggested solution would represent a common platform - universally utilizable and applicable in any field of human activity.

Before the interface implementation, it is necessary to carry out a close analysis of current state of used technologies and standards in the GIS and spatial data publishing. In other words the goal is to gain a clearer idea of the requests for designed system. It is required to create usable solution of a good quality, thus after the analysis the modeling of several diagrams will follow. It has become a custom to use the UML as the framework for the diagrams. The basic ones will include Class diagram, Use Case diagram and Deployment diagram.

Following list of requirements gives better idea about the basic tasks of this work:

- Create an individual structure of the objects that appear in existing map applications (layers, compositions, styles, ..).
- The ability to manage several "real" projects at one data storage supplies.
- The possibility to assign a definitions which data are relevant to a project.

An important part of the system is to create a simple and attractive graphical user interface that will provide an elegant approach to its functions. Being a web application, as the developmental technology was selected ASP.NET and C# using Microsoft Visual Studio 2008. Data layer will be ensured by MS SQL Server.
It is good to mention here that the technologies have changed from the primary intention to use an open-source Java EE, EJB with PostgreSQL.

2 Current situation analysis of the state of technologies and standards used in the GIS sphere and spatial data publishing.

This section brings the OWS standards overview, map servers description and relative technologies widely used in the GIS sphere.

2.1 OGC Standards

All mentioned standards in this subsection are covered by the Open Geospatial Consortium (OGC) which is an international industry consortium participating in a consensus process to develop publicly available interface standards.

OGC Web Service (OWS) Web Service defined by the OGC offers access to the system functionality over the network like the other web services. The main difference is that the OWS has a strictly defined and described set of the operations which can be executed and called[**?**]. The OWS as a set of the specifications contains for example the WMS, WCS, WFS and other services.

These standards are accessible on the OGC web pages. Common properties for all OWS are defined within the OGC Web Services Common Specification[**?**].

The architecrure of the OWS doesn't differ very much from the other web service architecure. The communication between the service provider and user is based on the usage of the communication protocol (mostly HTTP) with the XML serialization.

Web Map Service (WMS) Web Map Service Interface Standard (WMS) provides a simple HTTP interface for requesting geo-registered map images from one or more distributed geospatial databases. A WMS request defines the geographic layer(s) and the area of interest to be processed. The response to the request is one or more geo-registered map images (returned as JPEG, PNG, etc.) that can be displayed in a browser application. The interface also supports the ability to specify whether the returned images should be transparent, and whether the layers from multiple servers can be combined or not.

Web Feature Service (WFS) Web Feature Service (WFS) request consists of a description of the query or data transformation operations that can be applied to one or more features. The request is generated on the client and is posted to a web feature server using HTTP. Then the web feature server reads and consequently executes the request. Web Feature Service allows the client to retrieve the geospatial data encoded in GML (Geography Markup Language). GML is built on the standard web language XML. A WMS can exist without WFS, but to provide full querying and data retrieval at the attribute level, WFS must be employed.

Web Coverage Service (WCS) Web Coverage Service Interface Standard (WCS) defines a standard interface and operations that enable interoperable access to geospatial "coverages"[**?**]. The term "grid coverages" typically refers to the content such as satellite images, digital aerial photos, digital elevation data and other phenomena represented by values at each measurement point. WCS also provides access to potentially detailed and rich sets of geospatial information, in a form that is useful for client-side rendering, multi-valued coverages, and input into scientific models and other clients.

2.2 Map server technologies

Spatial data visualization and publishing is realized by a map server in the web environment. The communication between the client and the map server is based on the client-server architecture. Required parameters are provided by a web server in the cooperation with a map server. Response file, as the result of this collaboration contains required map or a command outcome.

We can divide current map servers into three groups:

- Commercial application servers
  - ArcIMS (ESRI)
  - Geomedia Web Map (Intergraph)
  - MapXtreme
  - MapGuide (Autodesk)
- Commercial maps API
  - GoogleMaps
  - Yahoo Maps
- Noncommercial (open-source) application servers
  - UMN MapServer
  - Deegree
  - GeoServer

This projec is focused on a noncommercial solution therefore open-source application map servers and their main characters are described in the next few subsections. Open-source solutions was selected because of company's needs to have a map server as a free solution under the GNU GPL licence.

3 Comparison and choice of appropriate enterprise technology

This section contains all requirements desired for the map server, table of the advantages and disadvantages, list of the deficits and final decision of chosen solution.

3.1 Requirements

The list of the map server's requirements defines some needs which are necessary for this project. Chosen application server has to provide all of this items:

- Open-source licence as a first of all desired conditions means the possibility to develop and extend existing solution. An open-code, free distribution, interested communities of developers, all of these attributes put this type of license to the top.
- Enterprise application - complex solution with the administration through the GUI and http interfaces is expected.
- API of the technology is required.
- OGC standards support as a foundation stone secures proper access and application of the GIS standards.
- Spatial database support in the enviroment, where the map elements as a layers, points, compositions are used, belongs to the fundamentals for the map server.

3.2 Comparison table

The evaluation of the advantages and disadvantages for the most interesting open-source map servers are contained within the table. The categories, which the servers were compared in, contain: offered license, used technology, available properties of WMS/WFS/WCS/CSW/GML services and database support.

This summary provides clear view of all pros and cons for fast reference and suggests which solution could be used as a map server in this work.

3.3 GeoServer as a right choice

It was decided to use GeoServer as the application map server for this work because it covers all requirements defined above. It is based on a Java technology and supports the OGC standards for manipulation and publishing the spatial geodata. Its enterprise architecture allows the communication and administration through the WEB and REST interface (as an extension).

The WFS Transational support as one of the advantages extends WFS services for LockFeature operation and Transaction(insert, update, delete).

An effective cooperation with dynamic data visualisation provided by OpenLayers library makes GeoServer very strong and effective solution.

3.4 Deficits and requirements

The GeoServer is a very sophisticated solution, however, many important properties and abilities are missing here. This part is sold out by next section. Following list clearly shows missing requirements:

- No project recognition. Current GeoServer contains the namespaces or workspaces, but this features don't restrict access to the resources. Required feature should control the access to the same data store for all projects and define which resources are accessible for concrete project.
- Only basic user and role management. Final solution has to have the option to specify a limited set of users for the projects.
- No built-in REST user interface for http requests.
- No possibility to create various map compositions from external map servers.

GeoServer offers a communication interface, so it is possible to work with the settings of the map layers, points, etc. Final application of this work should be used as a centralized map system management served as an access point in communication between the projects and GeoServer.

Centralized map system management must be an object-oriented solution, therefore all data structures, layers, points, compositions, users, projects and other elements will be described as the objects with appropriate relations between them.

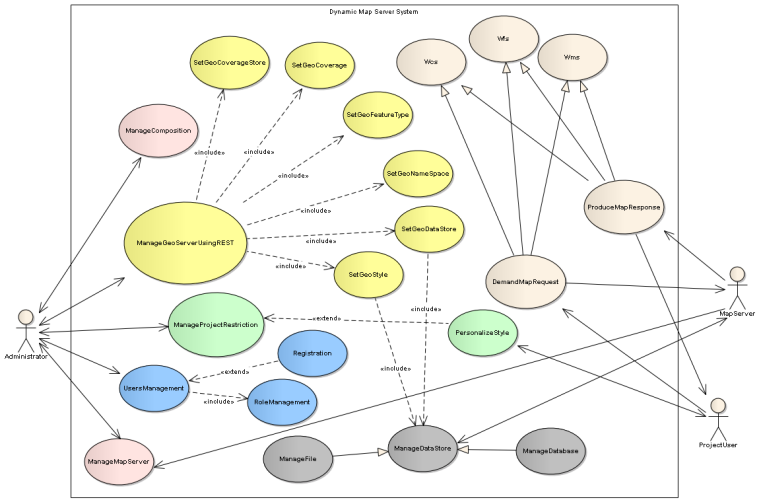
4 Design of the system in the UML language

First of all, it is necessary to understand the user requirements. After this step, when a designer understands required needs, it follows the design sketched by the UML diagrams.

4.1 Requirements

The analysis of the requirements is the process of understanding the customer expectations from a proposed system. This part is divided into functional and non-functional requirements.

Functional requirements The management of requirements is a systematic way to capture, understand and organize changing requirements of the system. The Use Case diagram was selected as the best approach to provide this very important task. Another part of UML, the User Stories, clarifies important actors and their roles in this scheme.



Obrázek 1. Use Case Diagram

Primary Use Case [1] describes all important requirements using 3 main actors: *ProjectUser, Administrator* and *GeoServer (as a map server).*

The idea of the various colour grouping is used for a better orientation inside this diagram. The use cases are grouped in the same colour group when they logically belong together, i.e. use cases: *ManageFile* and *ManageDatabase* are

covered with the same colour as the *ManageDataStore* because they are derived from it.

Extended diagram shows a detailed view on the generalization between actors and the << *use* >> relation between use cases. Next detailed view of the composition setting is showed on another diagram.

4.2 Class model

Class Diagram describes static system structure, objects and relations between them. Objects are represented by the classes with the attributes and methods. The diagram, included in the appendices, is divided into two halves because of its quite big size. Left part and right part model actual state of the system.

The extended version presents the idea of complex system model with the enviroment and planned surroundings. The various colour grouping is used here again for better orientation inside this diagram and it has the same meanig as in the use case diagram. Constructed class diagram contains some important classes which functions are described inside the text.

Data dictionary Data dictionary belongs to a data analysis and can be considered as a foundation for the information systems where the databases, data storing and searching create a significant part of the whole system functionality. The outcome of the data analysis is, except the others, the data dictionary. It includes the objects represented by the tables and their attributes as a columns that are atomically and indivisible. The relations between the objects are already after the decomposition, where all many-to-many associations are expanded to on-to-many type.

5 Features of the system

Most important features of the system are described by following subsections.

5.1 User controls

For a better orientation and also the fact that there are many web formulars within the bounds of this project, it was necessary to divide them to several user controls. User control usually contains one web control which is already set up and binded to a datasource. But it is nothing special when a one user control contains another controls that logically belong together. After this short introduction and explanation, why the user controls are very useful, it is worth to describe all important user controls. Next subsections are sorted according to the graphic user interface.

Project. The Project is a basic control for user project handling. The main goal of this feaure of the system is the management of existing projects and creating the new ones. Within the project there are specified important restrictions:

1. The name of the project should be as representative as possible in the list of all projects.
2. Allowed users, who will have an access to the system and who will provide the operations on this project.
3. Defined compositions were meant to customize the output of the maps as a response for the client requests.
4. The last restriction is created by the list of available map servers on which the users can send their requests.

In every phase of the project management, the user can see which users, compositions and map servers are selected for current project. The interface is designed to give this information in a very pleasant way.

User and user roles. As it is used in every application where an interaction with the users exists, the user management is very useful. This project contains this feature as well as a part of the main menu items. There is a possibility to create, edit, delete the records including user registration information and user roles.

Each user can be assigned to a several user roles. This statement gives an opportunity to use many-to-many relation between the user and role.

Edit form consists of a tab control with 2 tabs. The first one, User, contains items as a login, password, firstname and surname to provide the editation. The second one, Roles, offers a form where the administrator can select the roles to assign them to currently edited user.

An approach, where it is possible to edit user parameters in one editation, select or deselect the roles and manage the list of accessible roles, is accounted a success and an advantage for the administrator who will provide all the changes.

Composition. The composition part of the system creates with the project restrictions management the most required functions of this dynamic server application. Inside this feature the administrator can manage selected layers which will be used and displayed within edited composition. The tree of the layers, which are parsed from a selected map server included in the list above the tree.

As you can see, there is expanded parent layer containing child layers. All of this items can be selected and assigned to a composition. In this case, all styles related to a selected layers are assigned to the collection of the composition styles. Next version of this system will show an advanced option on how to selecet the styles from the list of each layer.

Map server. The tab devoted to a map servers offers a complete control over the used servers. Map server as a foundation stone of this application contains, except simple parameters as a url and name, one special function called *Layer refreshing*.

This feature is provided by a web service, therefore it is called asynchronous. There are two possibilities, how to call this service:

- From a client script, when a metada attribute is required in the definition of the web service class.
- From a client script using callback, which is subsequently processed on the server side.

When the web service is called, a loading panel informs the user about the actual layers' refreshment. After the opertion is completed, the loading panel disappears and the user can continue in the interaction with the system. This functionality has been tested with a several types of map servers (GeoServer, MapServer, ..) and no error occured during these tests.

The main goal of this function is to parse and persist the layers and their styles to the database. The method is designed, using the recursion, to parse the whole hierarchy of child layers up to unlimited number of levels.

REST. The GeoServer was selected as a primary map server for this project for its many advantages. REST as one of them contributes in a positive way to its selection.

The REST architecture, as an architectural tool for building large-scale distributed hypermedia systems[**?**], is based on four statements:

1. Requested resources are identifiable through the URI, based on a global addressing space.
2. Interactions through hyperlinks are labeled as a stateful, which means that the interaction is based on the concept of explicit state transfer.
3. Resources are self-descriptive by attached metada. The content of the resource may have various format, therefore the metada are useful to detect the errors or to perform the authentication.
4. Access interface is defined by the following http methods: POST, PUT, DELETE and GET. Each method is used to perform different operation on selected resource.

6 Further system development

To use this dynamic server map system as a full-value solution, it is required to implement the part of client side. Client application will be installed and deployed at the customer side. The communication between the client and the

server will be handled by the web browsers over the internet.

Web client, as the result application, should cover all required functions and needs defined by the customer. Some of these requirements are described by the following paragrahps.

Intuitive ability to create the HTTP requests (WMS, WCS, WFS), which will be processed on the server side, and the map responses or appropriate results then will be sent back to the client. The request can have some parameters like desired response map format. All parameters must be set before the request will be forwarded to the server.

Another need can be understood as a necessity to have a modern and purpose-built map publishing with a rich amount of the options meant to set and modify the final output..

The way, how the map output can be styled, should be controled by the style management. Map server offers many layers with defined styles for them. All projects, which use similar layers within the composition, have similar style settings. To make the map outputs closer to costumer desired idea, the style management handles the issue of this kind of personalization.

As the dynamic server map system has many operations provided by the AJAX technology, so the client application will use this kind of scripting as well for the dynamic content refreshing.

The system authentication is required to secure the application before unauthorized access to a sensitive information. The process of indetity identification should prove that the user is the claimed one.

7 Conclusion

This work was focused on the concept of design and system implementation for the complex GIS application management. The solution should be able to play a main role in a company or an organization.

At the beginning it was necessary to provide the analysis of the current state of technologies and standards used in the GIS sphere and spatial data publishing. The solutions which could be findable on the market were technologically outdated or too specialised in concrete functionality. Suggested solutions should represent a common platform - universally utilizable and applicable in any field of human activity.

The OGC consortium provides and publishes GIS standards and all its important specifications are mentioned and described inside this work. The WMS, WCS and also WFS form a set of services, which every high-quality GIS application should use as a ground for the GIS components integration. There are several players on the market concerning the map servers. Some of them are

available for commercial usage, the rest is published under the LGPL license. Higher attention was devoted to the open-source solutions.

The GeoServer was chosen as a best map server because it filled all needed requirements. Its geospatial features and possibilities, in comparison with other servers, met all defined criteria. However the GeoServer is a very sophisticated solution, many important properties and features are still missing. These deficits should be solved by the Dynamic Server Map System. Design of the prototype was described by the UML diagrams and specifications. Use case diagrams described the surroundings of the system. All functions and responsibilities for the actors were decided. User stories with their high-level definition of a requirements produced for the developer an efficient model for its further implementation. The description of the static system structure, objects and relations between them was covered by the class diagram. This model was selected as the most efficient tool.

Before the whole implementation, the technologies have changed from the open-source to the commercial products. The ASP.NET with C# in the backend were choosed instead of the Java and Enterprise Java Beans framework. Planned IDE was changed as well, from the NetBeans to Visual Studio 2008. The impact of this change was minimal on the system because the design was created as an indenpendent part of selected technology.

Implementation phase came out from the design and completed analysis. The outcome was the prototype of the application, ready for testing and verification. Extensive part of this section was devoted to the database control there. Almost all important techniques of handling the database operations are described with a lot of code listing examples.

One of the last steps in the system, the deployment part, provided installation of all components related to the prorotype application. System integration and its technical assessments brought the final satisfaction feedback of using the prototype.

Reference

1. Ožana R., *Posouzení vlastností Geonetwork opensource a jeho uplatnitelnosti pro úeely národního metaportálu.*, Diplomová práce VŠB-TUO. 103 stran., 2007.
2. OGC, Web Service Common Implementation Specification, *Web Service Common* [online].[cit. 2009-11-11]. Available from WWW: <http://www.opengeospatial.org/standards/common>
3. OGC, Web Service Common Implementation Specification, *Glossary of Terms - C* [online].[cit. 2009-11-11]. Available from WWW: <http://www.opengeospatial.org/ogc/glossary/c>
4. Pautasso C., Zimmermann O., Leymann F. *RESTful Web Services vs. "Big" Web Services: Making the Right Architectural Decision.*, Beijing, China. 2008.