Using the WaterML Standard Information Model, in a SCADA Federation Web Service

Iulia Ştefan*, Szilárd Enyedi*,
Andrei Scurtu*, Liviu Miclea*, Ioan Stoian, Dorina Căpăţînă****

** Technical University of Cluj-Napoca, Cluj-Napoca, România (e-mail: Iulia.Stefan@aut.utcluj.ro, Szilard.Enyedi@aut.utcluj.ro, scurtuandrei@rocketmail.com, Liviu.Miclea@aut.utcluj.ro).*
*** IPA S.A., Cluj-Napoca subs. Cluj-Napoca, Romania (e-mail: stoian@automation.ro, dorina@automation.ro)*

Abstract: Large scale, water management infrastructures have various parts – water collection, distribution, wastewater treatment, to name a few. Being so large, these systems have their own SCADA monitoring and control structures. However, these systems also depend on each other, so they need to communicate and, also, external authorities need to extract information from them. If these systems were connected in a federation, or, software wise, a cloud, they could interchange data easier, among each other and with authorized externals. This paper describes a web service, developed to offer controlled access to the resources of a multi-member federation of SCADA systems. Access to the federation and its data is through a single point of entry and is curated by gateways. The web service returns the requested data in WaterML format, an XML standard specific to water systems.

Keywords: information system, cyber-physical system, SCADA federation, WaterML, web service.

1. INTRODUCTION

Large scale infrastructures have always been difficult to monitor, if only for their geographical extent. Harnessing the power of nature is a challenging task and requires a human and technological concerted effort. Nowadays, the technology includes not only the classic equipment that makes these systems work, but also wireless sensor networks (Folea and Moiș, 2015) and Internet-enabled sensors and actuators, as well as the software to monitor and control all the parts. All these components qualify these large systems as true cyber-physical ones.

Many of these infrastructures are managed with the aid of SCADA (Supervisory Control and Data Acquisition), the de facto standard for large system/plant monitoring and control.

1.1 Water Management Systems

A particular kind of these infrastructures is one that deals with water management – river basins, lakes, drinking or industrial water, its supply and distribution network, wastewater and its treatment. Most of these systems have medium response times, compared to, say, managing the power grid. On the other hand, bad weather, floods, and irresponsible human behaviour may precipitate events, so quick control mechanisms must always be in place.

1.2 Water SCADA

An interesting aspect of monitoring and controlling water management systems is that these systems are more or less interdependent. For example, the weather specialists must interact with the river professionals, who should communicate with the water supply experts, but they must inform about water delivery to the wastewater treatment engineers and so forth.

1.3 F2S Project

The work presented in this paper is part of the pilot project, F2S (Federation of SCADA Systems), on the Someș river basin of Romania. On one hand, since each part of this water management system functions independently, there are a few SCADAs, managing each. On the other hand, the resources they control are interlinked, so these SCADA structures need to communicate (Moiș et al., 2015). Additionally, regional authorities need access to water information, which should arrive swiftly and in electronic form.

For the reasons above, these SCADA systems should be interconnected. Our solution represents a federative SCADA structure. The federation members have equal rights. Physically, there is little change to the previous, non-federative situation. From software point of view, though, there is an F2S cloud. The federation members and external authorities may access federation data, with proper credentials.

(Stoian et al., 2010) define the concept of federative SCADA systems, from the main global characteristics through the architecture and main components, towards the complex event processing unit, by extending the cooperation between the individuals, using "envelope" functions and the concept of services. We propose an improved solution, that adds a mechanism for implementing a standardized model for the delivery of such services through web services and a WaterML based XML structure.

1.4 Data Collected from Federation members

Due to the heterogeneous nature of the federation, the data collected from the member SCADAs present various structures.

Most data must specify a timestamp about its recording/storage moment. Some data is actually a series of measurements, either each having their own timestamp, or a start timestamp and a measurement frequency.

Some data is analogue, with decimal values; other data is digital, with a simple one or zero as value. Some data is specified backward in time, having the timestamp of the last measurement and then the measurement frequency and measurement count, backwards in time (especially for measurements which are short-lived and are not kept indefinitely).

Other values are actually read and then written down by the operator, not measured automatically. These are higher level values; they do not come directly from the sensors. Yet other types of measurements are the event related data, written in the log; these records are not numerical, but in text form, about the status of the system, or some mission-critical actuator, all with date and time. Some other values are not momentary measurements, but averages, because recording the actual measurements would take too many resources. One main goal of the F2S SCADA federation is to establish a unified way for structuring this heterogeneous data. There are ISO standards about geospatial information (ISO 19115-1:2014 (*, 2014) and ISO/TS 19139:2007 (*, 2007)). However, these deal only with the metadata, i.e. the properties of the transmitted data, not the data itself.

(Ramanathan and Korte, 2014) chose the JSON (JavaScript Object Notation) format for data interchange, based on the small number of information needed for data interchange. The information source and the type of sensors are hidden, the only data maintained being the id, the value, the unit, the measurement itself, an approach not reliable when the identity of the provider and the measurement type are necessary as a federation of SCADAs requires.

The importance of standardization in weather data transfer is evident from (Kaoa et al., 2011) also. Water Data Transfer Format (WDTF) is the standard in use for the specific information to be interchanged: flows, groundwater, water storages, meteorology, water use, urban water and water quality and is mainly used in Australian water monitoring systems. For control purposes, to view and to verify the WDTF data, the authors present a specialized tool, the Visualisation Tool, based on Java. The tool facilitates the inspection of the time series, generates plots, compares the time series and creates ratings. Currently, WDTF is included in the WaterML 2.0 standard.

1.5 Web Services

In order to standardize the access, both from security and ease of development point of view, the F2S cloud offers the data through a web service, called E-Service. Web services are applications running on web servers that can be contacted and queried through the web, meaning they work even through proxies and routers. Also, the technique to access the methods offered by a web service, is described in a standard fashion, so that any client software, developed in any language, is able to access the served resources.

2. STATE OF THE ART

For a better understanding of WaterML Web services, a short analysis regarding the WaterML usage is required.

(Kadlec et al., 2015) present a prototype of WaterML R, a software resource that facilitates hydrological data management and storage. The package offers support for several defining methods from the WaterOneFlow web service specifications. The resource offers the possibility for a client to use the data from CUAHSI, Water Data Centre and Global Earth Observation Systems.

In (Yu et al., 2015), the authors underline the importance of a water observation interchange standard. The WaterML 2.0 standard is considered from two points of view: design and implementation. The importance of a WaterML validation service is also discussed, following the concern that there may be suspicions regarding the integrity of interchanged data. The authors propose the use of Schematron, an XML validation language. By defining rules, the validation language is able to detect the compliance with WaterML 2.0, for each water data interchange by providers. The final solution is exposed as a validation service posted online, on the WaterML site.

(Hussaina et al., 2015) describe an OGC (Open Geospacial Consortium) based framework, in order to integrate interoperability directives in water data interchange format, through web services. The main feature is the integration of format independent client schema. The developed architecture follows the component-based model. Using an ontology support system, the interchange file that each client provides - Database, Excel, Web service - is parsed and the Output Data to be integrated in the Sensor Observation Service database is generated. The model was in testing phase at the publishing date and the challenges were raised by the diversity of the file types exchanged by different providers.

(Dow et al., 2015) analyze the Software as a Service paradigm in congruency with water-data community. The services are available on the Internet, for a large community that produces, monitors, exposes, analyses and provides water based extended content, based on the paradigm mentioned before. Also, the authors underline the important contribution of the CUAHSI Hydrologic Information System (HIS) for all the gathering and interchange processes to be happening. One objective is to offer means of exploration regarding the understanding of the specific binding between weather conditions and the quality and safety of local water bodies, not only locally, but at a global scale. Another objective is given by the need for promoting the inter-domain cooperation, in order to identify new data sources and new application development opportunities.

Next, the details regarding the versions of WaterML are introduced.

3. WATERML 1.0/1.1

3.1 Short History

Starting in 2005, the Consortium of Universities for the Advancement of Hydrologic Science created a number of web services, allowing different users to get access to the hydrological data. The development of the services was ad-hoc, serving different needs without a central coordination, linking them into a unified platform of a data provider. As the number and the heterogeneity of data increased and more services were integrated into the system, their scaling and management became more and more difficult.

Consequently, the system's architects become aware of the need to define a common language, unifying metadata semantics. As a result, WaterML became a common language providing a suitable denominator for the CUAHSI platform. The Consortium was developing, at the same time, an information model called Observations Data Model (ODM) to represent observation data in a generic way. The objective was to unify the various source schemas.

Among the goals of WaterML encoding language, there are several mentioned:

- establishing a common exchange format for hydrological data,
- creating a standard as GML (Geography Markup Language),
- creating a solution to store information including quality data, etc.

Regarding the file extensions, the WaterML standards use:

- XML – classic XML file,
- WML – specific abbreviation for WaterML 2.0,
- WMZ – an extension that indicate that the file is archived (zipped WML).

An important aspect of working with WaterML file stored information is the transport of data. There are some standard approaches such as:

- classic ways – email, ftp transfer, copy, etc.,
- http transfer,
- on demand, through a web service.

3.2 Core Concepts

WaterML is now the Water Markup Language of Consortium of Universities known, also, as CUAHSI Water Markup Language. WaterML is a custom XML schema, specially designed to encode the semantics of hydrologic observation discovery, being both a generic and an unambiguous data format.

There are two important variables that are embedded in WaterML: the time and the space information that is part of any observation. In this particular case, the act of observation is considered to be the procedure of assigning a number, symbol or term to a phenomenon that makes the subject of the observation. The observations vary based on the process of measurement, even if they share a common ground, by the fact that measurement is used with a narrower meaning, always providing a quantitative outcome, while observation can be more generic. The objective of the hydrologic observations is to encapsulate different phenomena related to specific time and space information. The hydrologic observations are identified by a few characteristics such as:

- the space represented by the geographical coordinates where the observation is made,
- the date and the time representing the exact point in time when the observation is made,
- the specific variables that make the object of the observation such as water quality, water surface elevation, streamflow, etc.

The source from where WaterML derives its data is the CUAHSI Observations Data Model (ODM). Even if the main purpose of WaterML is to provide a common representation language for ODM data, its purpose is broader.

3.3 Observation Network and Series

The concept of observation series is defined with the scope of integration of individual observations into a series catalogue entities database, which can be easily referenced.

3.4 XML Schema

WaterML takes full advantage of using polymorphic typing for extending XML schemas. This way, WaterML is not defined narrowly, but it is a generic schema type adapted to specific needs of CUAHSI and more. For example, the <LatLonPointType> node extends the <GeoLocationType> to include child elements to be able integrating latitude and longitude of a point in space. In a similar way, <LatLonBoxType> is defined in a way to extend <GeoLocationType>, this being generated in the first version of WaterML.

3.5Variable Elements

WaterML defines variable elements as well, that allow encoding any sort of data matrix required by the system. There are two types of information stored, first providing variable general information and second, variable specific information, meant to store content information recorded into that variable. <VariableInfoType> allows defining the variable such as <variableCode>, <variableName>, <sampleMedium>, <units>.

3.6 Time Information

Time related information can be encoded in more ways. Two examples:

Using timeInterval tag:

```
<timeInterval>
 <beginDateTime>2015-11-09T19:50:00Z</beginDateTime>
 <endDateTime>2015-11-09T20:50:00Z</endDateTime>
</timeInterval>
```

Using variableTimeInterval:

```
<variableTimeInterval xsi:type="TimeIntervalType">
 <beginDateTime>2015-11-09T19:50:00Z</beginDateTime>
 <endDateTime>2015-11-09T20:50:00Z</endDateTime>
</variableTimeInterval>
```

4. WATERML 2.0

Once the WaterML model gained popularity, further development of the first version was imminent. The new release of WaterML is based on OGC Observation and Measurement standard ISO 19156.

The integration of the new features required by different factors added significant improvement of various aspects related to the previous version.

WaterML 2.0 became a new, common information model based on different standards related to different countries' usage of the OGC standard. The new release increased the scalability of communication among different systems and standards, and maintainability. At the same time, the new release brought a challenge to the systems providers who used the first version, to adapt the communication to the new format.

Some of the most used classes in WaterML 2.0, compared to the previous version:

- /conf/xsd-xml-rules
- /conf/xsd-collection
- /conf/xsd-timeseries-observation
- /conf/xsd-timeseries-tvp-observation
- /conf/xsd-timeseries-tvp
- /conf/xsd-feature-of-interest-monitoring-point
- /conf/xsd-measurement-timeseries-tvp
- /conf/xsd-observation-process.

4.1 Categorical timeseries

Categorical time series use the classes:

- /conf/xsd-xml-rules
- /conf/xsd-timeseries-tvp-observation
- /conf/xsd-categorical-timeseries-tvp

4.2 Document Metadata

This information tells the client which requested the data, about the SCADA system where the data was collected, as well as the creation timestamp for the WaterML document (XML structure) containing the data.

WaterML 1.1:

```
<creationTime>
 2015-11-25T12:00:00Z
</creationTime>
<criteria>
 <locationParam>F2S:Marisel</locationParam>
 <variableParam>Marisel:cota_lac</variableParam>
</criteria>
```

WaterML 2.0:

```
<wml2:DocumentMetadata gml:id="dm_1">
 <wml2:generationDate>
  2015-11-25T12:00:00Z
 </wml2:generationDate>
 <wml2:generationSystem>
  Marisel
 </wml2:generationSystem>
</wml2:DocumentMetadata>
```

4.3 Site Info

The data collection site is specified in this block. The exact geolocation is not compulsory, especially because the F2S federation members know each other and can look up the location of a given site in their own database, by name.

WaterML 1.1:

```
<sourceInfo xsi:type="SiteInfoType">
 <siteName>Marisel</siteName>
 <siteCode network="F2S" siteID="1">
  Marisel_1
 </siteCode>
 <geoLocation>
  <geogLocation
   xsi:type="LatLonPointType" srs="EPSG:3844">
   <!-- precision intentionally diluted in
   this excerpt -->
   <latitude>46.6</latitude>
   <longitude>23.1</longitude>
  </geogLocation>
 </geoLocation>
 <!-- precision intentionally diluted in
 this excerpt -->
 <elevation_m>1000</elevation_m>
</sourceInfo>
```

WaterML 2.0:

```
<wml2:MonitoringPoint
 gml:id="monitoring_point_Marisel_1">
 <gml:description>Marisel 1</gml:description>
 <!-- address intentionally changed in this excerpt -->
 <gml:name
  codeSpace="http://193.228.7.104/f2s/eservice/geo">
  Marisel
 </gml:name>
 <sf:sampledFeature
  xlink:href="http://193.228.7.104/f2s/eservice/geo"/>
 <sams:shape>
  <gml:Point gml:id="location_Marisel_1">
   <!-- precision intentionally diluted in
   this excerpt -->
   <gml:pos srsName="urn:ogc:def:crs:EPSG::3844">
    46.6 23.1
   </gml:pos>
  </gml:Point>
 </sams:shape>
 <wml2:verticalDatum
  xlink:href="urn:ogc:def:crs:EPSG::3844"
  xlink:title="Romanian height datum"/>
 <wml2:timeZone>
  <wml2:TimeZone>
   <wml2:zoneOffset>+2:00</wml2:zoneOffset>
   <wml2:zoneAbbreviation>EET</wml2:zoneAbbreviation>
  </wml2:TimeZone>
 </wml2:timeZone>
</wml2:MonitoringPoint>
```

4.4 Begin and End Times

A WaterML document is usually a dynamically generated response to a request, not a file that is simply sent to the client. This is especially significant when dealing with time-stamped data. Depending on the server's data gathering and retrieval capabilities, a client may specify a time window for the data in which it is interested, by supplying the desired begin and end moment, in the request.

4.5 Data Properties

In addition to the sequence of the actual measured values, the WaterML structure should specify some properties of these values such as value identifier, measurement frequency, measurement units – e.g. „water_level", „hourly", „meters above sea level".

4.6 Time Series Data

This is the actual data, in sequence. Table 1 shows the data in tabular format.

Table 1. "Cota_lac" Measurement Sequence.

Timestamp	Measurement unit	Value
2015-11-09T19:45:00+02:00	mdm	123.555
2015-11-09T20:00:00+02:00	mdm	123.565
2015-11-09T20:15:00+02:00	mdm	123.559
2015-11-09T20:30:00+02:00	mdm	123.555
2015-11-09T20:45:00+02:00	mdm	123.555

Timestamps are in ISO 8601 format, as required by the standard.

5. TECHNOLOGIES CONSIDERED

For service development, several available solutions were employed: the well-established ones with efficient software development environments, others extremely fast, and others highly portable.

5.1 NodeJS

The NodeJS platform (Mardan, 2014) supports web service application development using the JavaScript language (Myers, 2014). As an event based system, NodeJS is efficient, scalable and available for a wide range of platforms. The main feature is the JavaScript language, used here on the server side, not as usually, on the client side. NodeJS is promoted both by Google and Microsoft.

The most common use case for NodeJS application development is the combination of tools as: Express library, MongoDB database server and AngularJS library (Freeman, 2014) for client side application.

5.2 Simple Object Access Protocol (SOAP)

SOAP is a way for a web service client and server to communicate requests and results. It is based on XML (eXtensible Markup Language). Since it is standardized, once the client knows the web service's API (Application Programming Interface), it can invoke the service's methods, encapsulating the method call in a SOAP message, to which the server responds with the results, also encapsulated in SOAP.

5.3 Representational State Transfer (REST)

REST is a web-based client-server communication technique. It is faster, leaner, easier to develop for, than SOAP, however SOAP/XML is more robust, more reliable and has better support in RAD (Rapid Application Development) like .NET or Java/Spring.

Since SOAP-based web services are more robust, they are more fitting for the SCADA federation's needs.

6. PROPOSED SOLUTION

6.1 The Web Service's Functionalities

The E-Service's two main functionalities are:

- to retrieve and analyse data, from federative members,
- to accept requests and send the information requested from/toward web clients.

The main structure is presented in Fig. 1.

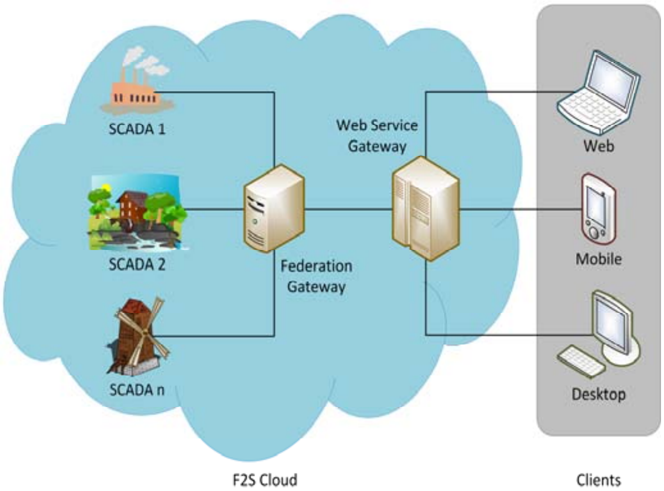

Fig. 1. The system's structure.

This service trades the available information about the federation, with all interested entities. There are several accepted clients; the only condition is to hold authentication data and to send the request in a service identifiable format. For water systems, the Water Markup Language (WaterML) standard is the recommended template for data interchange (*, WaterML Standard, 2017). The language is created by the Open Geospatial Consortium (OGC) especially for transmitting the domain specific information. The E-Service renders data in the WaterML format, through the classic standard for web services - SOAP (Simple Access protocol) (Daigneau, 2011).

6.2 Data Obtained from SCADA Federative Members

The data given by SCADAs included in the federation are received in various formats. The next two tables present an exemplification.

Table 2. Identification Structure Fragment.

Code	SCADA type	Location	GIS coordinates
001	Hidro	SCADA1	

Table 3. "Cota_lac" Data Structure Fragment.

DataHour	Measurement unit	Value
2015-11-09T19:45:00+02:00	mdm	123.555
2015-11-09T20:00:00+02:00	mdm	123.565
2015-11-09T20:15:00+02:00	mdm	123.559
2015-11-09T20:30:00+02:00	mdm	123.555
2015-11-09T20:45:00+02:00	mdm	123.555

6.3 Limiting Access to Data

The clients must authenticate themselves, before gaining access to the data. The data carries an "access level" attribute:

- "owner": visible only to the provider of the data,
- "federative": the data is accessible to the SCADA federation members,
- "external": data available also to federation outsiders, but only with proper login/password authentication.

The client-server authentication and channel encryption is based on WS-Security. The security measures (Nordbotten, 2009) include SAML (Security Assertion Markup Language), WS-Trust, F2S-ScT (Security Token Service).

For fine-grain control over the data access, the system also supports user roles.

6.4 Request Propagation

The client sends a request to the web service gateway, which forwards the request to the federation gateway, then the request arrives to the targeted SCADA systems. The data is then propagated back, towards the client, as shown in Fig. 2.

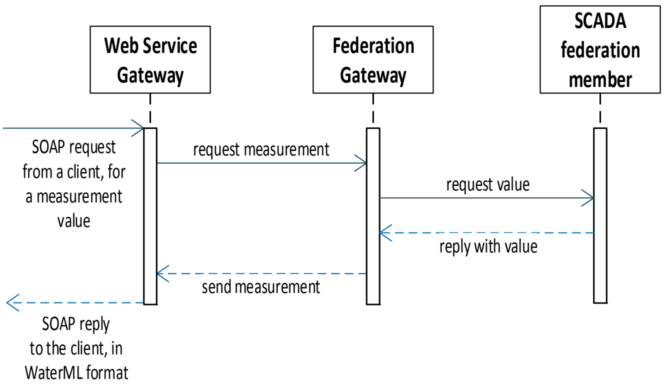

Fig. 2. The request's propagation.

6.5 Data Manually Uploaded by a Human Operator

If a current extreme situation requires, the data could be manually loaded into the system. The SCADA's human operator would be required to upload the file directly on the server. The E-Service will use the updated data to respond to different clients.

The upload is made using a shared restricted folder available on the server, automatically verified for changes. The 15 minute intervals represent an acceptable frequency for upload checks, to avoid server overload. If the data transfer needs a smaller interval, a web interface is available for the human operator to change the duration. The server will immediately process and update the settings.

The server functionalities include several access roles to the file: read, write/change/modify and buffer storage.

One intermediate storage format to which the Excel file is converted, is JSON (JavaScript Object Notation) (*, The JSON Data Interchange Format, 2013). A code fragment for this conversion, on NodeJS platform, is presented below:

```
var xlsx = require('xlsx');
var workbook = xlsx.readFile('scada.xlsx');
var datejson = {};

workbook.SheetNames.forEach(function(foaie) {
  var vectorranduri =
xlsx.utils.sheet_to_row_object_array(workbook.Sheets[foaie]);
  if (vectorranduri.length > 0) {
    datejson[foaie] = vectorranduri;
  }
});
…
```

6.6 The Request's Parameters

Usually, before the data request, the client will apply for available measurements from a specific federation member. The second step represents a request with specific information details:

- which SCADA installation,
- which measurements, one or several,
- which information type (online, historical, report),
- which time interval, for the historical data.

Also, the request contains the authentication parameters to certify the access rights. Fig. 3 shows the handshake and the data interchange procedure.

6.7 Web Service Description Language (WSDL)

The WSDL implementation is one of the available standardized formats for web service description based on SOAP protocol specifications.

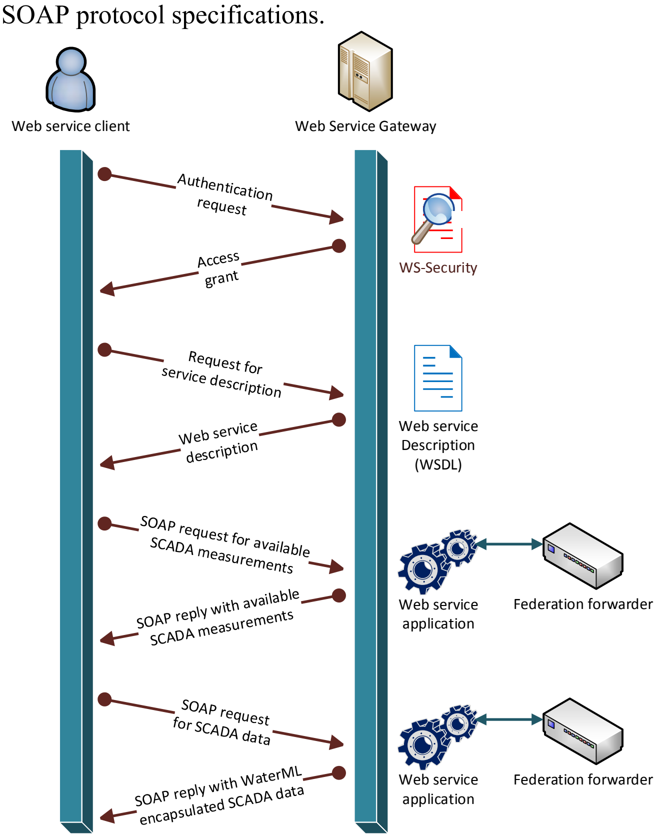

Fig. 3. Client-server handshake and data transfer.

An excerpt from the WSDL 2.0 implementation of E-Service:

```
<?xml version="1.0" encoding="UTF-8"?>
<description
 xmlns:esns="http://197.234.8.90/f2s/eservice/schemas"
 …>
  <documentation>
   This document presents the E-Service within the F2S consortium.
  </documentation>
  <!-- data types -->
  <types>
…
<!-- parameters for service info -->
     <xs:element name="infoservCerere" type="tinfoservCerere" />
     <xs:complexType name="tinfoservCerere">
       <xs:sequence>
         <xs:element name="inIdscada" type="xs:string" />
         <xs:element name="inFormatinfo" type="xs:string" />
       </xs:sequence>
     </xs:complexType>
     <xs:element name="infoservRaspuns" type="tinfoservRaspuns" />
     <xs:complexType name="tinfoservRaspuns">
       <xs:sequence>
         <xs:element name="outVersiuneserv" type="xs:string" />
         <xs:element name="outComenziacceptate" type="xs:string" />
       </xs:sequence>
     </xs:complexType>
     …
<!-- error -->
     <xs:element name="eroareIntrareinvalida" type="xs:string" />
   </xs:schema>
  </types>

   <!-- interface -->
<interface name="eserviceInterfata">
   <fault name="problemaIntrareinvalida"
                   element="esns:eroareIntrareinvalida" />
                 <!-- method for obtaining info about the service -->
   <operation name="opInfoserv"
                   pattern="http://www.w3.org/ns/wsdl/in-out"
                   style="http://www.w3.org/ns/wsdl/style/iri"
                   wsdlx:safe="true">
     <input messageLabel="In" element="esns:infoservCerere" />
     <output messageLabel="Out" element="esns:infoservRaspuns" />
     <outfault ref="tns:problemaIntrareinvalida" messageLabel="Out" />
   </operation>
…
  </interface>
<!-- access protocols -->
<binding name="eserviceLegatura"
    interface="tns:eserviceInterfata"
    type="http://www.w3.org/ns/wsdl/soap"
    wsoap:protocol="http://www.w3.org/2003/05/soap/bindings/HTTP/">

<operation ref="tns:opInfoserv"
      wsoap:mep="http://www.w3.org/2003/05/soap/mep/soap-response"
  />
…
   <fault ref="tns:problemaIntrareinvalida" wsoap:code="soap:Sender" />
</binding>
…
</description>
```

Based on WSDL metadata, the client knows how to formulate its request. Also, the server offers the access interface, based on WSDL, with the global implementation of yielded methods.

6.8 Water Markup Language (WaterML)

WaterML is an XML language, dedicated for water related information encapsulation and transmission using web services. An example of the WaterML 2.0 structure of the E-Service is presented next:

```
<?xml version="1.0" encoding="UTF-8"?>
<wml2:MeasurementTimeseries
  xmlns:wml2="http://www.opengis.net/waterml/2.0" … >
  <!-- document properties -->
  <gml:description>Citire cotă lac.</gml:description>
  <wml2:metadata>
    <wml2:DocumentMetadata gml:id="dm_1">
   <wml2:generationDate>2015-11-25T12:00:00Z
</wml2:generationDate>
      <wml2:generationSystem>Tarnița2</wml2:generationSystem>
    </wml2:DocumentMetadata>
    <wml2:MeasurementTimeseriesMetadata>
      <wml2:temporalExtent>
        <gml:TimePeriod gml:id="cota_lac">
<gml:beginPosition>2015-11-09T19:50:00Z</gml:beginPosition>
          <gml:endPosition>2015-11-09T20:50:00Z</gml:endPosition>
…
  </wml2:metadata>
…
  <!-- the first value in the sequence, with properties -->
  <wml2:point>
    <wml2:MeasurementTVP>
      <wml2:time>2015-11-09T19:45:00Z</wml2:time>
      <wml2:value>123.555</wml2:value>
      <wml2:metadata>
        <wml2:TVPMeasurementMetadata>
          <wml2:quality
xlink:href="http://www.opengis.net/def/waterml/2.0/quality/suspect"
xlink:title="suspect" />
          <wml2:interpolationType />
          <wml2:accuracy>
            <swe:Quantity
definition="http://sweet.jpl.nasa.gov/2.0/sciUncertainty.owl#Accuracy">
              <swe:label>Precizie</swe:label>
              <swe:uom code="%" />
              <swe:value>97</swe:value>
            </swe:Quantity>
          </wml2:accuracy>
        </wml2:TVPMeasurementMetadata>
      </wml2:metadata>
    </wml2:MeasurementTVP>
  </wml2:point>
  <!-- the rest of the values -->
  <wml2:point>
    <wml2:MeasurementTVP>
      <wml2:time>2015-11-09T20:00:00Z</wml2:time>
      <wml2:value>123.565</wml2:value>
    </wml2:MeasurementTVP>
  </wml2:point>
  <wml2:point>
    <wml2:MeasurementTVP>
      <wml2:time>2015-11-09T20:15:00Z</wml2:time>
      <wml2:value>123.559</wml2:value>
    </wml2:MeasurementTVP>
  </wml2:point>
…
</wml2:MeasurementTimeseries>
```

7. IMPLEMENTATION ISSUES

WaterML, although it is the de facto standard for web services that deal with water management, is more resource-intensive to transfer and process, than simpler data structures, like JSON, for example. However, WaterML is self-describing and thus, very robust.

The local SCADA system must be separated from the rest of the federation, preferably even from the outside world altogether (Ştefan et al., 2016). However, we have to access data from the system and make it available to the requesting federation member or outside party.

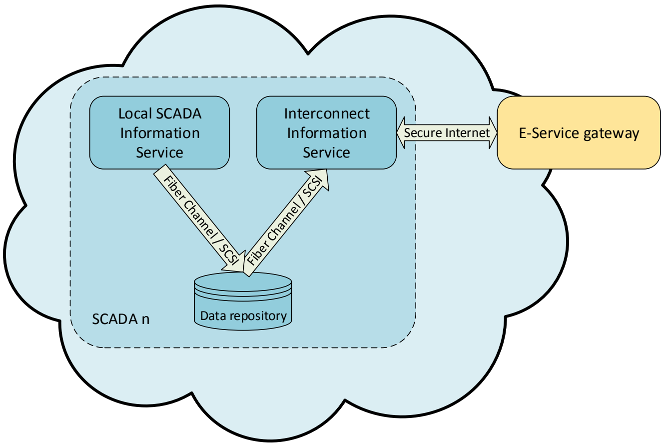

Fig. 4. Security through shared data repository.

In order to regulate the access to SCADA internal data from external actors, we pass the data stream through a shared data repository, with one-way hardware read/write control (Fig. 4). The local SCADA data service has write access to the repository, but cannot read from it, while the federation data service can read from the repository, but cannot write to it.

The data connection, from client to server, is protected with end-to-end encryption using the RSA (Rivest–Shamir–Adleman) algorithm.

An additional security issue was that, even with the proper access credentials, the client should not be able to access the data from unauthorized devices. Therefore, an additional layer of security obliges a digital certificate from the client at the time of initiating the connection (Fig. 5).

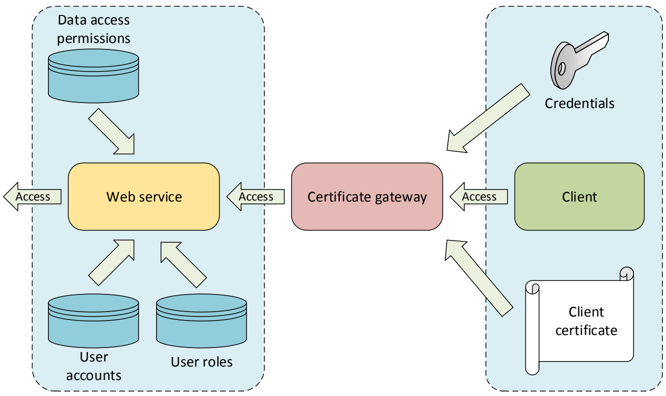

Fig. 5. Security through client certificates

8. CONCLUSIONS

Communication between SCADA systems in a river basin is more efficient when assembled together in a federation. The solution presented here enhances the data exchange, since federative data is synchronized between the client and server automatically and quasi-instantaneously, both procuring fresh data about the federation. The work presented here deals with the web service offered by the federation. This web service is the gateway to the data of the SCADAs within the federation. The clients (federation members or external authorities) may access the data, if they present the correct credentials. Our solution embeds the data access levels in the data objects themselves.

The web service is SOAP based and the data that is sent to the clients is encapsulated in WaterML, the de facto standard for water management web services.

A sample of the client interface to access the data through a request towards the F2S E-Service is presented below, in Fig. 6.

Fig. 6. WaterML Client.

The solution has proved to be robust and it is running in experimental phase since 2016.

ACKNOWLEDGMENT

This work was supported by the Project “F2S”, ctr. no. 2/2014, funded by the Romanian Executive Agency for Higher Education, Research, Development and Innovation Funding (UEFISCDI) through the Partnerships (PCCA) program.

REFERENCES

*. (2007). *ISO/TS 19139:2007, Geographic information – Metadata – XML schema implementation.*

*. (2013). *The JSON Data Interchange Format*. Retrieved 2 12, 2016, from Ecma International, : http://www.ecma-international.org/publications/files/ECMA-ST/ECMA-404.pdf

*. (2014). *ISO 19115-1:2014, Geographic information – Metadata – Part 1: Fundamentals.*

*. (2017). *WaterML Standard*. Retrieved 12 12, 2017, from Open Geospacial Consortium: http://www.opengeospatial.org/standards/waterml

Daigneau, R. (2011). *Service Design Patterns: Fundamental Design Solutions for SOAP/WSDL and RESTful Web Services*. Addison-Wesley.

Dow, A. K., Dow, E. M., Fitzsimmons, T. D., and Materise, M. M. (2015, May 725-736). Harnessing the Environmental Data Flood: A Comparative Analysis of Hydrologic, Oceanographic, and Meteorological Informatics Platforms,. *Bulletin of the American Meteorological Soc*.

Folea, S., and Moiş, G. (2015, Feb.). A Low-Power Wireless Sensor for Online Ambient Monitoring. *Sensors Journal, vol.15*(no.2), pp.742-749.

Freeman, A. (2014). Pro AngularJS. *Apress*.

Hussaina, A., Wua, W., Anzaldib, G., and Abeckerc, A. Implementation of OGC Compliant Framework for Data Integration in Water Distribution System, [. A. (2015). Implementation of OGC Compliant Framework for Data

Integration in Water Distribution System. *Procedia Engineering, 119*, 1366–1374.

Kadlec, J., StClair, B., Ames, D., and Gill, R. (2015, July). WaterML R package for managing ecological experiment data on a CUAHSI HydroServer. *ECOLOGICAL INFORMATICS, 28*, 19-28.

Kaoa, S., Ranatunga, K., Squirec, G., Prattc, A., and Deea, D. (2011, December). Visualisation of hydrological observations in the water data transfer format. *Environmental Modelling and Software, 26*(12), 1767–1769.

Mardan, A. (2014). Practical Node.js: Building Real-World Scalable Web App. (Apress, Ed.)

Moiș, G., Folea, S., Sanislav, T., and Miclea, L. (2015). Communication in Cyber-Physical Systems. *19th International Conference on System Theory, Control and Computing(ICSTCC)* , (pp. 303-307).

Myers, M. (2014). A Smarter Way to Learn JavaScript. *CreateSpace*.

Nordbotten, N. (2009). XML and Web Services Security Standards. *Communications Surveys and Tutorials*, pp. 4-21.

Ramanathan, R., and Korte, T. (2014). Software service architecture to access weather data using RESTful web services. *International Conference on Computing, Communication and Networking Technologies (ICCCNT)*, (pp. 1-8). Hefei.

Ștefan, I., Enyedi, S., Miclea, L., Stoian, I., and Căpățînă, D. (2016). Security Concerns regarding a Federative Web Service. *Proceedings of 2016 International Conference on automation, Quality and Testing, Robotics (AQTR).* Cluj-Napoca.

Stoian, E., Stâncel, S., Ignat, S., and Balogh, S. (2010). Federative SCADA-Solution for Evolving Critical Systems. *Journal of Control Engineering and Applied Informatics (CEAI), 12*, 52-58.

Yu, J., Taylor, P., Cox, S. J., and Walker, G. (2015, September). Validating observation data in WaterML 2.0. *Computers and Geosciences, 82*, 98–110.