AASCIT Communications
Volume 3, Issue 2
February 24, 2016 online
ISSN: 2375-3803

American Association for Science and Technology

Integration of Data Using Java for Web Applications

Dănuț-Octavian Simion Economic Informatics Department, Athenaeum University, Bucharest, Romania

Keywords

Framework, Java Objects Integration, XML Files, Open-Source Java Components, UML Diagram, Web APPLICATIONS

The paper presents the integration of data using Java for Web Applications in the open-source domain and emphases aspects like: integration of data that will be used in Web Applications will be through a Java custom driver, the usage of XML files as a common platform for the integration of data and through the E. T. L. - Extract, Transform and Load instruments used to integrate data for data warehouse. The next step is to integrate open-source Java components through the presented frameworks into Web Applications. There will be presented aspects for the development of existing Java objects with open-source components that will improve performance and facilities of existing tools used to access data from different DBMS-s or XML files. Encapsulation of data in these objects permits to set and change values and also to get the appropriate values by using listeners and methods specific to application levels.

Introduction

Connecting users to database would not be possible without implementing connect drivers that are specific to each DBMS's, including in this technology which is very complex. Among the best drivers are implemented in Java, offering multiple facilities and possibilities of customization required different types of Web applications. An important role in these applications have Java beans that are reflecting the entity database, transacts executes user commands and their use in technologies such as Java RMI (Java Remote Method Invocation - is an API (Application Programming Interface) equivalent actions performed by Remote Procedure Calls), CORBA - Common Object Requesting Broker Architecture – which is a standard defined by the Object Management Group that allows software components written in different languages and it is running on different machines to work together [2], [5].

Integration of Data That Will Be Used in Web Applications

An important contribution in building Web applications for DBMS's, by highlighting the best solutions for building these applications taking into account the costs, risks, robustness DBMS the chosen power programming language facilities of different frameworks plants, the ability to offer customers applications to access the Web interface, and modifying data. My recommendation is to use the DBMS, a DBMS new generation that includes powerful storage facilities, handling, connection, protection, etc. (a DBMS such as Oracle), the Java programming language - it is an object oriented language easily use includes powerful features for the Web, specific classes to connect to databases, does not depend on the operating system and allows portability increased and the possibility to split tasks between Web designers, Java developers, software architects, software analysts and specialists in the field of business and open source frameworks such as Eclipse. My recommendation is to use LAMP technology (Linux, Apache HTTP Server, MySQL (database software) and PHP - programming language) as solutions open-source, free and easy to use and implemented by a lot of small and medium enterprises. For firms overall dimensions are recommended technologies such as DBMS - Oracle Linux - operating system, application server - OAS (Oracle Application Server) and Java as a programming language to build Web applications robust, stable, which can be accessed used by a large number of users via a Web browser. [1], [3], [8].

The proposed solutions are to build a custom driver connection using the Java programming language (a class of its own is included in an overall package, a class that can be extended by specific methods of working with databases and can be developed further and other programmers, a class that allows a client to connect to any database server through sockets). Another solution is to use XML files that contain structured data and XML descriptor files needed to access Web services data. This solution is easy to implement since most new generation DBMS sites allow data transfer or import XML files from these files.

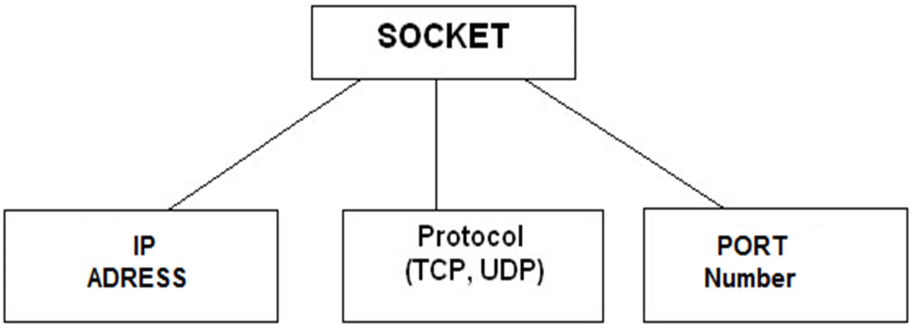
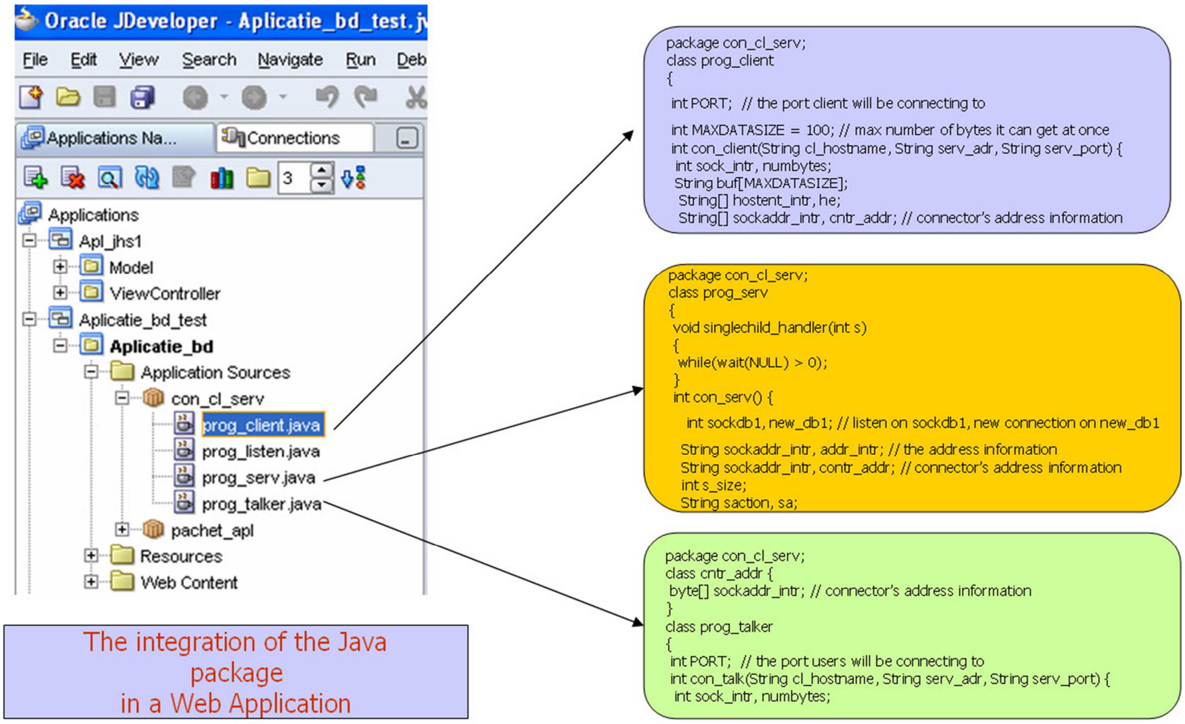


Fig. 1. *The structure of the SOCKET.*

The platform of XML files is the future in working with data in the Web and will be developed and adopted by most manufacturers of software solutions. Another solution is to build data warehouse solutions that can contain data of different types under different DBMS sites or semi-structured files. This data can be integrated into data warehouses via ETL tools (Extract, Transform and Load). Extracting data from outside sources, transforming them to match the architecture of data warehouse and loading them into a database of its kind, allows the use of these data at a higher level can be queried through superior tools for data mining, OLAP (Online Analytical Processing) and not least be represented in reports and graphs for significant business process. In this area is leading Oracle Warehouse Builder and Oracle technology as a tool for query, reporting, data mining is Oracle Business Intelligence Suite Enterprise Edition [3], [6], [9].



```
package con_cl_serv;
class prog_client
{
 int PORT;  // the port client will be connecting to
 int MAXDATASIZE = 100; // max number of bytes it can get at once
 int con_client(String cl_hostname, String serv_adr, String serv_port) {
  int sock_intr, numbytes;
  String buf[MAXDATASIZE];
   String[] hostent_intr, he;
   String[] sockaddr_intr, cntr_addr; // connector's address information
```

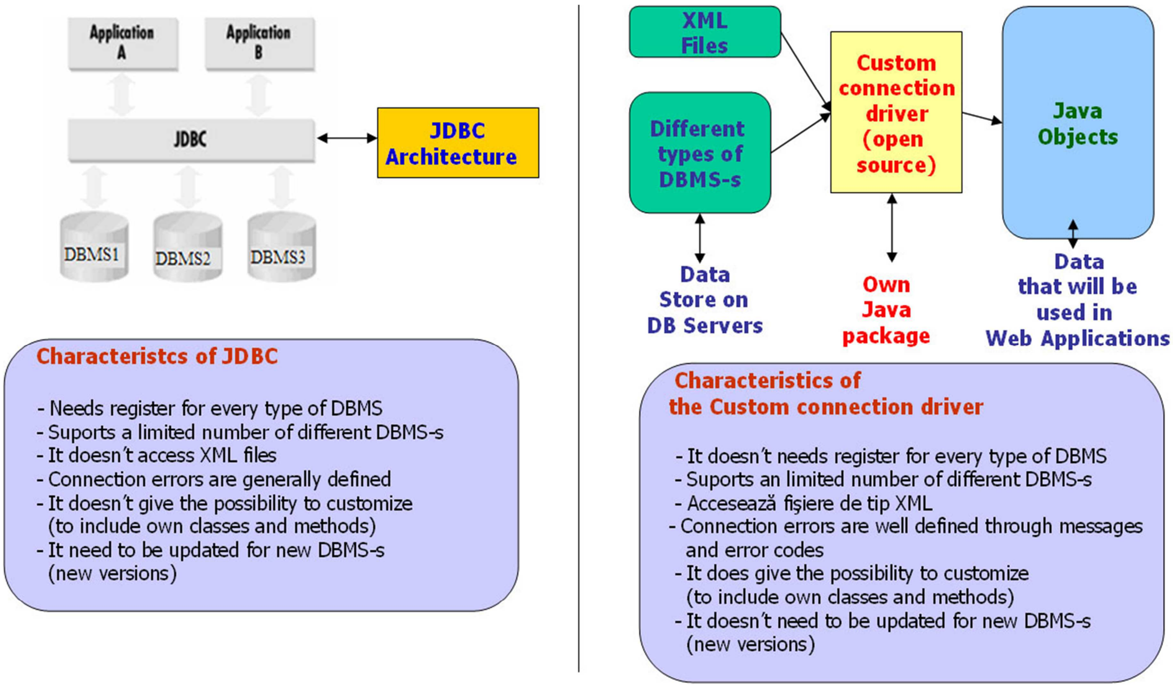
```
package con_cl_serv;
class prog_serv
{
 void singlechild_handler(int s)
 {
  while(wait(NULL) > 0);
 }
 int con_serv() {
   int sockdb1, new_db1; // listen on sockdb1, new connection on new_db1
  String sockaddr_intr, addr_intr; // the address information
  String sockaddr_intr, contr_addr; // connector's address information
  int s_size;
  String saction, sa;
```

```
package con_cl_serv;
class cntr_addr {
 byte[] sockaddr_intr; // connector's address information
}
class prog_talker
{
 int PORT;  // the port users will be connecting to
 int con_talk(String cl_hostname, String serv_adr, String serv_port) {
  int sock_intr, numbytes;
```

Fig. 2. *The integration of the open-source code in a Web Application.*

The results are reflected in the construction of a general driver connection that can be used within Web applications for DBMS sites so that customers can connect to any database server through socket OF THE. This driver was designed and made in the programming language object-oriented Java, so it is open-source, free, independent of the operating system can be improved by the inclusion of methods and specific classes work with databases, can be used any type of Web application and DBMS can be customized sites and other Java programmers to meet the requirements of their programs. [4], [5].

The results are based on studies that consider issues such as tests related to obtaining a stable connections through driver connection, tests concerning packet loss data streams of bytes of faults due to network which links between client and server errors as services talker-listener offered server failures caused by misuse of driver connectivity to the customer using methods Java inside the package without their arguments or arguments that represent different types of data other errors due to communication or mainframe systems to Web services used. Were also taken into account the cost of implementing such a solution, the response time in the business of transmitting-receiving packet data connection stability and permanence related targets throughout use in applications this custom driver connection. The results of this research led to the conclusion that this general connection driver is stable, robust, delivering faster access and lower response (it's a fast driver), provides a permanent connection across a web application.



Characteristcs of JDBC

- Needs register for every type of DBMS
- Suports a limited number of different DBMS-s
- It doesn't access XML files
- Connection errors are generally defined
- It doesn't give the possibility to customize (to include own classes and methods)
- It need to be updated for new DBMS-s (new versions)

Characteristics of the Custom connection driver

- It doesn't needs register for every type of DBMS
- Suports an limited number of different DBMS-s
- Accesează fişiere de tip XML
- Connection errors are well defined through messages and error codes
- It does give the possibility to customize (to include own classes and methods)
- It doesn't need to be updated for new DBMS-s (new versions)

Fig. 3. The comparison between JDBC driver and custom connection driver.

In the following figure are presented the principal characteristics of the custom connection driver:

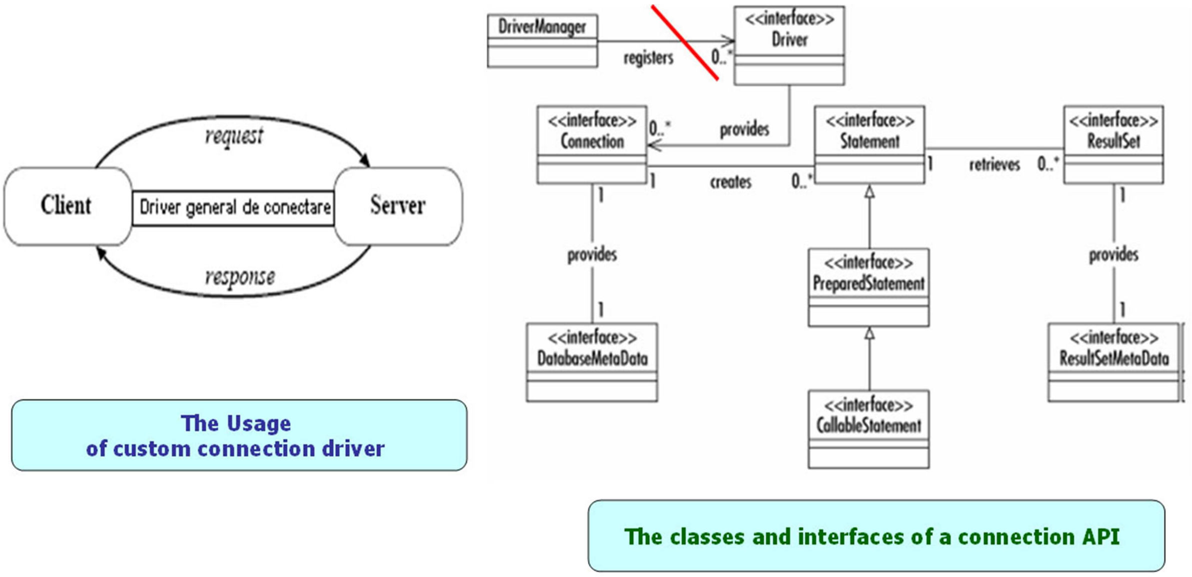

request
Client
Driver general de conectare
Server
response
The Usage of custom connection driver
DriverManager
registers
0..*
<<interface>> Driver
<<interface>> Connection
0..*
provides
<<interface>> Statement
1
retrieves
0..*
<<interface>> ResultSet
1
creates
0..*
1
provides
1
1
provides
1
<<interface>> PreparedStatement
<<interface>> DatabaseMetaData
<<interface>> ResultSetMetaData
<<interface>> CallableStatement
The classes and interfaces of a connection API

Fig. 4. The characteristics of the custom connection driver.

Another modality to access different types of data stored in DBMS-s or semi-structured data stored in files is through XML files that offer a unique standard to structure data, basically established a template for data, structured in records (rows) or pairs like description-values. Almost all DBMS-s have the facility to transform data into XML files that represent an international standard in the IT domain, and so the data become available to a large number of programming languages.

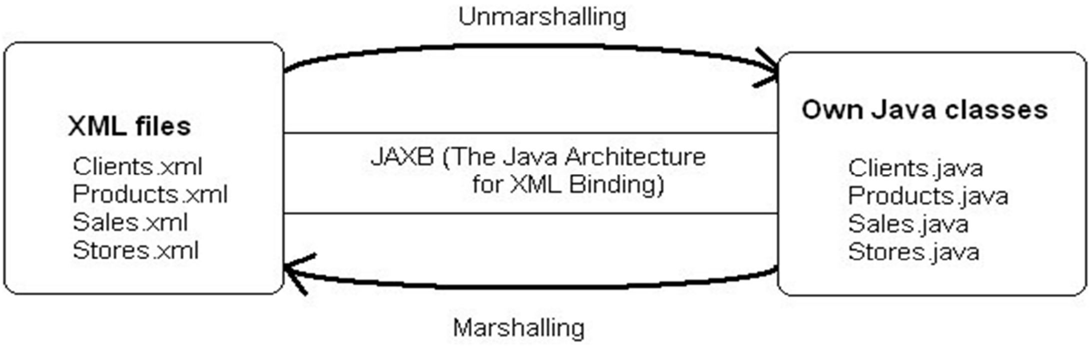


Fig. 5. *The usage of JAXB class for XML files.*

The link between Web Applications and these types of files can be made through Java programming language that contain packages dedicated for accessing those files and the transformation of data in own Java classes and also the possibility of transforming data from Java classes into XML files that are different as form and content from the original files [6], [10].

Also, another possibility to access data store in different DBMS-s or semi-structured data stored in files is through the transformation and loading of those in a unique DBMS that is very efficient, by the usage of E. T. L. - Extract Transform and Load tools and building data warehouse. The advantage of building a data warehouse is very important, even though the processes for loading and transforming data needs a significant programming effort, but the reunion of those in a unique DBMS allow the integration of data into complex applications that offers many possibilities to interrogate data and data mining with low resources of times in comparison with other applications [1], [4], [7].

Another important aspect is referring at the usage of ETL (Extract, Transform and Load) instruments for data warehouse, by providing to the programmer powerful tools for data mining, OLAP (Online analytical processing), reporting, etc. Other aspect refers to the usage of ETL instruments of data warehouse, by providing to the programmer extraction of data from outside sources, transformation of those to correspond with the data warehouse architecture and loading of those in a database of the same type, permits the usage of those data at a superior level and they can be interrogate through superior instruments for data mining, OLAP (Online analytical processing) and they can be represented in reports and significant graphics for the business process. In this domain an important role has Oracle with the technology Oracle Warehouse Builder and as an instrument is OBIEE-Oracle Business Intelligence Suite Enterprise Edition.

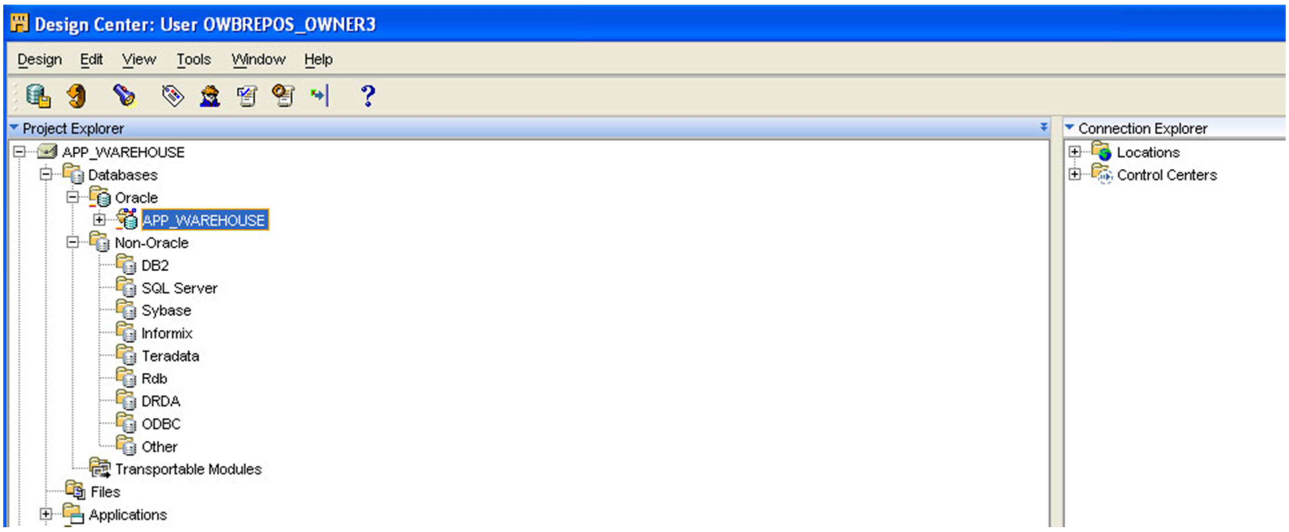


Fig. 6. *ETL instruments for the integration of data.*

The structuring of those in cubes and dimensions makes possible the storage of data in a unique mode after the data have been validated and transformed through specific procedures and their aggregation in comparison with the time dimension permits the building of comparative reports very complex.

These open source methods used for the integration of data can be improved by another programmer or groups of programmers by adding new classes and methods or by integrating in their own Web applications the existing objects build on XML files or by integrating the existing data warehouse in more complex star schemas of a bigger data warehouse.
To implement a business object it is recommended to use a Java object that can implement methods and properties specific to an entity. The following code is a template for an entity bean:

```
public class Temp_Entity_Bean implements java.io.Serializable
{
protected int a_Specific_Value;
public Temp_Entity_Bean ()
{
}
public void set a_Specific_Value (int new_Specific_Value)
{
a_Specific_Value = new_Specific_Value;
}
public int get_Specific_Value ()
{
return a_Specific_Value;
}
}
```

The Java beans may change values through events that are specific to the business logic. The following code shows the changes in sales for a product:

```
public class Sale_Change_Event extends java.util.EventObject
{
//the new sale value
protected double the_Sale_Value;
//the constructor
public Sale_Change_Event (Object source, double sale_Value)
{
super(source);
// save the new sale value
the_Sale_Value = sale_Value;
}
//get the sale value
public double get_Sale_Value ()
{
```

```
return the_Sale_Value;
}
}
```

It is mandatory to implement an event listener like in the following code:

```
public interface Sale_Change_Listener extends java.util.EventListener
{
//the method is called when the sale value is changed
void sale_Value_Changed(Sale_Change_Event evt);
}
```

In the source code bellow is shown how properties can be changed:

```
public class Sales_Custom extends Property_Change_Support
{
// the current sale value
protected double current_Sale = 150;

public Sales_Custom (double start_val_Sale)
{
this();
current_Sale = start_val_Sale;
}
public Sales_Custom ()
{
super(this);
}
// the get method
public double get_Current_val_Sale ()
{
return current_Sale;
}
// notify listening objects of Current_val_Sale property changes
protected void notify_Current_val_Sale ()
{
// fire the event
fire_Property_Change("Current_val_Sale ",
null, new Double(current_Sale));
}
}
```

The usage of these Java type of object can improve the performance of an application and can made the data more flexible and available to a large range of different users. Encapsulation of data in these objects permits to set and change values and also to get the appropriate values by using listeners and methods specific to application levels.

Conclusions

The integration of data is an important aspect because the programmer can use data from different types of sources like different DBMS-s, XML files or star schema already built [1], [4]. The open-source code made the implementation of these methods very easy and makes possible the customization afterwards by other groups of programmers and so the Web Application becomes open-source applications. The custom driver presented above can be used in different types of Web Applications, also may be improved by other programmers through the implementation of their own classes and methods and also may extend the current facilities of the JDBC driver [2], [3]. Other examples of the usage of own Java classes obtained from XML files is possible because other programmers can customize these classes according with their own specification or they can transform these classes into different type of XML files that are different from the original files. ■

Dănuţ-Octavian SIMION

Dănuţ-Octavian SIMION has a background in computer science and is interested in developing business applications and enterprise web applications. He is a PhD in Economic Informatics since 2011(title obtained at Faculty of Economic Cybernetics, Statistics and Informatics from the Bucharest Academy of Economic Studies). He is currently a PhD Lecturer at Athenaeum University, Bucharest, Romania. Other fields of interest include software development, data structures, database programming, object oriented programming in Java, JavaScript, Java web applications, CSS, DHTML, J2EE applications, C# for web applications, PHP, LAMP technologies and Business Intelligence. PhD Author (Lecturer) may be reached at danut_so@yahoo.com.

References

[1] David Reilly, *“Getting Started with JDBC”*, David Reilly webmaster, pp. 182-187, 2008.

[2] David Gallardo, *“Java design patterns 101”*, ibm.com/developerWorks, 2009.

[3] Dănuţ-Octavian Simion, *“Using Java in Business Applications”*, WSEAS Conferences in the Universitatea Politehnica, Bucharest, Romania, pp. 218–223, April 20-22, 2010, Conference/Session: ECC_COMPUTING, ISBN: 978-960-474-178-6, ISSN: 1790-5117.

[4] Doug Lea, *“Concurrent programming in Java design principles and patterns”*, Addison Wesley, pp. 247-253, 2009.

[5] E. Karakoc, P. Senkul, *“Composing semantic Web services under constraints”*, Expert Systems with Applications, Volume 36, Issue 8, October 2009, pp. 11021-11029, ISSN: 0957-4174, Imprint: PERGAMON.

[6] Wang Tzone I., Hsieh Tung Cheng, Kun Hua Tsai, Ti Kai Chiu, Ming Che Lee, *“Partially constructed knowledge for semantic query”*, Expert Systems with Applications, Volum 36, Nr. 6, pp. 10168-10179, 2009.

[7] URI: http://javaworld.com/javaworld/

[8] URI: http://www.packtpub.com/service-oriented-java-business-integration/

[9] URI: http://www.manageability.org/

[10] URI: http://www.javarules.org/