Web Service Based on GeoTools in The Atlas of Fire Protection

Jan Růžička
Institute of Geoinformatics
VSB – TU of Ostrava
`jan.ruzicka vsb.cz`

Keywords: WebService, GeoTools, atlas, fire protecttion

Abstract

The paper describes a simple way of a systems integration based on SOAP web services. The systems integration described in the paper is covered by a system named The atlas of a fire protection. The atlas is a set of dynamically created maps published in WWW browser. Technologies used for the solution are UMN MapServer, ArcIMS, PHP, GeoTools and Axis. GeoTools and Axis are used for building platform and programming language independent component for the atlas. The paper describes a software architecture used for the atlas and a role of a web service in the integration.

Introduction

It was a few years ago when my colleague ask me if there is a way how to integrate external component for the atlas that allows basic data classification. The atlas was completely build on PHP and PHP/MapScript. We could not find any PHP based component for this purpose. We have found a Java based component GeoTools Lite. There were a few ways how to integrate this component to the atlas. After some research we have found the way of web service based on SOAP (Simple Object Access Protocol) the most flexible and independent. Our decision was probably right and after a few years when the system was rebuilded to support ArcIMS and other tools, the only part based on GeoTools stayed without any change. A month ago one Turkish colleague ask me about documentation of GeoTools used for building web services. Unfortunately I did not have any at that time. I have decided to write this paper to fill this gap on the Internet. I hope that somebody will find this paper useful.

The atlas of a fire protection

The atlas development started in 2004 year with support of General Directorate of Fire Rescue Service of the Czech Republic. The atlas is a web based interactive application that enables a map creation based on user specified conditions. The atlas is based on a statistical database about incidents in a scope of fire brigade field of action. That's why there are two basic ways of thematic map creation:

- Choropleth maps.
- Diagram maps.

The proces of choropleth map definition starts with specification of the following conditions:

- interval of time from/to of events,
- incident type (e.g. fire-fighters injured during interventions),
- statistical method for generating class intervals (Jenks, Equal interval, etc.),
- number of classes,
- type of frequency (square km, population),
- colour scale for classes visualization.

An example of a possible result of the choropleth map creation is figured at fig. 1.

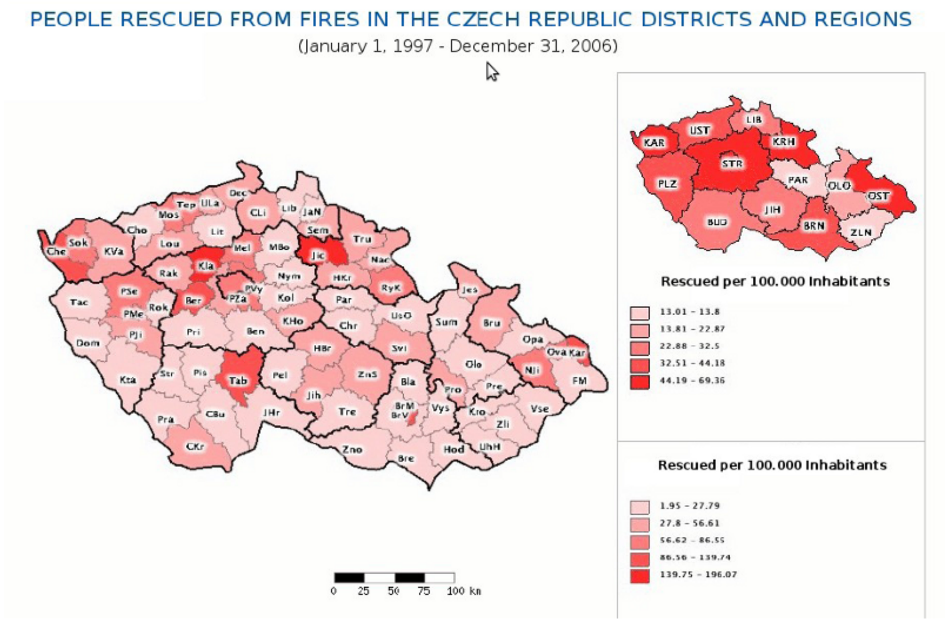

Figure 1: Choropleth map, Screenshot from Peňáz 2009

The statistical method

The atlas was based on UMN MapServer resp. on PHP/MapScript library. Authors of the atlas could not find any method for data classification available in used library. There were two ways how to solve this problem:

- implement algorithms,
- use external component.

The external component has been found in spring 2004. At that time was named GeoTools Lite 1.5.

GeoTools Lite

GeoTools Lite 1.5 library has been used for the atlas. GeoTools Lite 1.5 library is not nowadays developed. Developers can not download it from any official resource. There is available GeoTools Lite follower currently named GeoTools 2.

GeoTools 2

The GeoTools is a set of libraries written in Java language. The main features of GeoTools library are [1]:

- OGC Grid Coverage implementation,
- Coordinate reference system,
- Symbology using OGC SLD specification,
- Attribute and spatial filters using OGC Filter Encoding specification,
- Graphs and networks,
- Java Topology Suite (JTS),
- Two renderers,
- Raster and vectors formats.

GeoTools Lite 1.5

GeoTools Lite 1.5 is a library with basic features necessary for geodata manipulation. It consists of modules for:

- Geodata reading (raster and vector basic formats),
- Desktop rendering (including diagrams),
- Projections,
- Map algebra,
- Data classification.

GeoTools Lite 1.5 data classification

For data classification are available three methods:

- Equal Interval,
- Natural Breaks (Jenks),

- Quantile.

Why not GeoTools 2?

The reason why was not used the richer library GeoTools 2 instead of GeoTools Lite 1.5 is a quite simple. The module for the classification was not included in GeoTools 2 at the time of the implementation.

Way of integration

The atlas was based on PHP language and the library was based on Java language. There were three possible ways how to integrate the atlas with the GeoTools Lite:

- Use a source code of GeoTools Lite 1.5 and rewrite it in PHP,
- A native link of the Java library from PHP using JNI (Java Native Interface),
- Write a software component based on the GeoTools Lite 1.5 where the communication is based on an other way than JNI.

We believed that classification module will be included in the GeoTools 2 and that a future development of the module is expected at the time of the integration. That is why a rewriting of the code into PHP seemed as a wrong way. The technology named JNI and its usage in PHP was (and still is) experimental and we have not any experience with this technology at that time. We have started our research of SOA (Service Oriented Architecture) in 2003 and at the time of the atlas implementation it was a hot topic of our research (and it still is). That's why we have decided to use web services as a solution for building independent library based on GeoTools Lite available from PHP.

Web Services

Web services can be simply characterised with the following facts.

- Software components.
- Available via web (based on HTTP).
- Simillar to RPC (Remote Procedure Call).
- Based on Request / Response.
- Usualy based on XML (SOAP), but may be more flexible (REST) or simpler and hard-coded (WMS).

GeoWeb Services

GeoWeb services are services that we are dealing (delivering, manipulating, analysing) with geodata. There are several specifications for building GeoWeb services. Well known are for example:

- Web Map Service.
- Web Feature Service.
- Web Coverage Service.
- Catalogue Web Service.

Classification

The Carto Support Service (CSS) was written as SOAP based web service with four public methods:

- getEqualIntervalBreaks
- getNaturalBreaks
- getQuantileBreaks
- getCapabilities

The first three of the methods are used for classification. All of them have the same arguments: string that contains numbers (float or integer) delimited with semicolon and integer that holds number of requested classes. The methods returns string that contains class breaks delimited with semicolon. The fourth method returns XML with the service capabilities description. The description is based on OGC WMS GetCapabilities response. We had to deal with problems with SOAP implementation in PHP in the area of encoding of arrays and lists, that's why the encoding was simply based on text delimited using semicolon.

The SOAP interface for the CSS is based on Axis Library (Apache Foundation). Axis Library and the CSS is hosted on a Tomcat servlet container (Apache Foundation). For web service deployment is used simple autodeploy mechanism that is offered by Axis technology:

- A source code of a service is copied into directory specified by Axis with jws suffix in a name of the file.
- When a Tomcat servlet container is restarted Axis servlet checks content of the directory.
- If there is a change of the source code or if there is a new file, the Axis servlet performs compilation of the source code and deployment.

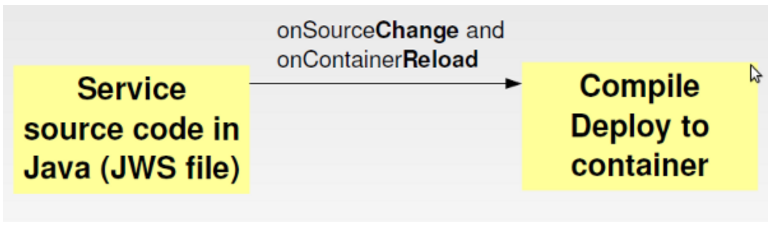
onSourceChange and
onContainerReload
Service source code in Java (JWS file)
Compile Deploy to container

Figure 2: Autodeploy mechanism (JWS)

The source code

The source code of CSS is simple, because it uses GeoTools Lite library and defines only interface for communication.

```
import uk.ac.leeds.ccg.geotools.classification.EqualInterval;
import uk.ac.leeds.ccg.geotools.classification.NaturalBreaks;
import uk.ac.leeds.ccg.geotools.classification.Quantile;
import uk.ac.leeds.ccg.geotools.classification.SimpleClassifier;
import uk.ac.leeds.ccg.geotools.SimpleGeoData;
import java.util.Hashtable;
import java.util.StringTokenizer;
import java.io.*;

public class CartoSupport {
    public String getEqualIntervalBreaks(String data, int breakscount) {
        EqualInterval ei = new EqualInterval(new SimpleGeoData(getHashtable(data)), breakscount);
        return getBreaks(ei);
    }
    public String getNaturalBreaks(String data, int breakscount) {
        NaturalBreaks nb = new NaturalBreaks(new SimpleGeoData(getHashtable(data)), breakscount);
        return getBreaks(nb);
    }
    public String getQuantileBreaks(String data, int breakscount) {
        Quantile q = new Quantile(new SimpleGeoData(getHashtable(data)), breakscount);
        return getBreaks(q);
    }
    public String getCapabilities() {
        try {
            FileInputStream fis = new FileInputStream("CartoSupportCapabilities.xml");
            int x= fis.available();
            byte b[]= new byte[x];
            fis.read(b);
            String content = new String(b);
            return content;
        } catch (IOException e) {
            return "Can not read XML for Capabilities from file CartoSupportCapabilities.xml";
        }
    }
    private Hashtable getHashtable(String data) {
        StringTokenizer st = new StringTokenizer(data, ";");
        Hashtable numbers = new Hashtable();
        int i = 0;
        while (st.hasMoreTokens()) {
            numbers.put(new Integer(i), new Double(st.nextToken()));
            i++;
        }
        return numbers;
    }
    private String getBreaks(SimpleClassifier sc) {
        String breaks = "";
        int ct = sc.getBinCount();
        for (int i = 0; i < ct; i++) {
            double br = sc.getBin(i).getUpperExclusion();
            breaks = breaks + br + ";";
        }
        return breaks;
    }
}
```

Testing CSS

Tomáš Peňáz made comparison test on this classification module with ArcGIS classification module. There was (but not significant) difference. The GeoTools Lite classification was confirmed as a suitable classification for the atlas.

The architecture

The architecture is figured at fig. 3. When a user selects data for a visualisation and specifies a number of classes and a method for classification, the atlas requests via SOAP the selected method on CSS. The request contains data and a number of classes. The response contains classes breaks. The breaks are used with PHP/MapScript library to create requested map.

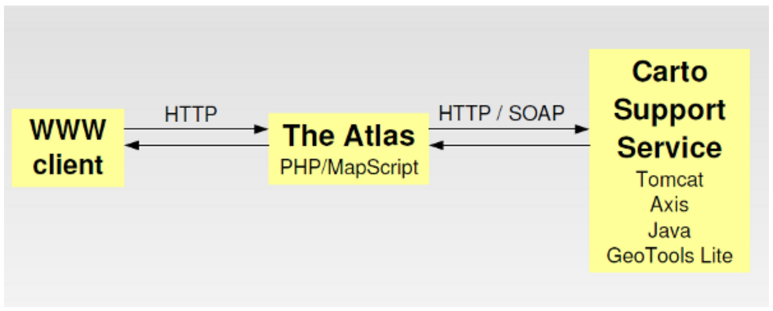

Figure 3: The architecture in 2005 – 2007

Changes in 2007

The system was refactored to be able use ArcIMS technology, this was requested from General Directorate of Fire Rescue Service of the Czech Republic. The architecture in a connection with the CSS stayed untouched. The breaks are used with ArcIMS via an ArcXML request to build the requested map.

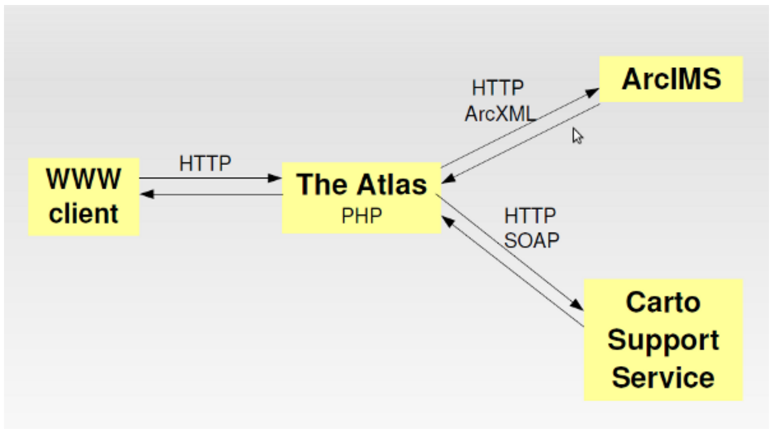

Figure 4: The architecture in 2007 – 2009

Other example

There is described an another example how to use GeoTools for building GeoWeb services in the following chapter.

Simple attribute search

The GeoTools can be simply used for searching in geodata according to geometry or attributes. It is probably more efficient to use some database system (e.q. PostGIS) for building GeoWeb service of this type, but there can be situations where is not allowed to use whole database system. For example in a case when is necessary to optimise your data flow using service migrating mechanism. In that case the GeoTools can be very useful, because the service with data can be simple migrated to another server as one archive file. This example has been prepared for purpose of this paper.

The method getParcelOwner searches in simple ESRI Shapefile according to parcel id specified as the attribute of the method. The method returns name and address of the parcel owner or exception text if there is not match with any id in the file. Names and addresses of owners are simply stored as an parcel text attribute.

```
package cz.vsb.gisak.ruz76.gt.examples;

import java.io.File;
import java.io.IOException;
import java.util.HashMap;
import java.util.Iterator;
import java.util.Map;
import org.geotools.data.DataStore;
import org.geotools.data.DataStoreFinder;
import org.geotools.data.FeatureSource;
import org.geotools.factory.CommonFactoryFinder;
import org.geotools.feature.FeatureCollection;
import org.geotools.feature.simple.SimpleFeatureImpl;
import org.opengis.filter.Filter;
import org.opengis.filter.FilterFactory2;

public class CadastreSearch {
    public String getParcelOwner(String id) {
        try {
            File cadfile = new File("cadaster.shp");
            Map map = new HashMap();
            map.put("url", cadfile.toURL());
            DataStore ds = DataStoreFinder.getDataStore(map); //open shapefile
            FeatureSource fs = ds.getFeatureSource("cadaster");
            FilterFactory2 ff2 = CommonFactoryFinder.getFilterFactory2(null);
            Filter filter = ff2.equal(ff2.property("id"), ff2.literal(id), false); //prepare filter on attribute id
            FeatureCollection col = fs.getFeatures(filter); //run filter to obtain feature collection
            Iterator iterator = col.iterator();
            if (iterator.hasNext()) {
                SimpleFeatureImpl parcel = (SimpleFeatureImpl) iterator.next(); //read first feature from collection
                return parcel.getAttribute("owner").toString();
            } else {
                return "No match";
            }
        } catch (IOException ex) {
            return "IOException";
        }
    }
}
```

Conclusion

WebServices are quite efficient tool to integrate different systems when you work in a synchronous environment with clearly defined interfaces. Every beginer with programming is able to write own web service based on Java, Axis and GeoTools. GeoTools are just one of many available libraries to build GeoWeb services.

References

1. GeoTools (GeoTools 2009)
   http://www.osgeo.org/geotools
2. Peňáz et. al. Fire Protection Atlas of the Czech Republic. Available in intranet only. (Peňáz 2009).

Support

The article is supported by Czech Science Foundation as a part of the project GA 205/07/0797.