MAP DESIGN PROCESS MODELLING USING BPEL LANGUAGE

Jan Růžička
jan.ruzicka@vsb.cz
Tomáš Peňáz
tomas.penaz@vsb.cz
Institute of geoinformatics
VSB-Technical University of Ostrava
17. listopadu 15, 708 33, Ostrava – Poruba, Czech Republic

Abstract

The paper describes a completely new point of view on a process of a map design. You can define a process of a map design as a set of steps that should be done in a specific order. Some of these steps can be repeated and a lot of decisions must be done during that process. This is similar to a business process. There is a goal, there are resources and steps. The paper explains a possibility to handle map creation as a business process and to employ some useful tools for a process modelling. The paper also discusses the possibilities of orchestration technique (methodology) in the process of a map design. An orchestration is a process where the processes (real or abstract) are modelled in the way of a formalized description. A process modelling is a technique that uses several description tools, mainly schemas or diagrams, to describe real processes that are usual inside an enterprise. The processes can lead across several organizations.
The paper shows such a process on an example of Fire Protection Atlas of the Czech Republic in which choropleth maps and diagram maps are dynamically created from the database.

Keywords. BPEL, GeoWeb, orchestration, choropleth map, map production.

1 Introduction

GeoWeb is a platform that consist of standards, open web services, people and technical equipment that can be used for building flexible geographic information systems (GIS). GeoWeb is nowadays at the beginning of the standard usage. There are a lot of standards available, but only several of them are in a common use. To be honest, there is only one standard used in such a scope that can be declared as a common standard. This standard is WMS (Web Map Service). Unfortunately this standard is too simple to base complex GIS only on itself. Of course, there are other standards that are well known, but their usage is reserved to top development companies or to research institutes. Most of this standards are quite old but not worked-out properly yet. They must be proved by the practical usage. There are new standards or technologies growing out every day (like GeoRSS or MapAPI) and we believe that GeoWeb can be the platform for building GIS of the future.

When we started to study the possibilities of the orchestration in the area of GeoWeb there were not any articles describing any theoretical or practical solution. We started our research in 2005. The first practical result from the research was system WSCO (Web Services Catalogue for Orchestration). This catalogue covered several metadata items that could be useful for orchestration. After that we started the analyses of languages for orchestration. We spent more than half a year on the research of WS-CDL language to find out at the end that this language is not suitable for orchestration. Then we obtained the support from Czech Science Foundation to continue our research. During two years of the research we have analysed several languages that can be used for an orchestration. We did several tests and prepared several orchestras.

In January 2009 a research project in the area of intelligent map server has been started and it should utilize a knowledge base. We discussed possibilities of my participation in that project and I learnt that some results from GeoWeb services orchestration project can be used for knowledge based map server. This paper is the first result from our cooperation and ideas from the paper will be proved in the following years.

2 Orchestration

An orchestration is the process within which the processes (real or abstract) are modelled in the way of a formalized description. A process modelling is a technique that uses several description tools, mainly schemas or diagrams, to describe usually real processes inside an enterprise. The processes can lead across several organizations.

Each process has the start point and the end point. There can be inputs from several sources incoming into process and these inputs are manipulated in a time of processing. Some manipulation can be done in an iterative way. A process can be divided into several ways (operations), that can be processed simultaneously. These ways can be selected according to conditions.

For example we can describe the process of flooded areas creation.

1. A customer (a source) defines conditions how should the map look, e.g. what area, what kind of flood (100 year, 50 year, real from 1997 year), what scale should be considered.
2. An analyst (a source) selects appropriate data sources according to the defined conditions.
3. The analyst gets the data sources.
4. The analyst does an analysis and extracts the flood areas.
5. A cartographer (a source) uses flooded areas from the analyst and selects another data sources to create the map.
6. The cartographer draws the map.
7. The map is delivered to the customer for an evaluation.
8. Comments from the customer are evaluated. According to results of the

evaluation the process flow is returned to a new analysis (items 2-4) or the process is returned only to changing the map (items 5-6).
9.
However the process can be more complicated, e.g. it can contain parts such as an agreement, payment, delivery.

A process modelling offers tools for describing such a process in a formal way. The formal description allows to control, test and visualise the process in a form that is intelligible to a common participant of the process.

Two languages are usually used for process modelling, UML (Unified Modelling Language) and BPMN (Business Process Modelling Notation). Each of them offers different possibilities for the description of a process but both of them (or others) can be used for a process description.

A common example of the process description inside an enterprise is ISO 9001 usage. For instance at VSB-Technical University of Ostrava, Faculty of Mining and Geology such a description is published for Ph.D. degree studies process.

If some description is detailed enough, we can start automating these processes or their parts. If an individual task of the process can be solved using a computer, then it is described in a form of algorithm and implemented as a web service (when working in the area of Web SOA).

A model of the process is transformed from abstract languages (BPMN, UML) to the form that can be directly run on a computer. In this area of runnable models of processes there is the most known BPEL (Business Process Execution Language). A process run includes reading inputs, invoking web services, deciding according to results, repeating some parts of the process and other necessary operations.

A process modelling offers possibilities how to formally describe processes inside an enterprise, to find duplicate processes, to find processes that are not optimised, etc. A process modelling helps with processes and sources management optimisation. If it is possible, then the process description is available in a form of BPEL language. So the processes can be directly invoked.

GeoWeb services orchestration can be done in many ways. The GA 205/07/0797 team (team of the project supported by Czech Science Foundation) has researched two ways of possible orchestration. These ways are described in [Růžička 2009].

3 BPEL

“BPEL (Business Process Execution Language) was originally created by Microsoft and IBM. OASIS made a standard from the specification. Current version of the standard is

2.0. The language allows to specify runnable or abstract processes.

BPEL defines a model and a grammar for describing the behaviour of a business process based on interactions between the process and its partners. The interaction with each partner occurs through Web Service interfaces, and the structure of the relationship at the interface level is encapsulated in what is called a partnerLink. The BPEL process defines how multiple service interactions with these partners are coordinated to achieve a business goal, as well as the state and the logic necessary for this coordination. BPEL also introduces systematic mechanisms for dealing with business exceptions and processing faults. Moreover, BPEL introduces a mechanism to define how individual or composite activities within a unit of work are to be compensated in cases where exceptions occur or a partner requests reversal.

BPEL utilizes several XML specifications: WSDL 1.1, XML Schema 1.0, XPath 1.0 and XSLT 1.0. WSDL messages and XML Schema type definitions provide the data model used by BPEL processes. XPath and XSLT provide support for data manipulation. All external resources and partners are represented as WSDL services. BPEL provides extensibility to accommodate future versions of these standards, specifically the XPath and related standards used in XML computation.“ [OASIS 2009].

3.1 BPEL process structure [OASIS 2009]

The process described using BPEL is structured into several sections, the main part of the structure is represented by activities.

<process...>
    <partnerLinks.../>
    <variables.../>
    <correlationSets.../>
    <faultHandlers.../>
    <compensationHandler.../>
    <eventHandlers.../>
    activities
</process>

- PartnerLinks are used for the definition of services and client interface references.
- Variables are used for data sharing between services and clients.
- CorrelationSets are used for identification of process instances.
- FaultHandlers are used for handling errors that can appear during a process.
- CompensationHandler is used for recovery from a fault.
- Events handlers are used for handling events that can occur during a process.

3.2 BPEL process activities [OASIS 2009]

There are more than 20 different activities, that can be used for describing a process. The list of activities includes:
- Receive parameters from client
- Invoke web services
- Reply to client
- Conditions
- Loops

3.3 BPEL editors

A designer can describe process using BPEL in any text editor because BPEL is XML based language. For a comfort work it is recommended to use some kind of IDE (Integrated Development Environment) or a designer especially designed for modelling based on BPEL.
In this kind of environment a user defines a process in a way of constructing a diagram of a process. For simple processes (like in examples in this article) the only way of diagram construction was used. For more complicated processes it is usually necessary to write some parts of code, mainly for transformation of variables.
For the code creating several tools are available. We have a real experience with two environments (NetBeans IDE and Oracle Jdeveloper) but several others were tested as well.

Examples for this paper were made in NetBeans IDE 6.5.

4 The Fire Protection Atlas of the Czech Republic

"The idea to create the Fire Protection Atlas of the Czech Republic comes from 2000. The initial stimulus was the detailed acknowledgement of the data from the statistical incidents investigations and data gathering (in Czech SSU). The most of information about objects or incidents connected with the fire-brigades activities have spatial character (interventions localities, dislocation of sources and devices, area regionalization, interventions statistics, etc.) as they join each event with the particular locality (address, measuring or the place description in words) and also with the area administration unit (a community, a district and a region). The data carrying these kinds of information has got a spatial character and so there is a possibility of taking spatial data visualization as an appropriate method using the thematic map." [Atlas 2009].

"The content and structuring of the designed electronic version of the Atlas were prepared within the Atlas macquette and published in a form of WWW in 2001. In 2004 that former Atlas content and the way of its publishing were modified according to the requests of the staff of the General Directorate of Fire Rescue Service of the Czech Republic. The Atlas structuring, known from the design, is determined by the group of

readers for whom the Atlas is created. The clear and easy structure helps even to non experts a lot while working with the Atlas. As to the content the designed Atlas of the fire protection will be formed by two parts that correspond with two basic approaches within the fire brigades system: organization of the fire brigades system, fire brigades intervention." [Atlas 2009]

4.1 Choropleth map

User of the atlas can use simple user environment (available via WWW browser) to specify conditions for map design.

In the first step the user specifies parameters for selection of events from the database:

- interval of time from/to of events (in a form of MMDDYYYY),
- incident type (e.g. fire-fighters injured during interventions).

In the second step the user specifies conditions for map design such as:

- statistical method for generating class intervals (Jenks, Equal interval, etc.),
- number of classes,
- type of frequency (square km, population)
- colour scale for classes visualization

After the second step the user obtains a picture that contains choropleth map similar to the following picture (fig. 1).

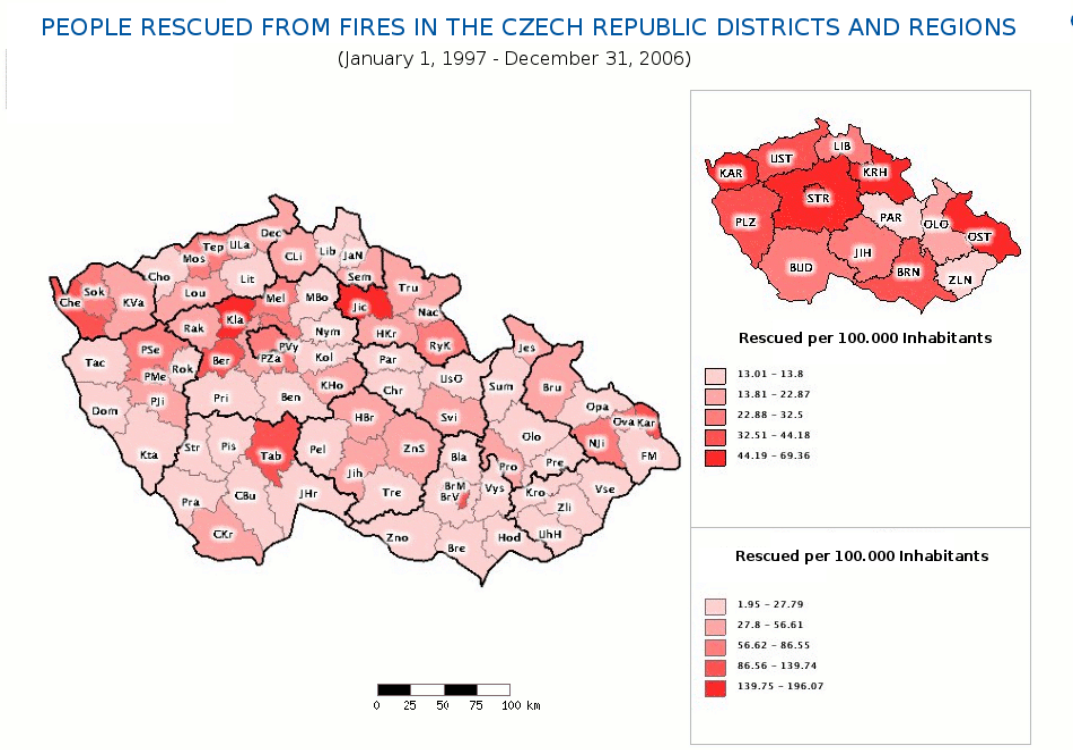

Fig . 1 Choropleth map created by user using the atlas system [Atlas 2009]

4.2 BPEL usage for choropleth map creating

To describe possibilities of BPEL modelling in the area of cartography we have prepared two examples based on the process of choropleth map creating.
The first example is based on a current situation of user's map creating in the atlas. This example is modelled using asynchronous process modelling technique.
The second example is based on the future possible stage of the atlas where the main role should be played by an intelligent system using a knowledge base. This example is modelled using synchronous process modelling technique.

4.3 Asynchronous process

The process is done in two sets of steps. At the beginning of each set the user input is required. This kind of a process must be modelled as asynchronous process. We have defined interfaces for the process steps and user interface.

The process is shown on fig. 2. The left side of the diagram is reserved to a client. The right side of the diagram is reserved to web services. The middle part of the diagram includes a sequence of the process steps.

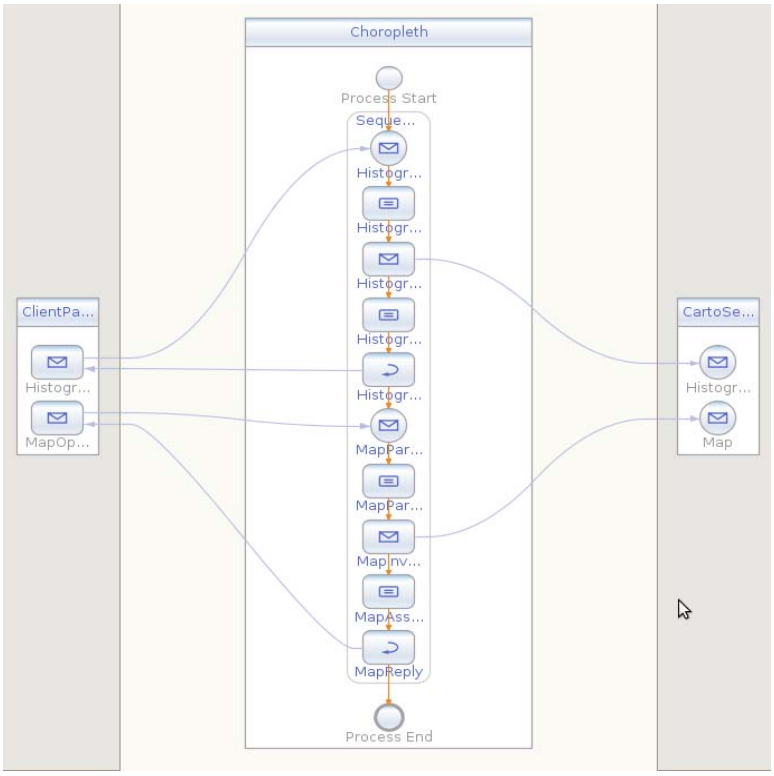

Fig . 2 Diagram of asynchronous process of a map design

The process is divided into the following steps. Steps marked by = sign are used for manipulating with variables and are excluded from the following list.

1. Receive the first input from the user that is necessary for a histogram creation.
2. Invoke web service that generates the histogram.
3. Reply the histogram to the user.
4. Receive a second input from the user that is necessary for the map creation and is based on user's evaluation of the histogram.
5. Invoke web service that generates a map.
6. Reply the map to the user.

Full list of steps, partner links and parameters of the process is described in the appendix A.

4.4 Synchronous process

The process is done in one set of steps. We have defined interfaces for the process steps and user interface.

The process is figured on fig. 3.

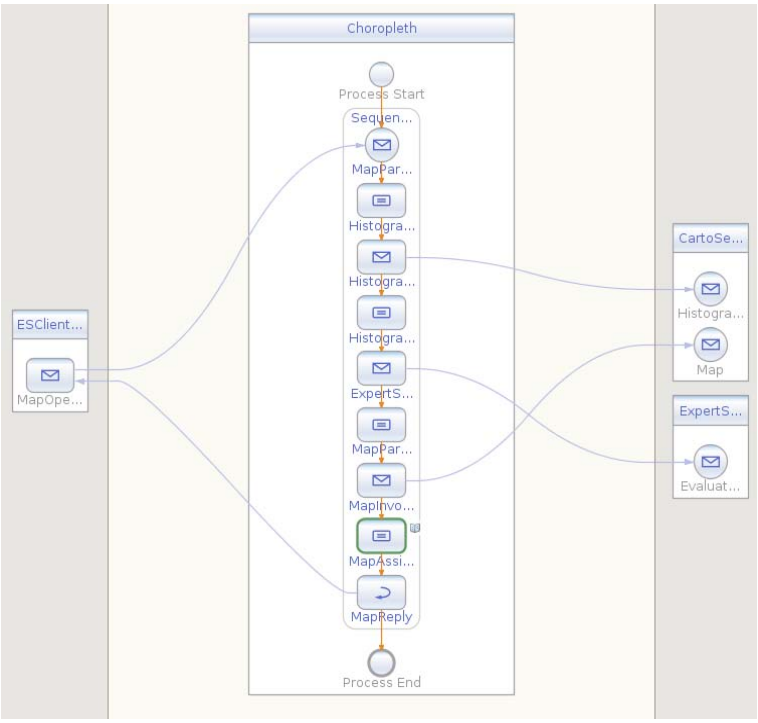

Fig . 3 Diagram of synchronous process of a map design

The process is divided into following steps. Steps marked by = sign are used for manipulating with variables and are excluded from the following list.

1. Receive an input from the user that is necessary for a histogram and a map creation.
2. Invoke web service that generates a histogram.
3. Invoke web service that evaluates histogram and generates parameters for a map creation.
4. Invoke web service that generates a map.
5. Reply the map to the user.

Full list of steps, partner links and parameters of the process is described in the appendix B.

5 Conclusion

We can use BPEL to describe process of a map creation. In a case where steps of the process can be solved by web services (e.g. The Fire Protection Atlas of the Czech Republic), the whole process can be deployed as a software component to application server and used by any user either in synchronous (one step from user side) or asynchronous (several steps from user side) way.

These two simple examples based on a real project can be a good start for the future research that can describe how to model more complicated process of a map design.

Reference

[Atlas 2009]: Peňáz et. al. Fire Protection Atlas of the Czech Republic. Available in intranet only.

[OASIS 2009] OASIS. http://www.oasis-open.org/committees/wsbpel/

[Prager 2009] PRAGER, M.; KLÍMEK, F.; RŮŽIČKA, J. GeoWeb Services Orchestration Based on BPEL or BPMN? Proceedings from GIS Ostrava 2009. VSB-TUO, Ostrava. Available at: http://gis.vsb.cz/GIS_Ostrava/GIS_Ova_2009/sbornik/Lists/Papers/057.pdf

[Růžička 2008] RŮŽIČKA, J. ISO 19115 for GeoWeb services orchestration. Proceedings from Geoinformatics on CTU 2008. CTU, Prague. Available at: http://geoinformatics.fsv.cvut.cz/wiki/index.php/ISO_19115_for_GeoWeb_services_orchestration

[Růžička 2009] RŮŽIČKA, J.; KLÍMEK, F.; ŠELIGA, M.; PRAGER, M. Do we know how to orchestrate on the GeoWeb platform?In Proceedings from GIS Ostrava 2009. Available at: http://gis.vsb.cz/GIS_Ostrava/GIS_Ova_2009/sbornik/Lists/Papers/052.pdf

Support

The article is supported by Czech Science Foundation as a part of the project **GA 205/07/0797** and the project **GA 205/09/1159**.

Appendix A. Asynchronous process in BPEL.

```
<?xml version="1.0" encoding="UTF-8"?>
<process
   name="Choropleth"
   targetNamespace="http://enterprise.netbeans.org/bpel/KartoBPEL/Choropleth"
   xmlns="http://docs.oasis-open.org/wsbpel/2.0/process/executable"
   xmlns:xsd="http://www.w3.org/2001/XMLSchema"

xmlns:sxt="http://www.sun.com/wsbpel/2.0/process/executable/SUNExtension/Trace"

xmlns:sxed="http://www.sun.com/wsbpel/2.0/process/executable/SUNExtension/Editor
"
   xmlns:tns="http://enterprise.netbeans.org/bpel/KartoBPEL/Choropleth"
   xmlns:ns0="http://xml.netbeans.org/schema/ChoroplethTypes"
xmlns:ns1="http://j2ee.netbeans.org/wsdl/Client">
   <import namespace="http://j2ee.netbeans.org/wsdl/Client" location="Client.wsdl"
importType="http://schemas.xmlsoap.org/wsdl/"/>
   <import namespace="http://j2ee.netbeans.org/wsdl/CartoService"
location="CartoService.wsdl" importType="http://schemas.xmlsoap.org/wsdl/"/>
   <partnerLinks>
      <partnerLink name="CartoServicePartnerLink"
```

```
xmlns:tns="http://j2ee.netbeans.org/wsdl/CartoService"
partnerLinkType="tns:CartoService" partnerRole="CartoServicePortTypeRole"/>
      <partnerLink                                          name="ClientPartnerLink"
xmlns:tns="http://j2ee.netbeans.org/wsdl/Client"          partnerLinkType="tns:Client"
myRole="ClientPortTypeRole"/>
   </partnerLinks>
   <variables>
      <variable name="MapOperationOut" messageType="ns1:MapResponse"/>
      <variable                                          name="CartoServiceMapOut"
xmlns:tns="http://j2ee.netbeans.org/wsdl/CartoService"
messageType="tns:CartoServiceMapResponse"/>
      <variable                                           name="CartoServiceMapIn"
xmlns:tns="http://j2ee.netbeans.org/wsdl/CartoService"
messageType="tns:CartoServiceMapRequest"/>
      <variable                                             name="MapOperationIn"
xmlns:tns="http://j2ee.netbeans.org/wsdl/Client" messageType="tns:MapRequest"/>
      <variable                                        name="HistogramOperationOut"
xmlns:tns="http://j2ee.netbeans.org/wsdl/Client"
messageType="tns:HistogramResponse"/>
      <variable                                     name="CartoServiceHistogramOut"
xmlns:tns="http://j2ee.netbeans.org/wsdl/CartoService"
messageType="tns:CartoServiceHistogramResponse"/>
      <variable                                      name="CartoServiceHistogramIn"
xmlns:tns="http://j2ee.netbeans.org/wsdl/CartoService"
messageType="tns:CartoServiceHistogramRequest"/>
      <variable                                       name="HistogramOperationIn"
xmlns:tns="http://j2ee.netbeans.org/wsdl/Client"
messageType="tns:HistogramRequest"/>
   </variables>
   <correlationSets>
      <correlationSet name="OnlyOne" properties="ns1:SessionID"/>
   </correlationSets>
   <sequence name="Sequence1">
      <receive          name="HistogramParametersRecieve"          createInstance="yes"
partnerLink="ClientPartnerLink"                          operation="HistogramOperation"
xmlns:tns="http://j2ee.netbeans.org/wsdl/Client"          portType="tns:ClientPortType"
variable="HistogramOperationIn">
         <correlations>
            <correlation set="OnlyOne" initiate="yes"/>
         </correlations>
      </receive>
      <assign name="HistogramParametersAssign">
         <copy>
            <from>$HistogramOperationIn.ExtentEventType/ns0:YearFrom</from>
```

```
            <to>$CartoServiceHistogramIn.ExtentEventType/ns0:YearFrom</to>
          </copy>
          <copy>
            <from>$HistogramOperationIn.ExtentEventType/ns0:YearTo</from>
            <to>$CartoServiceHistogramIn.ExtentEventType/ns0:YearTo</to>
          </copy>
          <copy>
            <from>$HistogramOperationIn.ExtentEventType/ns0:EventType</from>
            <to>$CartoServiceHistogramIn.ExtentEventType/ns0:EventType</to>
          </copy>
        </assign>
        <invoke name="HistogramInvoke" partnerLink="CartoServicePartnerLink"
operation="Histogram" xmlns:tns="http://j2ee.netbeans.org/wsdl/CartoService"
portType="tns:CartoServicePortType" inputVariable="CartoServiceHistogramIn"
outputVariable="CartoServiceHistogramOut"/>
        <assign name="HistogramOutAssign">
          <copy>
            <from variable="CartoServiceHistogramOut" part="Histogram"/>
            <to variable="HistogramOperationOut" part="Histogram"/>
          </copy>
        </assign>
        <reply name="HistogramReply" partnerLink="ClientPartnerLink"
operation="HistogramOperation" portType="ns1:ClientPortType"
variable="HistogramOperationOut"/>
        <receive name="MapParametersRecieve" createInstance="no"
partnerLink="ClientPartnerLink" operation="MapOperation"
portType="ns1:ClientPortType" variable="MapOperationIn">
          <correlations>
            <correlation set="OnlyOne" initiate="no"/>
          </correlations>
        </receive>
        <assign name="MapParametersAssign">
          <copy>
            <from>$MapOperationIn.MapParameters/ns0:NumberOfClasses</from>
            <to>$CartoServiceMapIn.MapParameters/ns0:NumberOfClasses</to>
          </copy>
          <copy>
            <from>$MapOperationIn.MapParameters/ns0:Method</from>
            <to>$CartoServiceMapIn.MapParameters/ns0:Method</to>
          </copy>
          <copy>
            <from>$MapOperationIn.MapParameters/ns0:Frequency</from>
            <to>$CartoServiceMapIn.MapParameters/ns0:Frequency</to>
          </copy>
```

```
            <copy>
               <from>$MapOperationIn.MapParameters/ns0:NuberOfInhabitants</from>
               <to>$CartoServiceMapIn.MapParameters/ns0:NuberOfInhabitants</to>
            </copy>
            <copy>
               <from>$MapOperationIn.MapParameters/ns0:ColourFrom</from>
               <to>$CartoServiceMapIn.MapParameters/ns0:ColourFrom</to>
            </copy>
            <copy>
               <from>$MapOperationIn.MapParameters/ns0:ColourTo</from>
               <to>$CartoServiceMapIn.MapParameters/ns0:ColourTo</to>
            </copy>
            <copy>
               <from>$HistogramOperationIn.ExtentEventType/ns0:YearFrom</from>
               <to>$CartoServiceMapIn.ExtentEventType/ns0:YearFrom</to>
            </copy>
            <copy>
               <from>$HistogramOperationIn.ExtentEventType/ns0:YearTo</from>
               <to>$CartoServiceMapIn.ExtentEventType/ns0:YearTo</to>
            </copy>
            <copy>
               <from>$HistogramOperationIn.ExtentEventType/ns0:EventType</from>
               <to>$CartoServiceMapIn.ExtentEventType/ns0:EventType</to>
            </copy>
         </assign>
         <invoke name="MapInvoke" partnerLink="CartoServicePartnerLink"
operation="Map" xmlns:tns="http://j2ee.netbeans.org/wsdl/CartoService"
portType="tns:CartoServicePortType" inputVariable="CartoServiceMapIn"
outputVariable="CartoServiceMapOut"/>
         <assign name="MapAssign">
            <copy>
               <from variable="CartoServiceMapOut" part="Map"/>
               <to variable="MapOperationOut" part="Map"/>
            </copy>
         </assign>
         <reply name="MapReply" partnerLink="ClientPartnerLink"
operation="MapOperation" portType="ns1:ClientPortType"
variable="MapOperationOut"/>
      </sequence>
</process>
```

Appendix B. Synchronous process in BPEL.

```
<?xml version="1.0" encoding="UTF-8"?>
<process
```

```
   name="Choropleth"
   targetNamespace="http://enterprise.netbeans.org/bpel/KartoBPEL/Choropleth"
   xmlns="http://docs.oasis-open.org/wsbpel/2.0/process/executable"
   xmlns:xsd="http://www.w3.org/2001/XMLSchema"

xmlns:sxt="http://www.sun.com/wsbpel/2.0/process/executable/SUNExtension/Trace"

xmlns:sxed="http://www.sun.com/wsbpel/2.0/process/executable/SUNExtension/Editor
"
   xmlns:tns="http://enterprise.netbeans.org/bpel/KartoBPEL/Choropleth"
   xmlns:ns0="http://xml.netbeans.org/schema/ChoroplethTypes"
xmlns:ns1="http://j2ee.netbeans.org/wsdl/Client">
   <import namespace="http://j2ee.netbeans.org/wsdl/Client" location="Client.wsdl"
importType="http://schemas.xmlsoap.org/wsdl/"/>
   <import namespace="http://j2ee.netbeans.org/wsdl/CartoService"
location="CartoService.wsdl" importType="http://schemas.xmlsoap.org/wsdl/"/>
   <import namespace="http://j2ee.netbeans.org/wsdl/ExpertSystem"
location="ExpertSystem.wsdl" importType="http://schemas.xmlsoap.org/wsdl/"/>
   <import namespace="http://j2ee.netbeans.org/wsdl/ESClient"
location="ESClient.wsdl" importType="http://schemas.xmlsoap.org/wsdl/"/>
   <partnerLinks>
      <partnerLink name="ESClientPartnerLink"
xmlns:tns="http://j2ee.netbeans.org/wsdl/ESClient" partnerLinkType="tns:ESClient"
myRole="ESClientPortTypeRole"/>
      <partnerLink name="CartoServicePartnerLink"
xmlns:tns="http://j2ee.netbeans.org/wsdl/CartoService"
partnerLinkType="tns:CartoService" partnerRole="CartoServicePortTypeRole"/>
      <partnerLink name="ExpertSystemPartnerLink"
xmlns:tns="http://j2ee.netbeans.org/wsdl/ExpertSystem"
partnerLinkType="tns:ExpertSystem" partnerRole="ExpertSystemPortTypeRole"/>
   </partnerLinks>
   <variables>
      <variable name="MapOperationOut1" messageType="ns1:MapResponse"/>
      <variable name="MapOperationIn1"
xmlns:tns="http://j2ee.netbeans.org/wsdl/ESClient"
messageType="tns:MapOperationRequest"/>
      <variable name="EvaluateDataOut"
xmlns:tns="http://j2ee.netbeans.org/wsdl/ExpertSystem"
messageType="tns:EvaluateDataResponse"/>
      <variable name="EvaluateDataIn"
xmlns:tns="http://j2ee.netbeans.org/wsdl/ExpertSystem"
messageType="tns:EvaluateDataRequest"/>
      <variable name="CartoServiceMapOut"
xmlns:tns="http://j2ee.netbeans.org/wsdl/CartoService"
```

```
messageType="tns:CartoServiceMapResponse"/>
      <variable                                                    name="CartoServiceMapIn"
xmlns:tns="http://j2ee.netbeans.org/wsdl/CartoService"
messageType="tns:CartoServiceMapRequest"/>
      <variable                                                name="CartoServiceHistogramOut"
xmlns:tns="http://j2ee.netbeans.org/wsdl/CartoService"
messageType="tns:CartoServiceHistogramResponse"/>
      <variable                                                 name="CartoServiceHistogramIn"
xmlns:tns="http://j2ee.netbeans.org/wsdl/CartoService"
messageType="tns:CartoServiceHistogramRequest"/>
   </variables>

   <sequence name="Sequence1">
      <receive              name="MapParametersReceive"              createInstance="yes"
partnerLink="ESClientPartnerLink"                                  operation="MapOperation"
xmlns:tns="http://j2ee.netbeans.org/wsdl/ESClient"   portType="tns:ESClientPortType"
variable="MapOperationIn1">

      </receive>
      <assign name="HistogramParametersAssign">
         <copy>
            <from variable="MapOperationIn1" part="ExtentEventType"/>
            <to variable="CartoServiceHistogramIn" part="ExtentEventType"/>
         </copy>
      </assign>
      <invoke      name="HistogramInvoke"      partnerLink="CartoServicePartnerLink"
operation="Histogram"           xmlns:tns="http://j2ee.netbeans.org/wsdl/CartoService"
portType="tns:CartoServicePortType"          inputVariable="CartoServiceHistogramIn"
outputVariable="CartoServiceHistogramOut"/>
      <assign name="HistogramOutAssign">
         <copy>
            <from variable="CartoServiceHistogramOut" part="Histogram"/>
            <to variable="EvaluateDataIn" part="Histogram"/>
         </copy>
      </assign>
      <invoke   name="ExpertSystemInvoke"   partnerLink="ExpertSystemPartnerLink"
operation="EvaluateData"       xmlns:tns="http://j2ee.netbeans.org/wsdl/ExpertSystem"
portType="tns:ExpertSystemPortType"                inputVariable="EvaluateDataIn"
outputVariable="EvaluateDataOut"/>
      <assign name="MapParametersAssign">
         <copy>
            <from variable="EvaluateDataOut" part="MapParameters"/>
            <to variable="CartoServiceMapIn" part="MapParameters"/>
         </copy>
```

```
        <copy>
          <from variable="MapOperationIn1" part="ExtentEventType"/>
          <to variable="CartoServiceMapIn" part="ExtentEventType"/>
        </copy>
      </assign>
      <invoke name="MapInvoke" partnerLink="CartoServicePartnerLink"
operation="Map" xmlns:tns="http://j2ee.netbeans.org/wsdl/CartoService"
portType="tns:CartoServicePortType" inputVariable="CartoServiceMapIn"
outputVariable="CartoServiceMapOut"/>
      <assign name="MapAssign">
        <copy>
          <from variable="CartoServiceMapOut" part="Map"/>
          <to variable="MapOperationOut1" part="Map"/>
        </copy>
      </assign>
      <reply name="MapReply" partnerLink="ESClientPartnerLink"
operation="MapOperation" portType="tns:ESClientPortType"
variable="MapOperationOut1" xmlns:tns="http://j2ee.netbeans.org/wsdl/ESClient"/>
    </sequence>
</process>
```

Appendix C. Web services WSDLs

```
<?xml version="1.0" encoding="UTF-8"?>
<definitions name="CartoService"
targetNamespace="http://j2ee.netbeans.org/wsdl/CartoService"
    xmlns="http://schemas.xmlsoap.org/wsdl/"
    xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:tns="http://j2ee.netbeans.org/wsdl/CartoService"
xmlns:ns="http://xml.netbeans.org/schema/ChoroplethTypes"
xmlns:plnk="http://docs.oasis-open.org/wsbpel/2.0/plnktype"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:ns0="http://xml.netbeans.org/schema/UserTypes">
    <types>
      <xsd:schema targetNamespace="http://j2ee.netbeans.org/wsdl/CartoService">
        <xsd:import namespace="http://xml.netbeans.org/schema/ChoroplethTypes"
schemaLocation="ChoroplethIOTypes.xsd"/>
      </xsd:schema>
    </types>
    <message name="CartoServiceHistogramRequest">
      <part name="ExtentEventType" element="ns:ExtentEventType"/>
    </message>
    <message name="CartoServiceHistogramResponse">
      <part name="Histogram" element="ns:HistogramURL"/>
    </message>
```

```
    <message name="CartoServiceMapRequest">
      <part name="MapParameters" element="ns:MapParameters"/>
      <part name="ExtentEventType" element="ns:ExtentEventType"/>
    </message>
    <message name="CartoServiceMapResponse">
      <part name="Map" element="ns:MapURL"/>
    </message>
    <portType name="CartoServicePortType">
      <operation name="Histogram">
        <input name="input1" message="tns:CartoServiceHistogramRequest"/>
        <output name="output1" message="tns:CartoServiceHistogramResponse"/>
      </operation>
       <operation name="Map">
        <input name="input2" message="tns:CartoServiceMapRequest"/>
        <output name="output2" message="tns:CartoServiceMapResponse"/>
      </operation>
    </portType>
    <binding name="CartoServiceBinding" type="tns:CartoServicePortType">
      <soap:binding                                                    style="document"
transport="http://schemas.xmlsoap.org/soap/http"/>
      <operation name="Histogram">
        <soap:operation/>
        <input name="input1">
          <soap:body use="literal"/>
        </input>
        <output name="output1">
          <soap:body use="literal"/>
        </output>
      </operation>
      <operation name="Map">
        <soap:operation/>
        <input name="input2">
          <soap:body use="literal"/>
        </input>
        <output name="output2">
          <soap:body use="literal"/>
        </output>
      </operation>
    </binding>
    <service name="CartoServiceService">
      <port name="CartoServicePort" binding="tns:CartoServiceBinding">
        <soap:address
location="http://localhost:${HttpDefaultPort}/CartoServiceService/CartoServicePort"/>
      </port>
```

```
    </service>
    <plnk:partnerLinkType name="CartoService">
        <!-- A partner link type is automatically generated when a new port type is added.
Partner link types are used by BPEL processes.
In a BPEL process, a partner link represents the interaction between the BPEL process
and a partner service. Each partner link is associated with a partner link type.
A partner link type characterizes the conversational relationship between two services.
The partner link type can have one or two roles.-->
        <plnk:role                                    name="CartoServicePortTypeRole"
portType="tns:CartoServicePortType"/>
    </plnk:partnerLinkType>
</definitions>

<?xml version="1.0" encoding="UTF-8"?>
<definitions                                                    name="ExpertSystem"
targetNamespace="http://j2ee.netbeans.org/wsdl/ExpertSystem"
    xmlns="http://schemas.xmlsoap.org/wsdl/"
    xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:tns="http://j2ee.netbeans.org/wsdl/ExpertSystem"
xmlns:ns="http://xml.netbeans.org/schema/ChoroplethTypes"
xmlns:plnk="http://docs.oasis-open.org/wsbpel/2.0/plnktype"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/">
    <types>
        <xsd:schema targetNamespace="http://j2ee.netbeans.org/wsdl/ExpertSystem">
            <xsd:import       namespace="http://xml.netbeans.org/schema/ChoroplethTypes"
schemaLocation="ChoroplethIOTypes.xsd"/>
        </xsd:schema>
    </types>
    <message name="EvaluateDataRequest">
        <part name="Histogram" element="ns:HistogramURL"/>
    </message>
    <message name="EvaluateDataResponse">
        <part name="MapParameters" element="ns:MapParameters"/>
    </message>
    <portType name="ExpertSystemPortType">
        <operation name="EvaluateData">
            <input name="input1" message="tns:EvaluateDataRequest"/>
            <output name="output1" message="tns:EvaluateDataResponse"/>
        </operation>
    </portType>
    <binding name="ExpertSystemBinding" type="tns:ExpertSystemPortType">
        <soap:binding                                                   style="document"
transport="http://schemas.xmlsoap.org/soap/http"/>
```

```
        <operation name="EvaluateData">
            <soap:operation/>
            <input name="input1">
                <soap:body use="literal"/>
            </input>
            <output name="output1">
                <soap:body use="literal"/>
            </output>
        </operation>
    </binding>
    <service name="ExpertSystemService">
        <port name="ExpertSystemPort" binding="tns:ExpertSystemBinding">
            <soap:address
location="http://localhost:${HttpDefaultPort}/ExpertSystemService/ExpertSystemPort"
/>
        </port>
    </service>
    <plnk:partnerLinkType name="ExpertSystem">
        <!-- A partner link type is automatically generated when a new port type is added.
Partner link types are used by BPEL processes.
In a BPEL process, a partner link represents the interaction between the BPEL process
and a partner service. Each partner link is associated with a partner link type.
A partner link type characterizes the conversational relationship between two services.
The partner link type can have one or two roles.-->
        <plnk:role                                    name="ExpertSystemPortTypeRole"
portType="tns:ExpertSystemPortType"/>
    </plnk:partnerLinkType>
</definitions>
```

Appendix D. Client WSDL for asynchronous process.

```
<?xml version="1.0" encoding="UTF-8"?>
<definitions name="Client" targetNamespace="http://j2ee.netbeans.org/wsdl/Client"
    xmlns="http://schemas.xmlsoap.org/wsdl/"
    xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:tns="http://j2ee.netbeans.org/wsdl/Client"
xmlns:ns="http://xml.netbeans.org/schema/ChoroplethTypes"
xmlns:plnk="http://docs.oasis-open.org/wsbpel/2.0/plnktype"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"    xmlns:vprop="http://docs.oasis-
open.org/wsbpel/2.0/varprop"
xmlns:ns0="http://xml.netbeans.org/schema/UserTypes">
    <types>
        <xsd:schema targetNamespace="http://j2ee.netbeans.org/wsdl/Client">
            <xsd:import       namespace="http://xml.netbeans.org/schema/ChoroplethTypes"
```

```
schemaLocation="ChoroplethIOTypes.xsd"/>
            <xsd:import                                    schemaLocation="UserTypes.xsd"
namespace="http://xml.netbeans.org/schema/UserTypes"/>
        </xsd:schema>
    </types>
    <message name="HistogramRequest">
        <part name="ExtentEventType" element="ns:ExtentEventType"/>
        <part name="User" element="ns:UserStatus"/>
    </message>
    <message name="HistogramResponse">
        <part name="Histogram" element="ns:HistogramURL"/>
    </message>
    <message name="MapRequest">
        <part name="MapParameters" element="ns:MapParameters"/>
        <part name="User" element="ns:UserStatus"/>
    </message>
    <message name="MapResponse">
        <part name="Map" element="ns:MapURL"/>
    </message>
    <portType name="ClientPortType">
        <operation name="HistogramOperation">
            <input name="input1" message="tns:HistogramRequest"/>
            <output name="output1" message="tns:HistogramResponse"/>
        </operation>
        <operation name="MapOperation">
            <input name="input2" message="tns:MapRequest"/>
            <output name="output2" message="tns:MapResponse"/>
        </operation>
    </portType>
    <binding name="ClientBinding" type="tns:ClientPortType">
        <soap:binding                                                    style="document"
transport="http://schemas.xmlsoap.org/soap/http"/>
        <operation name="HistogramOperation">
            <soap:operation/>
            <input name="input1">
                <soap:body use="literal"/>
            </input>
            <output name="output1">
                <soap:body use="literal"/>
            </output>
        </operation>
        <operation name="MapOperation">
            <soap:operation/>
            <input name="input2">
```

```
            <soap:body use="literal"/>
         </input>
         <output name="output2">
            <soap:body use="literal"/>
         </output>
      </operation>
   </binding>
   <service name="ClientService">
      <port name="ClientPort" binding="tns:ClientBinding">
         <soap:address
location="http://localhost:${HttpDefaultPort}/ClientService/ClientPort"/>
      </port>
   </service>
   <plnk:partnerLinkType name="Client">
      <!-- A partner link type is automatically generated when a new port type is added.
Partner link types are used by BPEL processes.
In a BPEL process, a partner link represents the interaction between the BPEL process
and a partner service. Each partner link is associated with a partner link type.
A partner link type characterizes the conversational relationship between two services.
The partner link type can have one or two roles.-->
      <plnk:role name="ClientPortTypeRole" portType="tns:ClientPortType"/>
   </plnk:partnerLinkType>
   <vprop:property name="SessionID" type="xsd:string"/>
   <vprop:propertyAlias                                    propertyName="tns:SessionID"
messageType="tns:HistogramRequest" part="ns0:User/id"/>
   <vprop:propertyAlias                                    propertyName="tns:SessionID"
messageType="tns:MapRequest" part="ns0:User/id"/>
</definitions>
```

Appendix E. Client WSDL for synchronous process.

```
<?xml version="1.0" encoding="UTF-8"?>
<definitions                                                    name="ESClient"
targetNamespace="http://j2ee.netbeans.org/wsdl/ESClient"
   xmlns="http://schemas.xmlsoap.org/wsdl/"
   xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
   xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:tns="http://j2ee.netbeans.org/wsdl/ESClient"
xmlns:ns="http://xml.netbeans.org/schema/ChoroplethTypes"
xmlns:plnk="http://docs.oasis-open.org/wsbpel/2.0/plnktype"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:ns0="http://xml.netbeans.org/schema/UserTypes">
   <types>
      <xsd:schema targetNamespace="http://j2ee.netbeans.org/wsdl/ESClient">
         <xsd:import       namespace="http://xml.netbeans.org/schema/ChoroplethTypes"
```

```
schemaLocation="ChoroplethIOTypes.xsd"/>
            <xsd:import                                    schemaLocation="UserTypes.xsd"
namespace="http://xml.netbeans.org/schema/UserTypes"/>
        </xsd:schema>
    </types>
    <message name="MapOperationRequest">
        <part name="ExtentEventType" element="ns:ExtentEventType"/>
        <part name="User" element="ns0:UserType"/>
    </message>
    <message name="MapOperationResponse">
        <part name="Map" element="ns:MapURL"/>
    </message>
    <portType name="ESClientPortType">
        <operation name="MapOperation">
            <input name="input1" message="tns:MapOperationRequest"/>
            <output name="output1" message="tns:MapOperationResponse"/>
        </operation>
    </portType>
    <binding name="ESClientBinding" type="tns:ESClientPortType">
        <soap:binding                                                    style="document"
transport="http://schemas.xmlsoap.org/soap/http"/>
        <operation name="MapOperation">
            <soap:operation/>
            <input name="input1">
                <soap:body use="literal"/>
            </input>
            <output name="output1">
                <soap:body use="literal"/>
            </output>
        </operation>
    </binding>
    <service name="ESClientService">
        <port name="ESClientPort" binding="tns:ESClientBinding">
            <soap:address
location="http://localhost:${HttpDefaultPort}/ESClientService/ESClientPort"/>
        </port>
    </service>
    <plnk:partnerLinkType name="ESClient">
        <!-- A partner link type is automatically generated when a new port type is added.
Partner link types are used by BPEL processes.
In a BPEL process, a partner link represents the interaction between the BPEL process
and a partner service. Each partner link is associated with a partner link type.
A partner link type characterizes the conversational relationship between two services.
The partner link type can have one or two roles.-->
```

```
      <plnk:role name="ESClientPortTypeRole" portType="tns:ESClientPortType"/>
   </plnk:partnerLinkType>
</definitions>
```

Appendix F. Used data types.

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
   targetNamespace="http://xml.netbeans.org/schema/ChoroplethTypes"
   xmlns:tns="http://xml.netbeans.org/schema/ChoroplethTypes"
   elementFormDefault="qualified">
   <xsd:complexType name="ExtentEventTypeType">
      <xsd:sequence>
         <xsd:element name="YearFrom" type="xsd:gYear"/>
         <xsd:element name="YearTo" type="xsd:gYear"/>
         <xsd:element name="EventType" type="xsd:string"/>
      </xsd:sequence>
   </xsd:complexType>
   <xsd:complexType name="MapParametersType">
      <xsd:sequence>
         <xsd:element name="NumberOfClasses" type="xsd:int"/>
         <!--
         For purpose of the article were used string codes.
         In a common practise should be used less memory consuming data types.
         -->
         <xsd:element name="Method" type="xsd:string">
            <xs:simpleType>
               <xs:restriction base="xs:string">
                  <xs:enumeration value="EqualIntervals"/>
                  <xs:enumeration value="Jenks"/>
                  <xs:enumeration value="Quantiles"/>
                  <xs:enumeration value="StandardDeviation"/>
               </xs:restriction>
            </xs:simpleType>
         </xsd:element>
         <xsd:element name="Frequency">
            <xs:simpleType>
               <xs:restriction base="xs:string">
                  <xs:enumeration value="BasedOnCountOfInhabitants"/>
                  <xs:enumeration value="BasedOnSquaryKmOfRegionArea"/>
                  <xs:enumeration value="BasedOnCountOfInterventions"/>
               </xs:restriction>
            </xs:simpleType>
         </xsd:element>
         <xsd:element name="NuberOfInhabitants">
```

```
            <xs:simpleType>
               <xs:restriction base="xs:int">
                  <xs:enumeration value="1000"/>
                  <xs:enumeration value="10000"/>
                  <xs:enumeration value="100000"/>
                  <xs:enumeration value="1000000"/>
               </xs:restriction>
            </xs:simpleType>
         </xsd:element>
         <xsd:element name="ColourFrom" type="xsd:string"/>
         <xsd:element name="ColourTo" type="xsd:string"/>
      </xsd:sequence>
   </xsd:complexType>
   <xsd:element name="ExtentEventType" type="tns:ExtentEventTypeType"/>
   <xsd:element name="MapParameters" type="tns:MapParametersType"/>
   <xsd:element name="HistogramURL" type="xsd:anyURI"/>
   <xsd:element name="MapURL" type="xsd:anyURI"/>
   <xsd:element name="SessionID" type="xsd:string"/>
</xsd:schema>

<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
   targetNamespace="http://xml.netbeans.org/schema/UserTypes"
   xmlns:tns="http://xml.netbeans.org/schema/UserTypes"
   elementFormDefault="qualified">
   <xsd:element name="UserStatus" type="tns:UserStatusType"/>
   <xsd:complexType name="UserStatusType">
      <xsd:sequence>
         <xsd:element name="id" type="xsd:string"/>
         <xsd:element name="clientType" type="xsd:string"/>
      </xsd:sequence>
   </xsd:complexType>
</xsd:schema>
```