IMPROVED XML PROCESSING AND USAGE OF WEB SERVICES IN MATLAB

M. Blaho[1], J. Pálfy[2], F. Schindler[3]

[1]Faculty of Electrical Engineering and Information Technology, [2]Faculty of Informatics and Information Technologies, Slovak University of Technology, Ilkovičova 3, 842 16 Bratislava 4, SK

[3]Faculty of Informatics, Pan European University, Tematínska 10, 851 05 Bratislava 5, SK

{michal.blaho, juraj.palfy}@stuba.sk, frank.schindler@uninova.sk

Abstract

This work deals with exploring XML and Java technologies of the Matlab environment. Introduce more APIs in Java with the ability to use the XML Structure. In Java Standard Edition compares the effectiveness of XML approaches. With Matlab methods simplifies work with an XML data. In Java Enterprise Edition based on web services, create an example to read and write XML with automatically generated Matlab methods.

1 Introduction

Stored data exists in multiple data formats. XML ensures that data is easily transferable among different platforms. To work with the data we need application. Java applications guarantee an independent data handling. Java using XML language to store data, reassures platform independent data processing.

MathWork's Matlab is a high-level language, interactive environment with powerful mathematical software packages. Matlab's environment accelerate the realization of new algorithms. Implementation results of algorithms automatically holds on its own format. Matlab provides the possibility to maintain the scientific data in a data format such as XML, Microsoft Excel, SAS XPort. Methods for working with XML data, are using the Document Object Model. Own methods of facilitating the transformation of the simulation data into Matlab structure.

Java API for XML Processing (JAXP) provides for Java developers application programming interfaces (APIs) for handling XML data. In Java Standard Edition (SE), we compared the parsing performance of XML data with methods Document Object Model (DOM), Simple API for XML (SAX), Streaming API for XML (STAX) Cursor and Iterator STAX.

“As software migrates from local PCs to distant Internet servers, users and developers alike go along for the ride.”[5] Web services based on service-oriented architecture (SOA) are applicable to distributed computing and to connect clients to cloud providers.

With created web service, we have shown the possibility to connect Matlab to cloud providers. Matlab used web services to read and write simulation data to data access tier. According to our knowledge, this approach on the Internet historically has not been documented. Therefore, we decided to describe the principle of Service Oriented Architecture Protocol (SOAP) based web services.

2 XML Fundamentals

There is a wide range of literature dealing with the XML language [1][2][3][18]. We mention XML, because this work deals with its processing. XML is a set of rules for defining semantic tags that break a document into parts and identify the different parts of the document. [4]. XML tags describe the structure of the document and determine significance. XML compared to HTML does not define formatting elements[1], but as a meta-markup language describes its own structure. From the stated example of declarative sentence we captured key data to display them in the XML structure. Earth is a planet in the solar system, which is on the Milky Way in our universe.

1 Document can be formatted with styles fro example CSS, XSLT or XSL-FO.

```
<universe>
  <galaxy name="Milky Way">
    <solar-system>
      <planet name="Earth" />
    </solar-system>
  </galaxy>
</universe>
```

The root element <universe> packed the entire structure of XML document. Contains the additional sub-elements. Element <galaxy> contain attribute pair named *name* and value of the Milky Way. Empty element using the attribute name <planet> for data storage. Empty elements in term of computer networks, has an appropriate structure for transferring XML data. On example were shown the main concepts of XML. Article [12] broken down in detail the XML in the context of relational databases. [13] are recommended cases to use XML to store data in a relational database.

3 Java Development Kit and XML

Java Development Kit (JDK v1.6) provide two main approaches for parsing XML, on the basis of stream and tree-based. Stream based parsers are much more efficient at translating large XML files, which requested for large size memory space. But in terms of usability of tree parser can with greater application much easier to work. Provide a complete view of data, while the current parser gives only limited information [7].

One of the most important part of any XML applications is XML parser. This component saves an important task in taking the input XML document and understand the document. Parser ensure that the document was formatted correctly. In case if DTD or schema is referenced, the parser will ensure verification of the validity of the document. The result of parsing XML document is usually a data structure that can be manipulated and controlled by other XML tools or Java API. To use XML data, it should be noted that the parser is one of the cornerstones [9]. Prompted by the diverse access requirements of various applications, different parsing approaches have evolved to satisfy these requirements [18].

- DOM[2] parsing
- Push parsing (SAX 2.0[3])
- Pull parsing (StAX[4] Cursor API and STAX Iterator API)

3.1 Progress for obtaining test XML data

Freely accessible database (employees_db[5]) appropriate for testing application and database servers, we have exported to XML format. From exported database, we received a number of XML files. XML file employees_salaries.xml (472 MB), we used to create 4 XML files. Between generated XML files was a tenfold size difference (100 kilobyte, 1 megabyte, 10 megabyte and 100 megabyte). Example shows an XML fragment of 100 kilobytes salary_100KB.xml file:

```
<?xml version="1.0" encoding="utf-8" ?>
<employees>
  <salaries>
    <emp_no>10001</emp_no>
    <salary>60117</salary>
    <from_date>1986-06-26</from_date>
    <to_date>1987-06-26</to_date>
  </salaries>
 . . .
```

2 DOM – Document Object Model, http://www.w3.org/TR/DOM-Level-3-Core/

3 SAX - Simple API for XML, http://www.saxproject.org/

4 StAX – Streaming API for XML

5 Patrick Crews a Giuseppe Maxia - Employee DB -https://launchpad.net/test-db/

After the end element </ salaries> are three points, that represent other elements <salaries> filled with employees annual income. Data packed into elements are homogeneous. This means low turnover during the parsing of XML data elements.

3.2 Comparison of parsing approaches

Our aim was to investigate the parsing performance from the perspective of time in the Java environment. Performance of Java approaches, DOM, SAX, StAX Cursor and Iterator APIs was 100 times measured the parsing time of individual XML file. We have repeated parsing time of XML files with sizes 100 kilobyte, 1 megabyte, 10 megabyte and 100 megabyte. From observed temporal data, we created a basic statistical analysis.

The diagram (Figure 1) are presented the test results. SAX and visibly StAXC achieve short times during the parsing of 100 megabyte XML document. StAXI is weaker than SAX and StaXC, but still an average of 43 % faster than the DOM approach. The big difference of minimum and maximum values DOM access is influenced by the application memory space *heap*[6]. Results of parsing smaller XML files showed similar results, are available at [14].

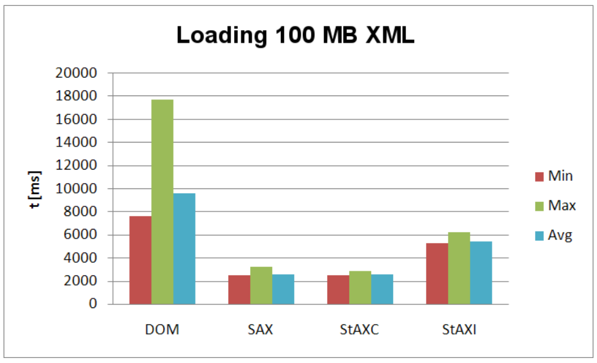


Figure 1: Minimal, maximal and average values of loading 100 megabyte XML file.

	100 KB	1 MB	10 MB	100 MB
SAX	2	26	254	2549
StAXC	2	26	255	2568
StAXI	6	62	547	5483
DOM	8	104	775	9614

Table 1: Average values of loading XML files in [ms].

Table. 1. show average values. In the first column are named parser approaches. The first line is the size of individual XML files. StAXC to 1 megabyte XML file according to the averages is as effective as SAX, which is intended only for reading XML data. In the case of 10 megabyte and larger XML file, SAX approach begins to be more effective than Cursor STAX API. StAX Iterator API used *event objects* for loading XML document.

In our case, for reading the XML file STAX cursor API, which uses low-level access to XML structure was always more powerful. DOM is the slowest way of parsing XML document, this approach also uses Matlab. Comparing the efficiency of loading XML documents, we have shown in the same environment and with relatively homogenous files with different sizes.

As a general rule, there is no best parsing approach. But exists a satisfactory parsing access for concrete problem [14].

3.3 XML methods for Matlab

Matlab is supporting many files types currently for example text, audio, video or spreadsheets. XML file format is supported too. The two functions xmlread and xmlwrite works only with Document Object Model (DOM) that represents of the current document defined by the World Wide Web consortium. Users can access all elements and attributes of the DOM trough Java API for XML Processing (JAXP). The less experienced user can have trouble with this, therefor Matlab like functions are necessary. In Matlab User's Guide [15] under xmlread function is example of functions that parse data from an XML file into a MATLAB structure array.

6 All Java objects are stored in the memory space heap. Heap is the amount of unused memory allocated for Java applications. Instantiating objects in the heap, takes a lot of CPU time.

We used source from this functions to create function that has more “Matlab” name xml2struct. Similar functions for writing structure data to XML file is not to find in User's Guide. For fast data export to the XML files from Simulink schemas we created own function simstruct2xml, which exports data from Simulink block To Workspace with the structure data format. With also found third party project (toolbox) for working with XML files. They can be useful for less experienced Matlab users to work with XML files easily [9].

Example:

```
>> simstruct2xml('testfile.xml',simout,{'sine','saw'})
<?xml version="1.0" encoding="utf-8"?>
<simout>
  <sine>0,0.84147,0.9093,0.14112</sine>
  <saw>1.165,0.62684,0.07508,0.35161</saw>
</simout>
```

4 Interaction Matlab with Web Services

Developed task by using Java technologies EE[7], JAX-WS[8], SOAP[9], WSDL[10]. Theoretical assumptions to perform this task are available in manual Matlab [16] and in the literature that deals with those technologies [4][6][8][17]. We used a model of three-tier architecture for context our program code.

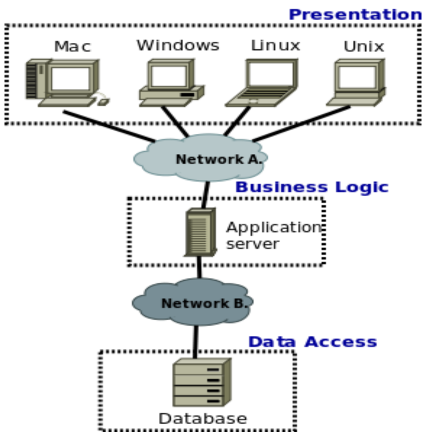


Figure 2: Three-tier architecture.

In Presentation tier (Figure 2) Matlab 2009a task was carried out thin client, which using SOAP messages to consume a web service. Application server Glassfish v3 Prelude was in Business Logic tier where provided his web services using HTTP and SOAP (HTTP for web page, and SOAP to send and receive data). Data Access tier generally is not visible to clients in Business Logic tier. PostgreSQL 8.4 database on the data access tier, communicates with application server and provide storage for XML data. Freely available PostgreSQL database supports data type *xml*[11]. XML data type in Java simplifies work with XML data.

---

7 Java EE - Java Enterprise Edition

8 JAX-WS - Java's API for XML Web Services.

9 SOAP - Tentatively called Simple Object Access Protocol, but its actual name is Service Oriented Architecture (SOA) Protocol.

10 WSDL - Web Service Definition Language

11 Commercial databases, Microsoft SQL Server 2008 and Oracle 11g supports xml data type. Popular non commercial MySQL database don't support xml data type [10].

4.1 Nastavenie prostredia

After the installation of individual components of three-tier architecture, we need to configure them. For management Postrgre SQL[12] database, we use pgAdmin v1.10 application. Follows is a SQL script that we used to create a table *simulations*.

```
CREATE TABLE simulations
( simul_id serial NOT NULL,
  simulation xml,
  simul_date timestamp without time zone NOT NULL,
  CONSTRAINT simul_pk PRIMARY KEY (simul_id) )
```

After creating a database, on application server we need to set up access to the database. First, we configure JDBC[13] Connection Pools for PostgreSQL database. Next we defined the *data source* on an application server. Using data source, Java application was able to generate a JDBC data source on the basis of Connection Pool [17].

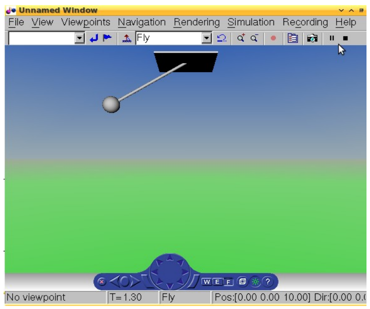

Unnamed Window
File View Viewpoints Navigation Rendering Simulation Recording Help
Fly
No viewpoint
T=1.30
Fly
Pos:[0.00 0.00 10.00] Dir:[0.00 0.0

Figure 3: Graphical interface pendulum simulation.

Figure 3 shows a pendulum simulation in Simulink. Simulation was used to obtain data. In a next step obtained data transformed to XML document. The picture is graphically illustrated the movement of the pendulum.

4.2 Matlab SOAP Client

4.2.1 Automatic generation of Matlab SOAP methods

There are two basic ways to access the web services in Matlab. Using createClassFromWsdl features or using SOAP functions [15]. The following list contains the SOAP functions:

- createSoapMessage - constructs a message to send to the server.
- callSoapService - sends a message to the endpoint (server).
- parseSoapResponse - receives a message and make data conversions.

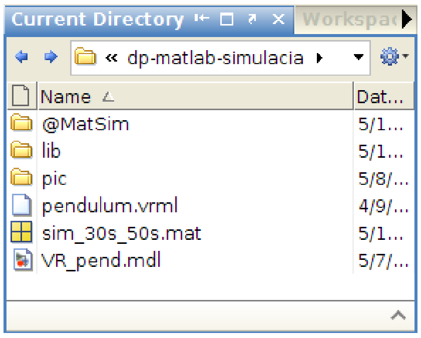
We use first approach with Matlab SOAP client. With function createClassFromWsdl we automatically generate web services methods.

```
>> createClassFromWsdl('http://localhost:8080/edu-ws-soap/MatSim?wsdl')
Retrieving document at 'http://localhost:8080/edu-ws-soap/MatSim?wsdl'.
Retrieving schema at 'http://localhost:8080/edu-ws-soap/MatSim?xsd=1',
relative to 'http://localhost:8080/edu-ws-soap/MatSim?wsdl'.
```

createClassFromWsdl created a new directory called @MatSim (Figure 4a). To the directory @MatSim, automatically saved Web services functions (Figure 4b).

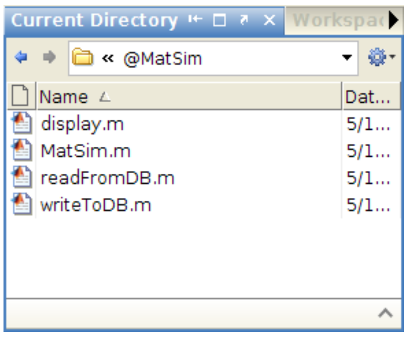
12 Book [18] deals with detailed description of PostgreSQL database.

13 JDBC - Java Database Connectivity .



Name	Dat...
@MatSim	5/1...
lib	5/1...
pic	5/8/...
pendulum.vrml	4/9/...
sim_30s_50s.mat	5/1...
VR_pend.mdl	5/7/...

Figure 4a: Created a new directory @MatSim.



Name	Dat...
display.m	5/1...
MatSim.m	5/1...
readFromDB.m	5/1...
writeToDB.m	5/1...

Figure 4b: Directory content @MatSim.

Figure 4: createClassFromWsdl function, created the directory and its contents.

4.2.2 Error correction parts of Matlab SOAP methods

After setting three-tier architecture, between Matlab SOAP client and GlassFish application server was incorrect communication. We wrote Java and Perl clients for testing SOAP services at application server. Java and Perl clients were able to communicate with web service.

On the application server we analyzed log file server.log[14] and used Wireshark[15] v1.2.22 to capture network traffic (SOAP messages). On analysis of network packages and log file, we found that Matlab SOAP client incorrectly communicates with an application server.

That generated Matlab SOAP methods operated, it is necessary to change them 'document' to 'rpc'. To eliminate the error messages, should be added *SOAP Action* value with double quotes (eg *'urn:readFromDB'* rewrite to *'"urn:readFromDB"'*).

4.2.3 Send and receive simulation data

To use Web services methods, it is necessary to create a new instance of the object.

```
>> obj = MatSim
   endpoint: 'http://localhost:8080/edu-ws-soap/MatSim'
       wsdl: 'http://localhost:8080/edu-ws-soap/MatSim?wsdl'
```

At this point SOAP servers web services are available. In Figure 4b shows the 4 files. CreateClassFromWsdl always creates a constructor (MatSim) and display method (display) class. ReadFromDB method is intended for reading from the database. Method writeToDB XML facilitate, writing data into the database. An example using web services loads XML data with record number 32. ReadFromDB result in this case was shown on the screen (it can also be stored in a variable).

```
>> readFromDB(obj,32)
```

Sending XML data via web services. In the first line with data2xml program we created from simulated data XML document (optimized for transmission over a network). XML document was stored in a variable xmlString.

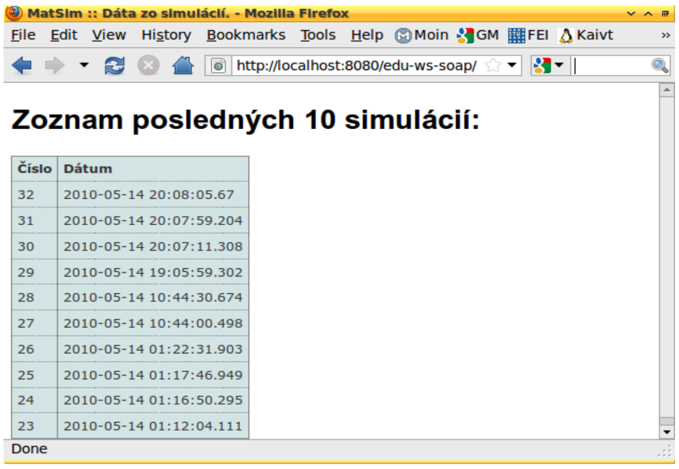
```
>> xmlString = data2xml.toChar(angle_1(1:100),caction_1(1:100),
                    time_1(1:100),tout_1(1:100))
xmlString =
<?xml version='1.0' encoding='UTF-8'?><simulation><!--
- angle : uhol.
- caction : riadiaci zasah.
- time : cas.
- tout : cas zo Simulinku.-->
```

14 Path from the root directory of Glassfish log file: glassfish/domains/domain1/logs/server.log

15 Wireshark - available at http://www.wireshark.org/download.html

<data angle="90.0" caction="0.0" time="0.0" tout="29.900171557343093" />...
</simulation>
>> writeToDB(obj,xmlString)

In the last line we send XML data to the SOAP server. After successful completion of operations the server sends a message “Simulacia bola ulozena”.



MatSim :: Dáta zo simulácií. - Mozilla Firefox
File Edit View History Bookmarks Tools Help Moin GM FEI Kaivt
http://localhost:8080/edu-ws-soap/
Done

Zoznam posledných 10 simulácií:

Číslo	Dátum
32	2010-05-14 20:08:05.67
31	2010-05-14 20:07:59.204
30	2010-05-14 20:07:11.308
29	2010-05-14 19:05:59.302
28	2010-05-14 10:44:30.674
27	2010-05-14 10:44:00.498
26	2010-05-14 01:22:31.903
25	2010-05-14 01:17:46.949
24	2010-05-14 01:16:50.295
23	2010-05-14 01:12:04.111

Figure 5: Number and date of the last 10 measurements.

Using Java Server Pages, we show the number and time-stamp of the last 10 measurements (Figure 5). For clarity of table, we are not seeing XML data from simulations.

5 Software and Hardware Equipment

Development machine included an operating system Linux with Ubuntu 9.10 distribution. We used the Eclipse 3.5.2.R35x_v20100119 integrated development environment. For comparison, parsing approaches, we used Java Standard Edition 1.6. In Chapter 4 We worked on platform Java Enterprise Edition 6, application server Glassfish v3 Prelude, Matlab 2009a and PostgreSQL database V8.4.

The computer contained the physical components Intel Pentium Core 2 Duo 2.0 GHz, 1 GB memory and 160 GB SATA hard disk.

6 Conclusion

In this paper, we have demonstrated the value of XML in Matlab environment. We tested the effectiveness of approaches loading XML files. To characterize the observations and time values, we used basic statistical analysis. Statistics results were used to compare the effectiveness of different approaches. Comparing the results we showed that the SAX API and the StAX Cursor API are ideal for loading XML files with different sizes.

In Matlab, we used our method to show simplified XML data handling. We present set of three-tier architecture. We employed source code for the web service interface, database and website. Web service with multiple attempts of Matlab were inaccessible. After analyzing the network packages, we found that Matlab 2009a sent not valid SOAP messages for web service. After correction of the erroneous Matlab generated SOAP library and finding a compatible configuration of SOAP messages on application server, Matlab SOAP client started to consume the Web services provided by application server. Stored simulation data were accessed on application server via dynamically generated web page (Java Server Page.

Contribution of our article is own functions for simplify work with XML and simulation data, comparing the efficiency of XML approaches, on the Internet up to the present historically undocumented web services for Matlab SOAP client.

Acknowledgement

This contribution was supported by Grant Agency VEGA SR under the grant 1/0141/10.

References

[1] BENZ, B. – DURANT, J. R. 2003. XML Programming Bible. Indianapolis : Wiley Publishing, 2003. ISBN 0764538292

[2] DYKES, L. – TITTEL, E. 2005. XML For Dummies® , 4th Edition. Indianapolis : Wiley Publishing, 2005. ISBN-13 9780764588457

[3] Extensible Markup Language. 1996. [online] W3C [World Wide Web Consortium], Revised September 6. 2010 04:04:22. [Cited on September 27. 2010]. Available at: <http://www.w3.org/XML/>

[4] HAROLD, E. R. 2004. XML 1.1 Bible, 3rd Edition. Indianapolis : Wiley Publishing, 2004. ISBN 0764549863

[5] HAYES, B. 2008. Communications of the ACM Volume 51, Number 7 (2008), Pages 9-11. [online]. ACM [Association for Computing Machinery], [Cited on October 1. 2010], Available at: <http://portal.acm.org/>

[6] HORSTMANN, C. S. 2008. Big Java, 3rd Edition. USA : John Wiley & Sons, 2008. ISBN 9780470105542

[7] MCLAUGHLIN, B. 2001. Java & XML, 2nd Edition. Sebastopol : O'Reilly, 2001. ISBN 0596001975

[8] MATTHEW, N. - STONES, S. 2005. Beginning Databases with PostgreSQL: From Novice to Professional, Second Edition. USA : Apress R , 2005. ISBN 1590594789

[9] Molinari, M. 2003. Matlab Central - XML Toolbox, 2003. [online]. The MathWorks, Natick, MA, USA, 2003 Revised 2005. [Cited on 06.10.2010], Avaible at: <http://www.mathworks.com/matlabcentral/fileexchange/4278>

[10] MySQL 5.5 Reference Manual. 2010. [online]. Oracle. [Cited on October 4. 2010]. Available at: <http://dev.mysql.com/>

[11] PÁLFY, J. 2008. Spracovanie XML dokumentácie v MS SQL. Bratislava : FEI STU, 2008.

[12] PÁLFY, J. - SCHINDLER, F. 2009. XML v prostredí SQL servera 2008. In ŠAGÁTOVÁ, A. - STACHO, M. - PAVLOVICOVÁ, J. ŠVOC 2009 : Študentská vedecká a odborná cinnost'. Zborník vít'azných prác. Bratislava, Slovak Republic, 29.4.2009. Bratislava: STU v Bratislave FEI, 2009, ISBN 9788022730945

[13] PÁLFY, J. 2010. Spracovanie XML s Javou v prostredí MS SQL. Bratislava : FEI STU, 2010.

[14] The MathWorks. *Getting Started Guide*. 2008

[15] Using Web Services with MATLAB, 2010. [online]. The MathWorks, Inc., [Cited on April 25. 2010]. Available at: <http://www.mathworks.com/>

[16] VASILIEV, Y. 2008. Beginning Database-Driven Application Development in JavaTM EE: Using GlassFishTM . USA : Apress R , 2008. ISBN 9781430209645

[17] VOHRA, A. - VOHRA, D. 2006. Pro XML Development with JavaTM Technology. USA : Apress R , 2005. ISBN 9781590597064

[18] Using Web Services with MATLAB. 2010. [online]. The MathWorks, Inc., [Cited on October 7. 2010]. Available at: <http://www.mathworks.com/>