5AHITS 2012/13

SOA

Paul Pfeiffer-Vogl, Harun Koyuncu

TABLE OF CONTENTS

1. ASSIGNMENTS OF TASKS

Das neu eröffnete Unternehmen iKnow Systems ist spezialisiert auf Knowledgemanagement und bietet seinen Klienten die Möglichkeiten Daten und Informationen jeglicher Art (ähnlich wikipedia) in eine Wissensbasis einzupflegen und anschließend in der zentralen Wissensbasis nach Informationen zu suchen.

Folgendes ist im Rahmen der Aufgabenstellung verlangt:

- Stellen Sie ein Datenmodell für die Wissensbasis (so einfach wie möglich) auf. [1 Punkte]
- Richten Sie 2 Services ein, implementieren und providen Sie diese:
    - – sucheEintrag
    - in: suchwort
    - out: List [2 Punkte]

    - – erstelleEintrag
    - in: thema, content
    - out: success? [3 Punkte]

- Implementieren Sie weiters einen Client, der Ihre Services verwendet.(Commandline/UI) [4 Punkte]

- Dokumentieren Sie im weiteren Verlauf den Datentransfer mit SOAP. [2 Punkte]

- Der Einsatz eines Enterprise Service Bus (openESB [glassfish]) [4 Punkte]

Für die Umsetzung relevante Verweise:

Apache Webservices Project:
http://ws.apache.org/

Apache Axis
http://ws.apache.org/axis2/

JAX-WS Web Services and Axis2

http://axis.apache.org/axis2/java/core/docs/jaxws-guide.html
http://www.ibm.com/developerworks/java/library/j-jws8/index.html

----
Gruppengröße: 2
Abgabedokument:
Es sind alle allgemeinen Vorgaben zu erfüllen, sowie folgende spezielle Punkte
- Datenmodell
- Dokumentation der Services
- Dokumentation der Service-Clients

2. WHAT IS SOA?

SOA establishes an architectural model that aims to enhance the efficiency, agility, and productivity of an enterprise by positioning services as the primary means through which solution logic is represented in support of the realization of the strategic goals associated with service-oriented computing.[1]

On a fundamental basis, the service-oriented computing platform revolves around the service-orientation design paradigm and its relationship with service-oriented architecture. In fact, the term "service-oriented architecture" and its associated acronym have been used so broadly by the media and within vendor marketing literature that it has almost become synonymous with service-oriented computing itself. It is therefore very important to make a clear distinction between what SOA actually is and how it relates to other service-oriented computing elements. [1]

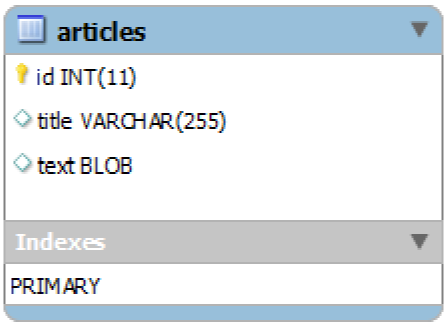
As a form of technology architecture, an SOA implementation can consist of a combination of technologies, products, APIs, supporting infrastructure extensions, and various other parts. The actual face of a deployed service-oriented architecture is unique within each enterprise; however it is typified by the introduction of new technologies and platforms that specifically support the creation, execution, and evolution of service-oriented solutions. As a result, building a technology architecture around the service-oriented architectural model establishes an environment suitable for solution logic that has been designed in compliance with service-orientation design principles. [1]

3. DIAGRAMS

After we searched a little bit in the internet we found a Netbeans version, which supports the Axis2 plugin and the creation of BPEL processes. So we decided to use this Netbeans distribution for the task.

3.1 ER DIAGRAM

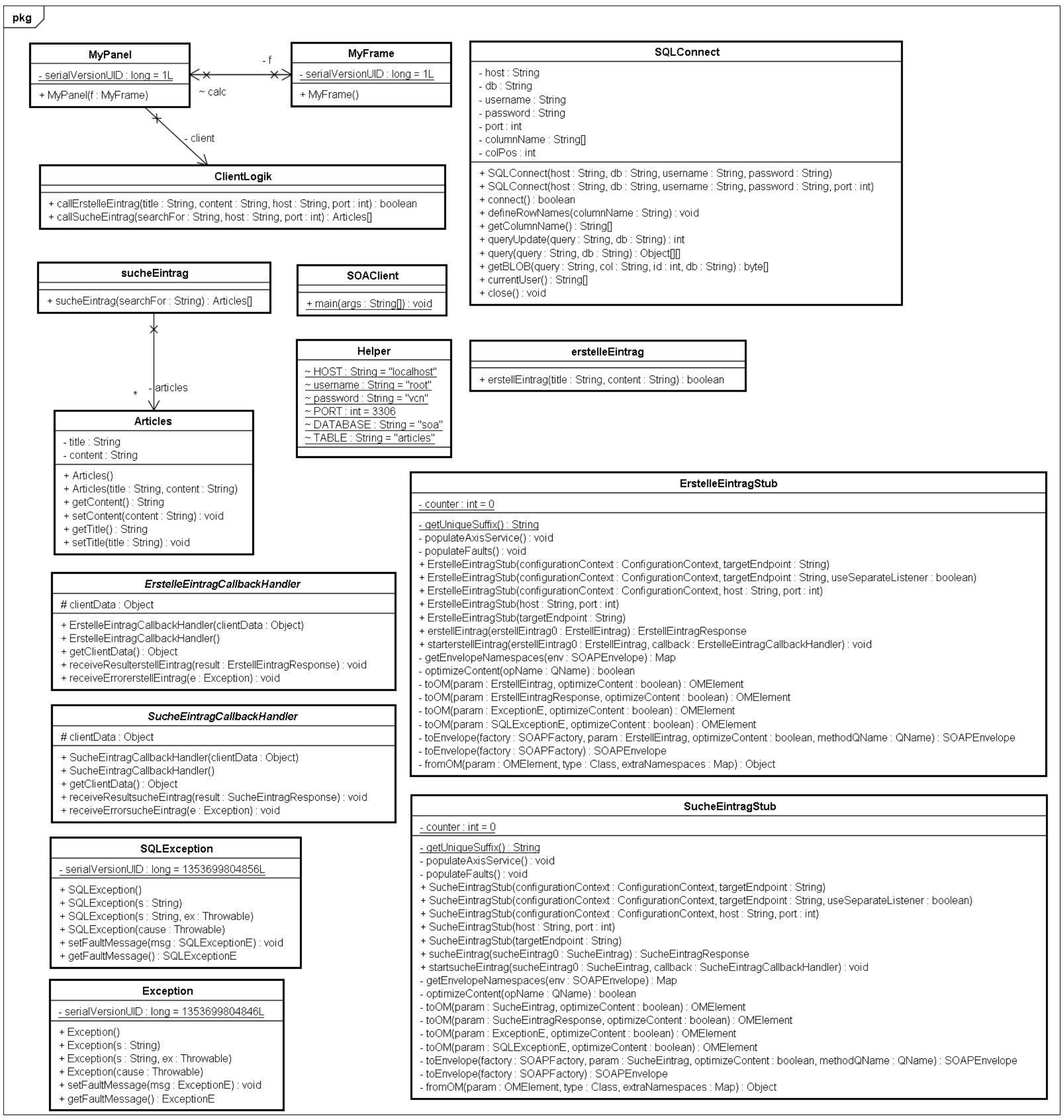
We decided to use a very simple knowledge base which is realized with mysql. So we design the following schema for this task:



[FIG 1] EER MODEL FOR THE KNOWLEDGE BASE

The filed id has auto generated values, which is handled by the DBMS. The filed “title” contains the title of an article and the content of the article is saved in the filed “text” which has a “BLOB” as datatyp.

3.2 CLASS DIAGRAM


pkg
MyPanel
- serialVersionUID : long = 1L
+ MyPanel(f : MyFrame)
- f
~ calc
MyFrame
- serialVersionUID : long = 1L
+ MyFrame()
- client
SQLConnect
- host : String
- db : String
- username : String
- password : String
- port : int
- columnName : String[]
- colPos : int
+ SQLConnect(host : String, db : String, username : String, password : String)
+ SQLConnect(host : String, db : String, username : String, password : String, port : int)
+ connect() : boolean
+ defineRowNames(columnName : String) : void
+ getColumnName() : String[]
+ queryUpdate(query : String, db : String) : int
+ query(query : String, db : String) : Object[][]
+ getBLOB(query : String, col : String, id : int, db : String) : byte[]
+ currentUser() : String[]
+ close() : void
ClientLogik
+ callErstelleEintrag(title : String, content : String, host : String, port : int) : boolean
+ callSucheEintrag(searchFor : String, host : String, port : int) : Articles[]
sucheEintrag
+ sucheEintrag(searchFor : String) : Articles[]
SOAClient
+ main(args : String[]) : void
Helper
~ HOST : String = "localhost"
~ username : String = "root"
~ password : String = "vcn"
~ PORT : int = 3306
~ DATABASE : String = "soa"
~ TABLE : String = "articles"
erstelleEintrag
+ erstellEintrag(title : String, content : String) : boolean
- articles
*
Articles
- title : String
- content : String
+ Articles()
+ Articles(title : String, content : String)
+ getContent() : String
+ setContent(content : String) : void
+ getTitle() : String
+ setTitle(title : String) : void
ErstelleEintragStub
- counter : int = 0
- getUniqueSuffix() : String
- populateAxisService() : void
- populateFaults() : void
+ ErstelleEintragStub(configurationContext : ConfigurationContext, targetEndpoint : String)
+ ErstelleEintragStub(configurationContext : ConfigurationContext, targetEndpoint : String, useSeparateListener : boolean)
+ ErstelleEintragStub(configurationContext : ConfigurationContext, host : String, port : int)
+ ErstelleEintragStub(host : String, port : int)
+ ErstelleEintragStub(targetEndpoint : String)
+ erstellEintrag(erstellEintrag0 : ErstellEintrag) : ErstellEintragResponse
+ starterstellEintrag(erstellEintrag0 : ErstellEintrag, callback : ErstelleEintragCallbackHandler) : void
- getEnvelopeNamespaces(env : SOAPEnvelope) : Map
- optimizeContent(opName : QName) : boolean
- toOM(param : ErstellEintrag, optimizeContent : boolean) : OMElement
- toOM(param : ErstellEintragResponse, optimizeContent : boolean) : OMElement
- toOM(param : ExceptionE, optimizeContent : boolean) : OMElement
- toOM(param : SQLExceptionE, optimizeContent : boolean) : OMElement
- toEnvelope(factory : SOAPFactory, param : ErstellEintrag, optimizeContent : boolean, methodQName : QName) : SOAPEnvelope
- toEnvelope(factory : SOAPFactory) : SOAPEnvelope
- fromOM(param : OMElement, type : Class, extraNamespaces : Map) : Object
ErstelleEintragCallbackHandler
# clientData : Object
+ ErstelleEintragCallbackHandler(clientData : Object)
+ ErstelleEintragCallbackHandler()
+ getClientData() : Object
+ receiveResulterstellEintrag(result : ErstellEintragResponse) : void
+ receiveErrorerstellEintrag(e : Exception) : void
SucheEintragCallbackHandler
# clientData : Object
+ SucheEintragCallbackHandler(clientData : Object)
+ SucheEintragCallbackHandler()
+ getClientData() : Object
+ receiveResultsucheEintrag(result : SucheEintragResponse) : void
+ receiveErrorsucheEintrag(e : Exception) : void
SucheEintragStub
- counter : int = 0
- getUniqueSuffix() : String
- populateAxisService() : void
- populateFaults() : void
+ SucheEintragStub(configurationContext : ConfigurationContext, targetEndpoint : String)
+ SucheEintragStub(configurationContext : ConfigurationContext, targetEndpoint : String, useSeparateListener : boolean)
+ SucheEintragStub(configurationContext : ConfigurationContext, host : String, port : int)
+ SucheEintragStub(host : String, port : int)
+ SucheEintragStub(targetEndpoint : String)
+ sucheEintrag(sucheEintrag0 : SucheEintrag) : SucheEintragResponse
+ startsucheEintrag(sucheEintrag0 : SucheEintrag, callback : SucheEintragCallbackHandler) : void
- getEnvelopeNamespaces(env : SOAPEnvelope) : Map
- optimizeContent(opName : QName) : boolean
- toOM(param : SucheEintrag, optimizeContent : boolean) : OMElement
- toOM(param : SucheEintragResponse, optimizeContent : boolean) : OMElement
- toOM(param : ExceptionE, optimizeContent : boolean) : OMElement
- toOM(param : SQLExceptionE, optimizeContent : boolean) : OMElement
- toEnvelope(factory : SOAPFactory, param : SucheEintrag, optimizeContent : boolean, methodQName : QName) : SOAPEnvelope
- toEnvelope(factory : SOAPFactory) : SOAPEnvelope
- fromOM(param : OMElement, type : Class, extraNamespaces : Map) : Object
SQLException
- serialVersionUID : long = 1353699804856L
+ SQLException()
+ SQLException(s : String)
+ SQLException(s : String, ex : Throwable)
+ SQLException(cause : Throwable)
+ setFaultMessage(msg : SQLExceptionE) : void
+ getFaultMessage() : SQLExceptionE
Exception
- serialVersionUID : long = 1353699804846L
+ Exception()
+ Exception(s : String)
+ Exception(s : String, ex : Throwable)
+ Exception(cause : Throwable)
+ setFaultMessage(msg : ExceptionE) : void
+ getFaultMessage() : ExceptionE

4. PACKAGES

<JAVA.SQL.SQLEXCEPTION>

This class handles an exception that provides information on a database access error or other errors. [4]

Documentation Link:
http://docs.oracle.com/javase/6/docs/api/index.html?java/sql/SQLException.html

< JAVA.RMI.REMOTEEXCEPTION >

A RemoteException is the common superclass for a number of communication-related exceptions that may occur during the execution of a remote method call. Each method of a remote interface, an interface that extends java.rmi.Remote, must list RemoteException in its throws clause. [4]

Documentation link:
http://docs.oracle.com/javase/6/docs/api/index.html?java/rmi/RemoteException.html

<JAVA.AWT.*>

This package contains all of the classes for creating user interfaces and for painting graphics and images. [4]

Documentation Link:
http://docs.oracle.com/javase/6/docs/api/index.html?java/awt/package-summary.html

<JAVAX.SWING.* >

This package provides a set of "lightweight" (all-Java language) components that, to the maximum degree possible, work the same on all platforms. [4]

Documentation Link:
http://docs.oracle.com/javase/6/docs/api/index.html?javax/swing/package-summary.html

5. REQUIREMENTS ANALYSIS

In the list below there are the requirements listed.

Requirements	**Description**
Eclipse 3.6 (Java 1.7)	Eclipse was used as IDE with integrated compiler
MySQL Database 5.5.xx	Needed for storing the entities
Astah	Generating the ER diagram
Netbeans & Glassfish	Deploying the project on Glassfish and programming
Axis2 Plugin	Apache Axis2™ is a Web Services / SOAP / WSDL engine

List about the financial costs:

Requirements	Costs
Eclipse 3.6 (Java 1.7)	free
MySQL Database 5.5.xx	free
Astah	free
Netbeans & Glasfish	free
Axis2 Plugin	free
Overall	**0 €**

6. WORKLOAD-SHARING

Tasks	Paul Pfeiffer-Vogl	Harun Koyuncu
Design	x	x
Understanding the given task	x	x
Understanding the given task	x	x
Programming the logic	x	x
Deploying the project	x	x
Comments	x	x
Translation German/English	x	x
Protocol	x	x

7. TIME SCHEDULE

Our first estimation for this task was about 12 hours.

We actually needed a work time about 18 hours because we got a lot of exceptions, that we couldn't resolve them in a short time and the integration of the axis2 plugin into Netbeans. In that case the work has become longer than it was planned.

Tasks	Paul Pfeiffer- Vogl	Harun Koyuncu	Estimated Time
Design	0 h 30 min	0 h 15 min	0 h 30 min
Understanding the given task	0 h 30 min	0 h 30 min	0 h 30 min
Understanding the architecture	1 h 00 min	1 h 00 min	1 h 30 min
Programming the logic	2 h 30 min	3 h 30 min	4 h 30 min
Deploying the project with axis2	1 h 30 min	0 h 30 min	1 h 00 min
Comments	1 h 30 min	0 h 30 min	1 h 30 min
Translation German/English	0 h 30 min	0 h 45 min	1 h 00 min
Protocol	1 h 00 min	1 h 30 min	2 h 00 min
Overall	**9 h 00 min**	**8 h 30 min**	**12 h 30 min**

Here is a summary from the time schedule:

Tasks	**Working Time**	**Estimated Time**
Design	0 h 45 min	0 h 30 min
Understanding the given task	1 h 00 min	0 h 30 min
Understanding the architecture	2 h 00 min	1 h 30 min
Programming the logic	6 h 00 min	4 h 30 min
Deploying the project with axis2	2 h 00 min	1 h 00 min
Comments	2 h 00 min	1 h 30 min
Translation German/English	1 h 15 min	1 h 00 min
Protocol	2 h 30 min	2 h 00 min
Overall	**17 h 30 min**	**12 h 30 min**

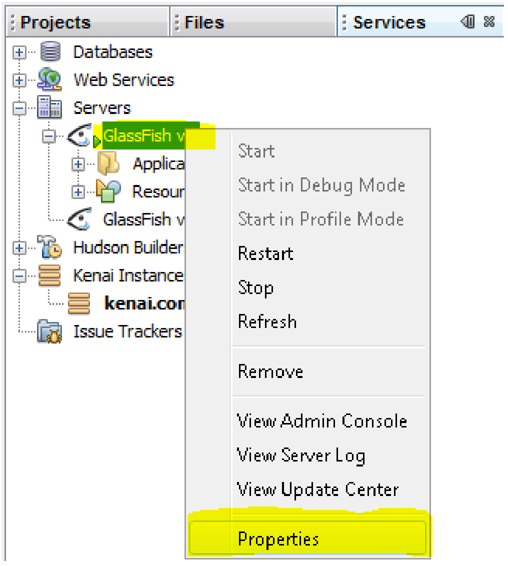
8. WHAT IS AXIS2

Apache Axis2™ is a Web Services / SOAP / WSDL engine, the successor to the widely used Apache Axis SOAP stack. There are two implementations of the Apache Axis2 Web services engine - Apache Axis2/Java and Apache Axis2/C. [3]

By using Axis2 it is possible to provide services. These services can be used by any client that knows the specific server address and port.

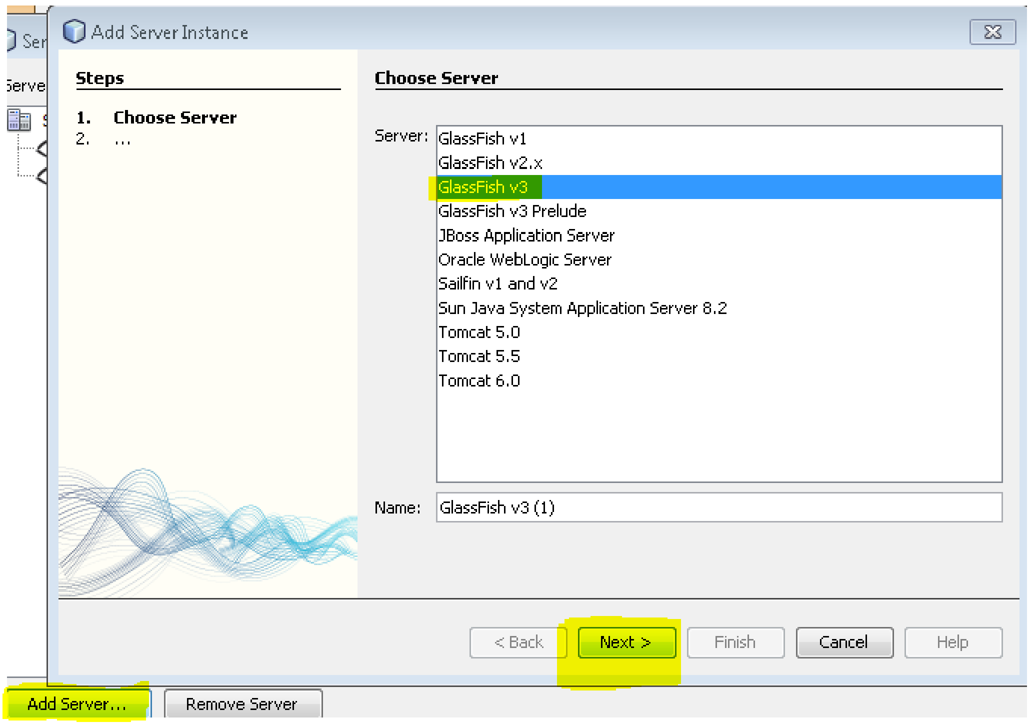
9. CONFIGURATION AXIS2

Before we can use the axis2 plugin successfully there must be done some tasks. First we create a new domain in Glassfish by right click on the servers like the following:



[FIG 2] OPENING THE PROPERTIES OF GLASSFISH

Then we click on the “Add server” button. After that a new window will appear. There we select the entry “GlassFish v3” and click the “Next” button.

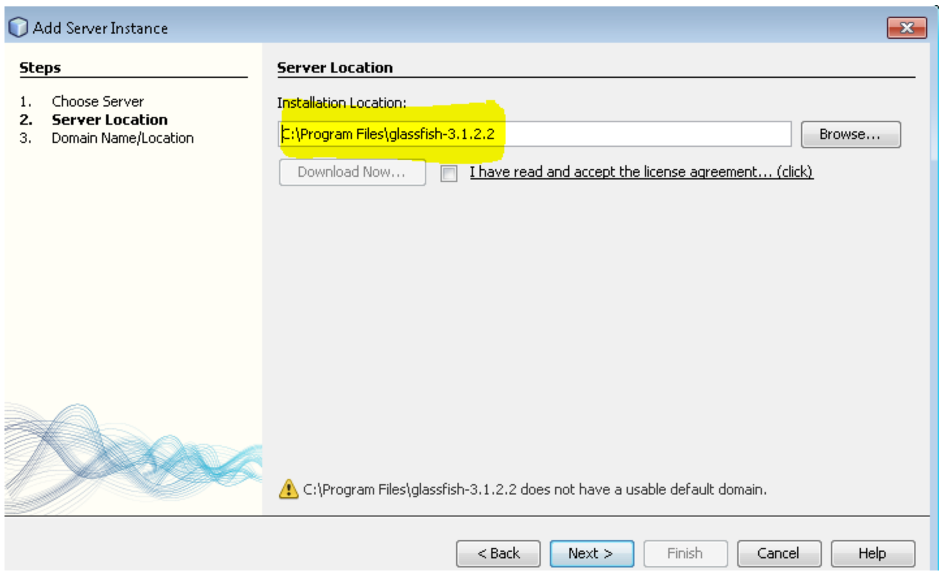


Server: GlassFish v1
GlassFish v2.x
GlassFish v3
GlassFish v3 Prelude
JBoss Application Server
Oracle WebLogic Server
Sailfin v1 and v2
Sun Java System Application Server 8.2
Tomcat 5.0
Tomcat 5.5
Tomcat 6.0

Name: GlassFish v3 (1)

[FIG 3] ADDING A NEW SERVER INSTANCE

On the next page we have to set the installation path of the glassfish like the following image and click then the “Next” button:



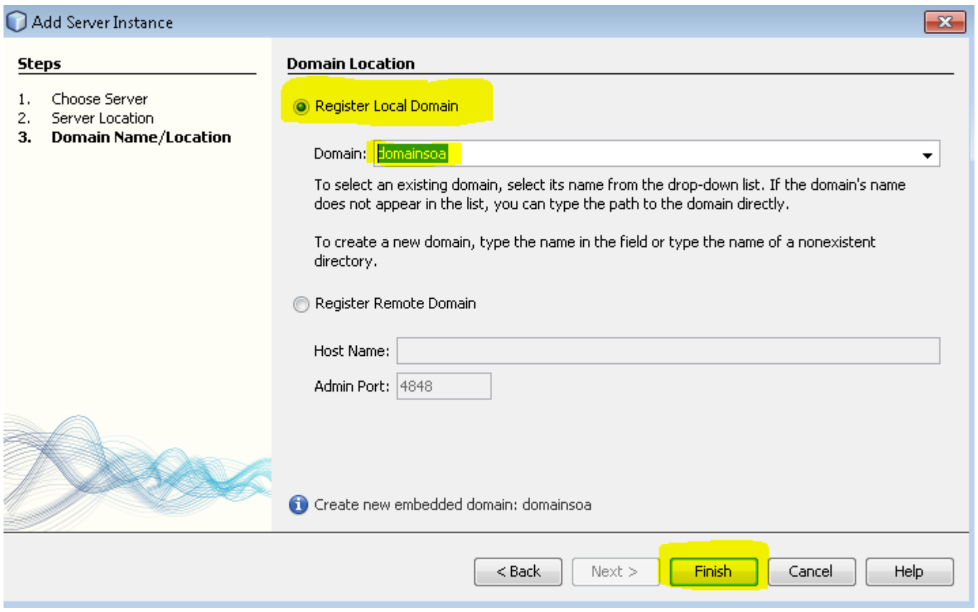
Installation Location: C:\Program Files\glassfish-3.1.2.2

I have read and accept the license agreement... (click)

C:\Program Files\glassfish-3.1.2.2 does not have a usable default domain.

[FIG 4] SET THE INSTALLATION PATH OF GLASSFISH

Now we give a new domain name and finally click the “Finish” button:



Add Server Instance

Steps

1. Choose Server
2. Server Location
3. **Domain Name/Location**

Domain Location

● Register Local Domain

Domain: domainsoa

To select an existing domain, select its name from the drop-down list. If the domain's name does not appear in the list, you can type the path to the domain directly.

To create a new domain, type the name in the field or type the name of a nonexistent directory.

○ Register Remote Domain

Host Name:

Admin Port: 4848

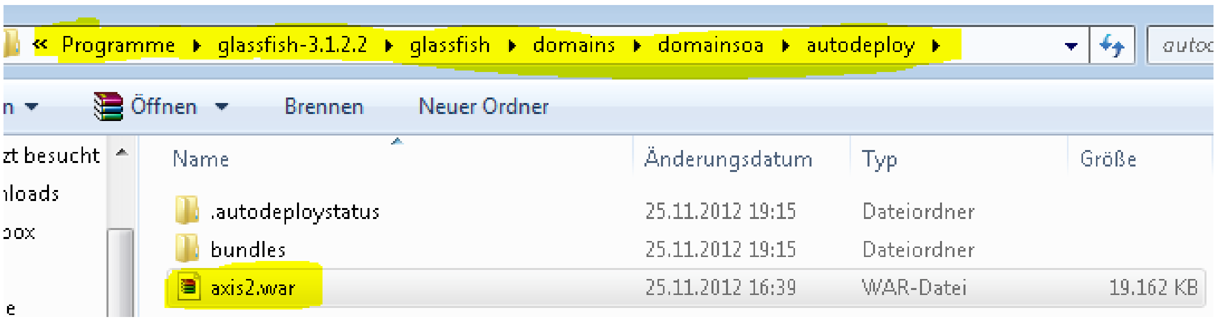
Create new embedded domain: domainsoa

[FIG 5] ENERTING A NEW DOMAIN FOR GLASSFISH

Now we download the “axis2.war” (War Distribution) file from the homepage of axis. To get directly to the download page use the following link:

- http://axis.apache.org/axis2/java/core/download.cgi

Extract the downloaded file and move the “axis2.war”, which is contained in the downloaded file, to the new generated domain folder. This folder is under the glassfish installation path. The „axis2.war“ must be placed in the new domain under the folder “autodeploy”:



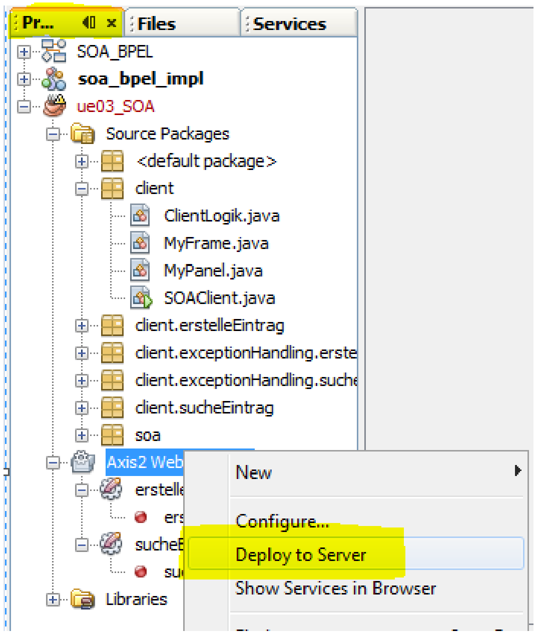
« Programme ▸ glassfish-3.1.2.2 ▸ glassfish ▸ domains ▸ domainsoa ▸ autodeploy ▸

Name	Änderungsdatum	Typ	Größe
.autodeploystatus	25.11.2012 19:15	Dateiordner	
bundles	25.11.2012 19:15	Dateiordner	
axis2.war	25.11.2012 16:39	WAR-Datei	19.162 KB

[FIG 6] MOVING THE AXIS2.WAR TO THE NEW DOMAIN

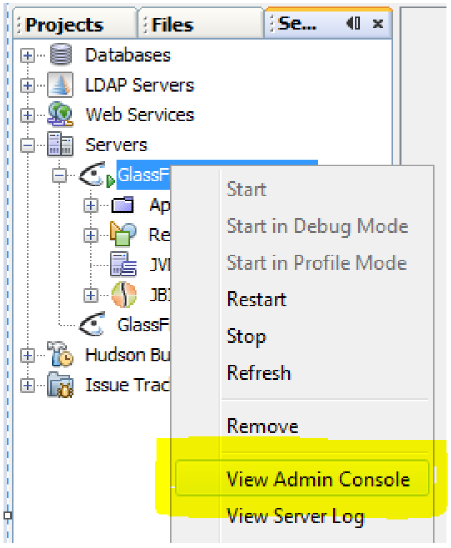
Important:
To get the correct port of the web service we have to deploy the web services in Netbeans first and then open the admin console site of our glassfish server. To open the admin console site, the glassfish server should be already running.

Now we deploy our services to the server as the following:



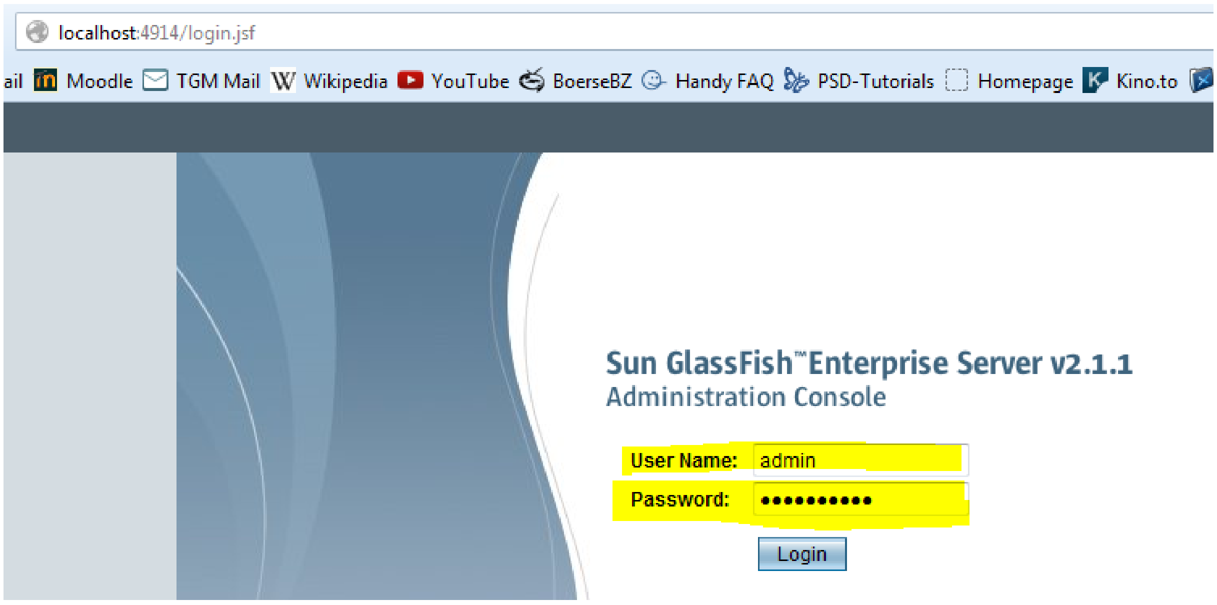
[FIG 7] DEPLOYING THE SERVICES TO THE SERVER

After that we deploy our services to the server we switch to the services tab and make a right click on our server and select "`View Admin Console`":



[FIG 8] OPENING THE ADMIN CONSOLE

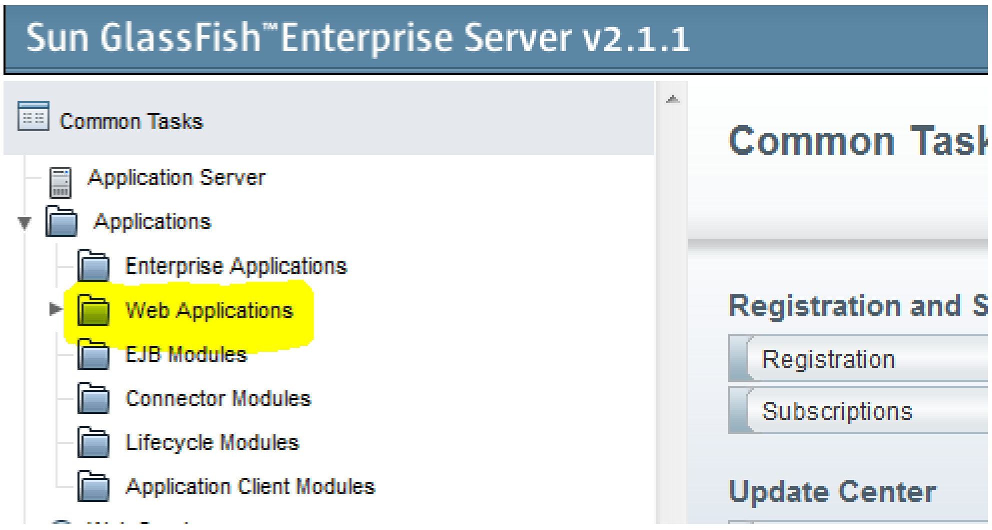
After a short time the admin console opens in a browser and the admin is called upon to enter his login information:


localhost:4914/login.jsf
ail Moodle TGM Mail Wikipedia YouTube BoerseBZ Handy FAQ PSD-Tutorials Homepage Kino.to
Sun GlassFish™ Enterprise Server v2.1.1
Administration Console
Login

User Name: admin
Password: ••••••••••

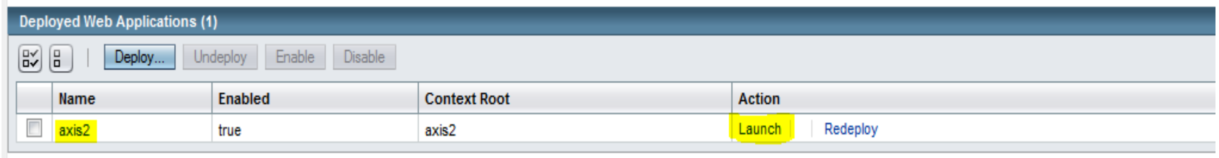
[FIG 9] ENTER THE LOGIN INFORMATION FOR THE ADMIN CONSOLE

After the login was successfully the following content appears and we click the "`Web Applications`" button:



[FIG 10] OPENING THE SERVICE WITH THE ADMINE CONSOLE

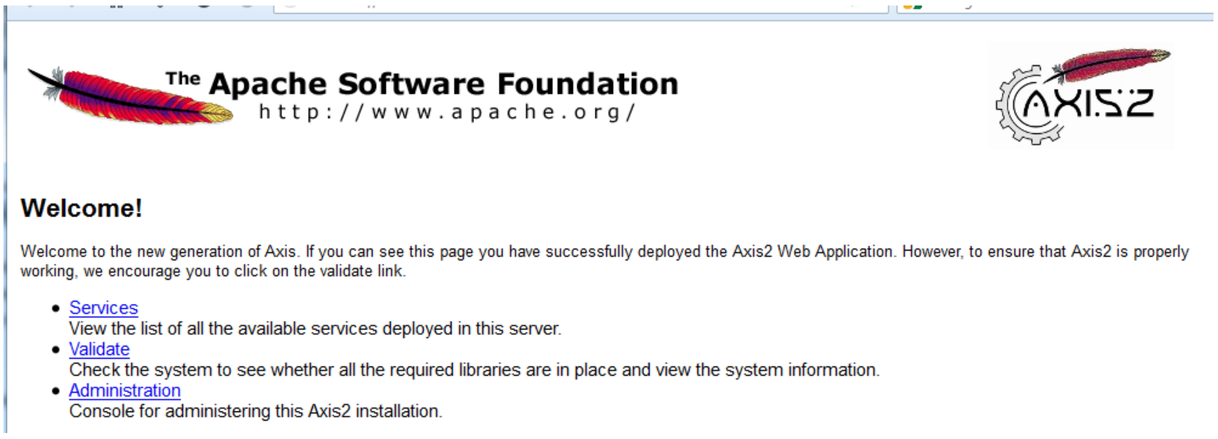
Our services should be listed as the following image. We click the "Launch" button and wait until the browser appears:


Deployed Web Applications (1)

	Name	Enabled	Context Root	Action
☐	axis2	true	axis2	Launch Redeploy

[FIG 11] LAUNCHING THE SERVICE OVER THE ADMIN CONSOLE

After the browser opens, we should see something like this:



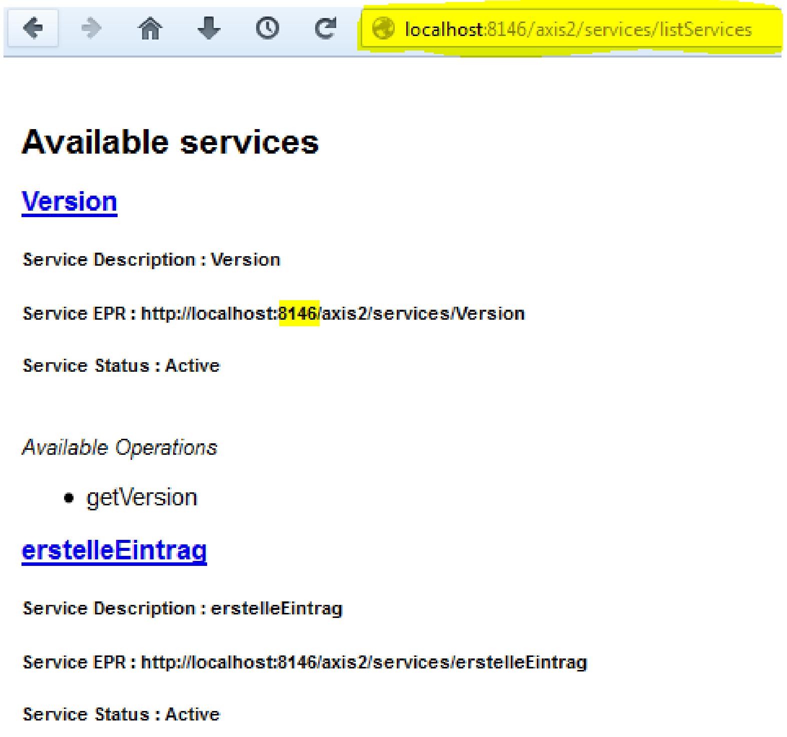
Welcome!

Welcome to the new generation of Axis. If you can see this page you have successfully deployed the Axis2 Web Application. However, to ensure that Axis2 is properly working, we encourage you to click on the validate link.

- Services
  View the list of all the available services deployed in this server.
- Validate
  Check the system to see whether all the required libraries are in place and view the system information.
- Administration
  Console for administering this Axis2 installation.

[FIG 12] OVERVIEW OF OUR SERVICE

Finally we click the "Services" button and then we get the correct port of our server:



Available services

Version

Service Description : Version

Service EPR : http://localhost:8146/axis2/services/Version

Service Status : Active

Available Operations

- getVersion

erstelleEintrag

Service Description : erstelleEintrag

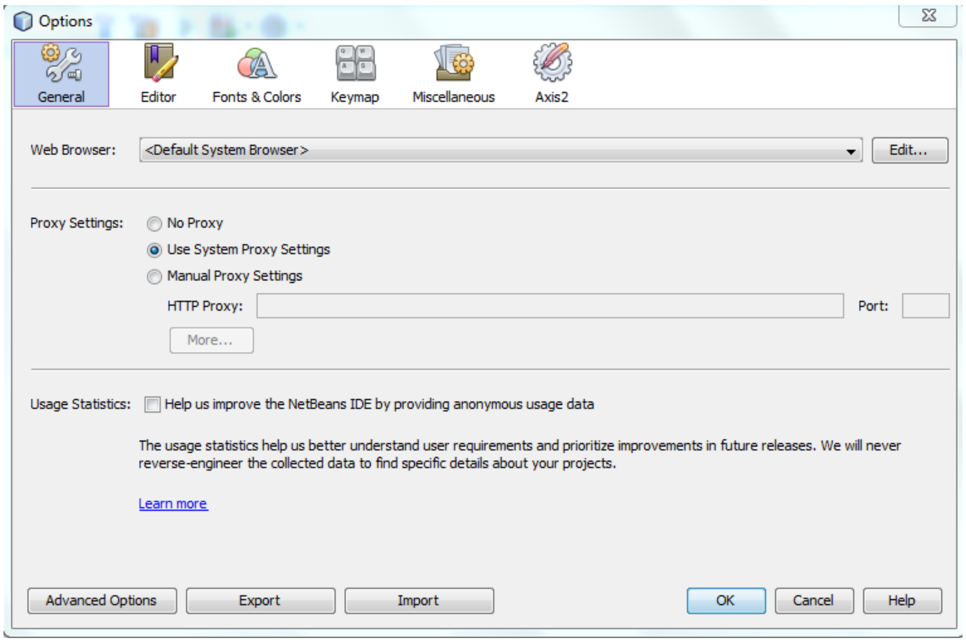
Service EPR : http://localhost:8146/axis2/services/erstelleEintrag

Service Status : Active

[FIG 13] DETERMINE THE PORT OF OUR SERVICES

In our case the port of our services is "**8146**".

Now we have to set the path of the "`axis2.war`" under Netbeans. First we select our SOA-project and then click on the "Tools". After that we click on "Options" and the following window should appear



Web Browser: <Default System Browser>

Proxy Settings: No Proxy [✓ Use System Proxy Settings] Manual Proxy Settings

HTTP Proxy: Port:

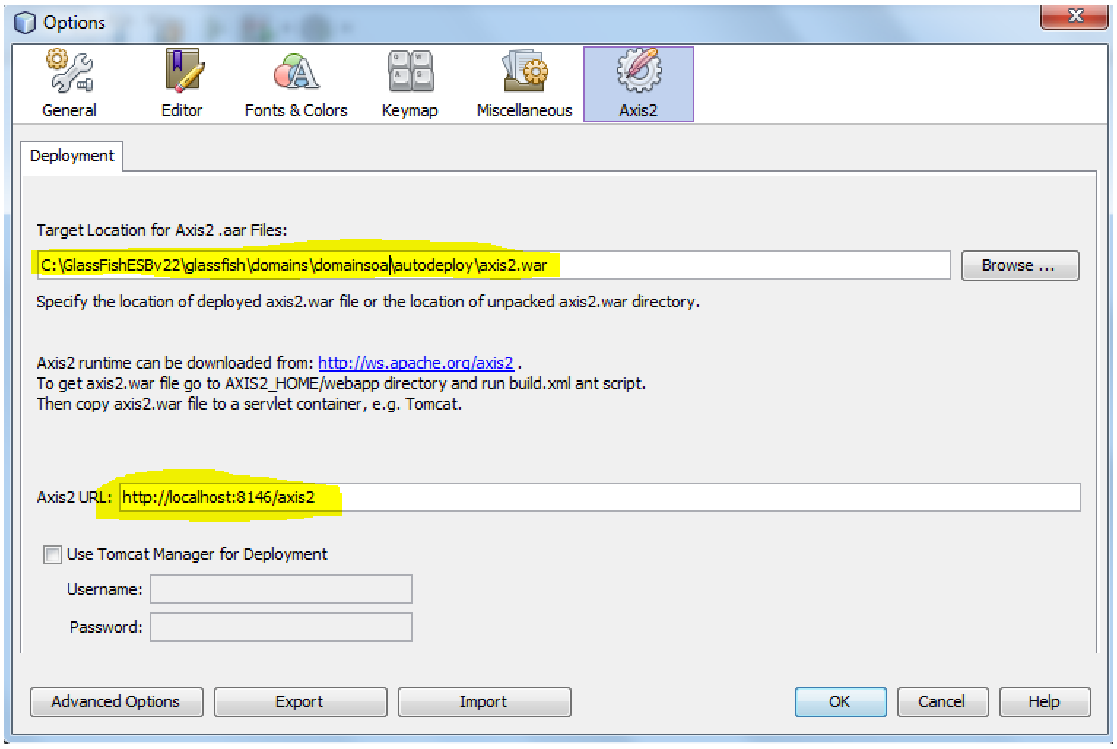
Usage Statistics: ☐ Help us improve the NetBeans IDE by providing anonymous usage data

The usage statistics help us better understand user requirements and prioritize improvements in future releases. We will never reverse-engineer the collected data to find specific details about your projects.

Learn more

[FIG 14] OPTION WINDOW

Now we click on the "Axis2"-Icon and fill the input fields with the following information:



Target Location for Axis2 .aar Files: C:\GlassFishESBv22\glassfish\domains\domainsoa\autodeploy\axis2.war

Specify the location of deployed axis2.war file or the location of unpacked axis2.war directory.

Axis2 runtime can be downloaded from: http://ws.apache.org/axis2 .
To get axis2.war file go to AXIS2_HOME/webapp directory and run build.xml ant script.
Then copy axis2.war file to a servlet container, e.g. Tomcat.

Axis2 URL: http://localhost:8146/axis2

☐ Use Tomcat Manager for Deployment

Username:

Password:

[FIG 15] SETTING THE AXIS2.WAR PROPERTIES IN NETBEANS

Now we download the MySQL connector from the following website and extract the downloaded file:

- http://www.mysql.de/downloads/connector/j/

Now we have to open the "axis2.war" in the domain folder with WinRar[5] and change into the "WEB-INF/lib" folder. Finally we drag and drop the downloaded MySql connector into this folder and save the "axis2.war" in WinRar.

Now the axis2 plugin is ready to use for our services.

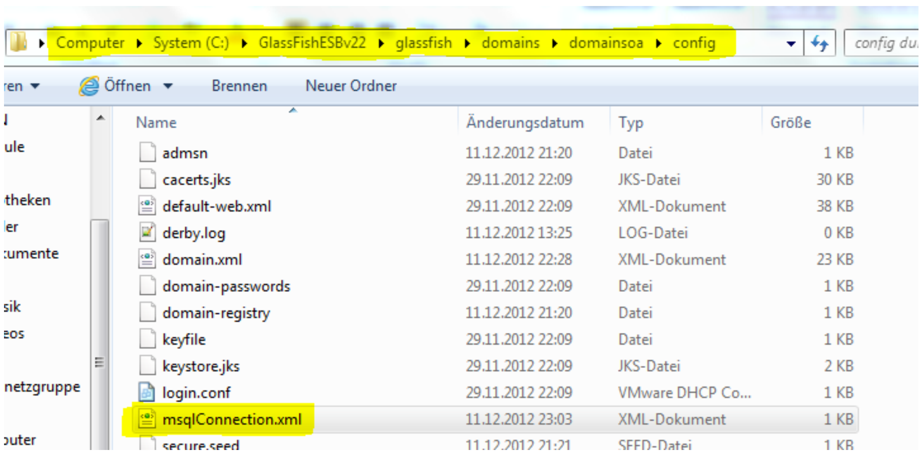
Now we should set up our knowledge data base, which is in our case just a mysql database. Our Application reads a XML-File, which contains all the needed information to connect successfully to a mysql database.

Important:
Please change the connection information to our own in order to use your own settimgs. Our application doesn't create the tables, or databases which is given in the configuration file. So make sure that you already created the database, table manually and the given login information's are correct. If just one of the entries are wrong our application tries to use the default settings, which are listed below:

Content of the configuration file called "`msqlConnection.xml`";

```
<?xml version="1.0"?>
<mysqldata>
      <host>localhost</host>
      <user>root</user>
      <password>pass</password>
      <database>soa</database>
      <table>articles</table>
      <port>3306</port>
</mysqldata>
```

In order to use the own mysql connection settings you have to move the configuration file "`msqlConnection.xml`" to the general "`config`" folder of the used domain(in our case was this "`domaimsoa`"):


Computer ▸ System (C:) ▸ GlassFishESBv22 ▸ glassfish ▸ domains ▸ domainsoa ▸ config
Öffnen
Brennen
Neuer Ordner

Name	Änderungsdatum	Typ	Größe
admsn	11.12.2012 21:20	Datei	1 KB
cacerts.jks	29.11.2012 22:09	JKS-Datei	30 KB
default-web.xml	29.11.2012 22:09	XML-Dokument	38 KB
derby.log	11.12.2012 13:25	LOG-Datei	0 KB
domain.xml	11.12.2012 22:28	XML-Dokument	23 KB
domain-passwords	29.11.2012 22:09	Datei	1 KB
domain-registry	11.12.2012 21:20	Datei	1 KB
keyfile	29.11.2012 22:09	Datei	1 KB
keystore.jks	29.11.2012 22:09	JKS-Datei	2 KB
login.conf	29.11.2012 22:09	VMware DHCP Co...	1 KB
msqlConnection.xml	11.12.2012 23:03	XML-Dokument	1 KB
secure.seed	11.12.2012 21:21	SEED-Datei	1 KB

[FIG 16] MOVING MYSQL-CONNECTION-FILE TO THE GENERAL CONFIG FOLDER

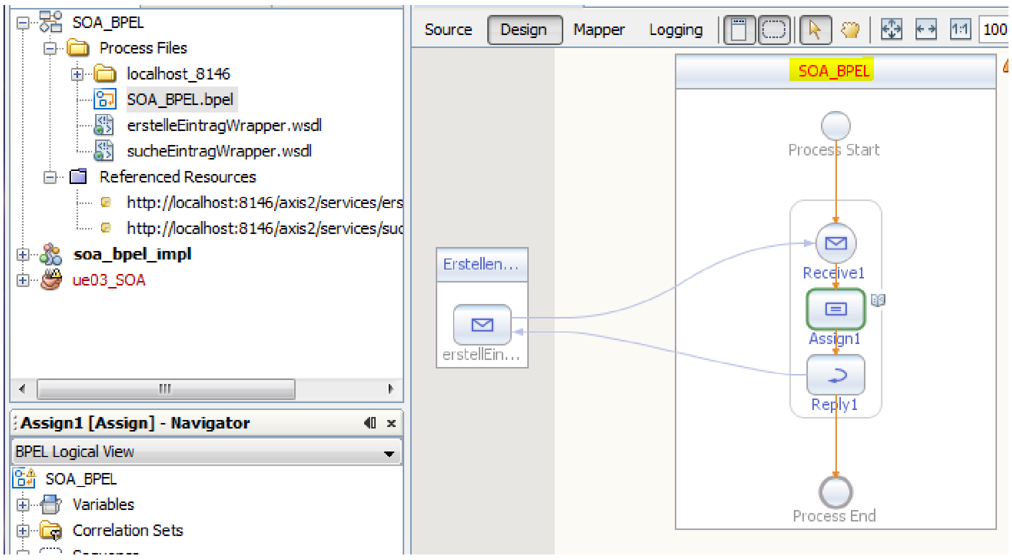
After this task is done, everything should be fine and you can use the services with the coming client.

10. OPENESB

For the BPEL implementation of our service we used the following online tutorial:

- http://wiki.open-esb.java.net/Wiki.jsp?page=OpenESBIntroductionTutorial (Part1) [6]

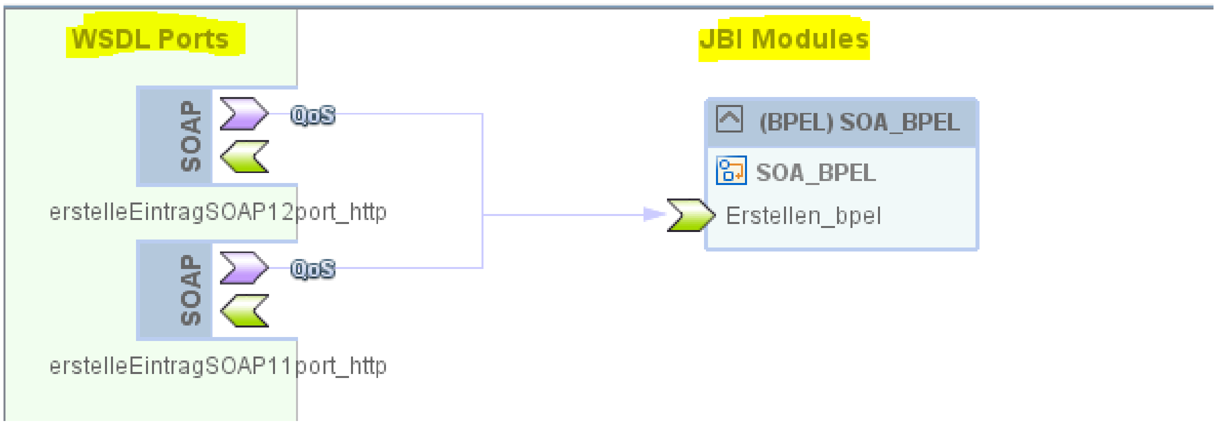
After the tutorial we get for our services the following BPEL design:



[FIG 17] BPEL DESIGN

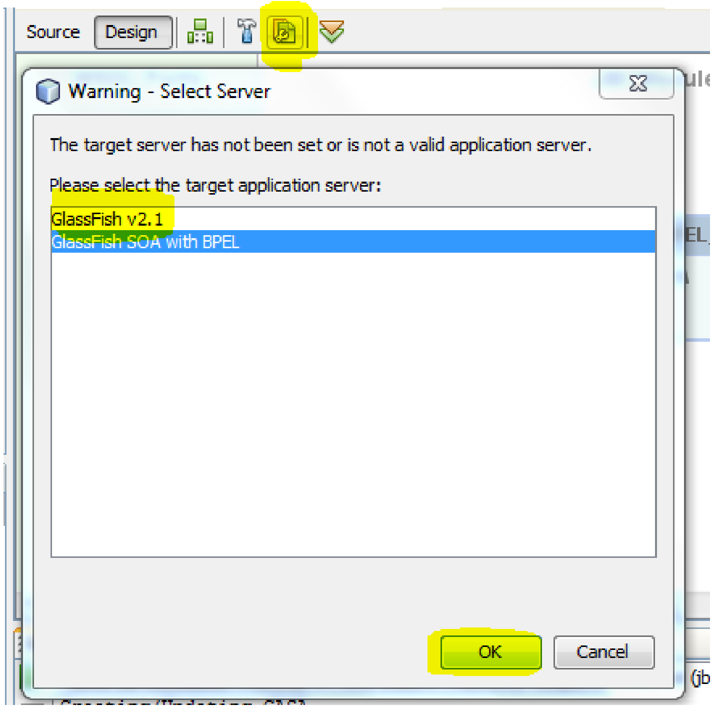
Before we create a “composite Application” we have to validate the current project. So we click to the validation button and wait for the output. If the Output contains some errors then the wsdl should be corrected. In our case we got exceptions about “SQLException”. First we searched in the internet for solving the problem but couldn’t find any solution. So we delete the “SQLExecption” parts from the wsdl file and then it works for us.

Now we create a new “composite Application”. Now we can drag and drop our BPEL process named “SOA_BPEL.bpel” into the GUI of the composite application and click the build button. After that we should get something like this:



[FIG 18] COMPOSITE APPLICATION

Now we have to deploy our composite application to the server. We click the deploy button and wait for the output:


Warning - Select Server

The target server has not been set or is not a valid application server.

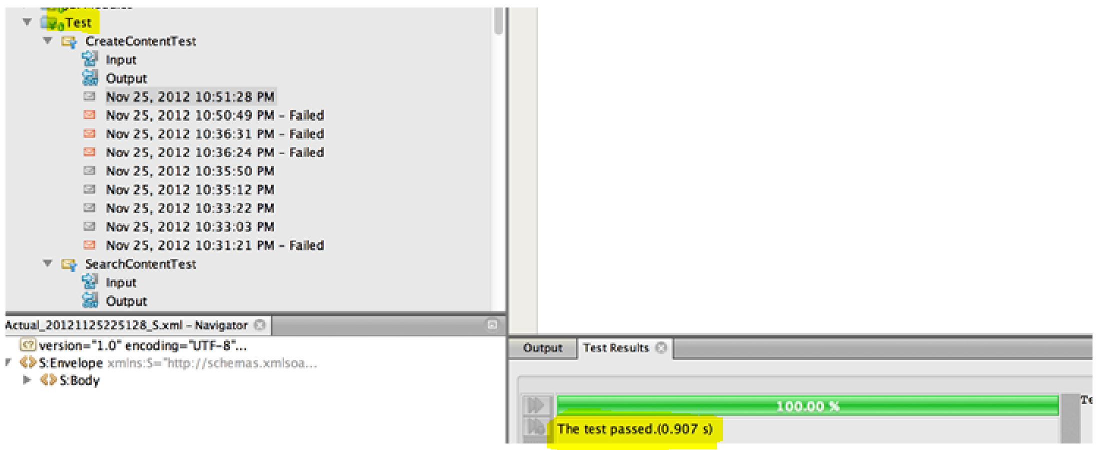
Please select the target application server:

GlassFish v2.1
GlassFish SOA with BPEL

[FIG 19] DEPLOYING THE COMPOSITE APPLICATION

It is suggested to use a glassfish server with the 2.1 version. After we deployed the application we can do some test cases like in the tutorial described.

When we get a successful test, it should look like something similar to this:


```
Test
  CreateContentTest
    Input
    Output
    Nov 25, 2012 10:51:28 PM
    Nov 25, 2012 10:50:49 PM - Failed
    Nov 25, 2012 10:36:31 PM - Failed
    Nov 25, 2012 10:36:24 PM - Failed
    Nov 25, 2012 10:35:50 PM
    Nov 25, 2012 10:35:12 PM
    Nov 25, 2012 10:33:22 PM
    Nov 25, 2012 10:33:03 PM
    Nov 25, 2012 10:31:21 PM - Failed
  SearchContentTest
    Input
    Output
```

Actual_20121125225128_S.xml - Navigator

```
version="1.0" encoding="UTF-8"...
S:Envelope xmlns:S="http://schemas.xmlsoa...
  S:Body
```

100.00 %

The test passed.(0.907 s)

[FIG 20] TESTCASE

11. CREATING THE CLIENT STUBS FOR THE SERVICES

We generated the client stubs by using the useful tools, which are coming with the axis2 binary distribution. For each service we generated a wsdl file and used the wsdl2java tool. We realized that is better to use the local path to the wsdl file during the client stub generation.

Usage:

- `wsdl2Java.bat -uri C:\Users\VcN\Documents\NetBeansProjects\ue03_SOA\xml-resources\axis2\META-INF\erstelleEintrag.wsdl`

- `wsdl2Java.bat -uri C:\Users\VcN\Documents\NetBeansProjects\ue03_SOA\xml-resources\axis2\META-INF\sucheEintrag.wsdl`

The tool generates the following classes:

- For the “`erstelleEintrag`” service:
  - `ErstelleEintragCallbackHandler`
  - `ErstelleEintragStub`
  - `Exception`
  - `SQLException`

- For the “`sucheEintrag`” service:
  - `SucheEintragCallbackHandler`
  - `SucheEintragStub`
  - `Exception`
  - `SQLException`

The exception handling is done in the two classes `Exception` and `SQLException`. The other two classes are maintaining the connection to the service and handling the response and the request.

12. CLIENT IMPLEMENTATION

The client implementation is done with the following code:

<u>E.g. for the service “`erstelleEintrag`”:</u>

```
ErstelleEintragStub stub = new ErstelleEintragStub(host, port);
ErstelleEintragStub.ErstellEintrag request = new
                           ErstelleEintragStub.ErstellEintrag();
request.setTitle(title);
request.setContent(content);
ErstelleEintragStub.ErstellEintragResponse response =
                                        stub.erstellEintrag(request);
boolean status = response.get_return();
```

The implementation for another service is the same like the example above but the only thing to be careful is to save the response of the service in the correct data type.

13. DATA TRANSFER WITH SOAP

SOAP is a lightweight protocol for exchange of information in a decentralized, distributed environment. It is an XML based protocol that consists of three parts: an envelope that defines a framework for describing what is in a message and how to process it, a set of encoding rules for expressing instances of application-defined datatypes, and a convention for representing remote procedure calls and responses. SOAP can potentially be used in combination with a variety of other protocols; however, the only bindings defined in this document describe how to use SOAP in combination with HTTP and HTTP Extension Framework.[2]

EXAMPLES OF SOAP MESSAGES

In this example, a “GetLastTradePrice” SOAP request is sent to a StockQuote service. The request takes a string parameter, ticker symbol, and returns a float in the SOAP response. The SOAP Envelope element is the top element of the XML document representing the SOAP message. XML namespaces are used to disambiguate SOAP identifiers from application specific identifiers. [2]

SOAP MESSAGE EMBEDDED IN HTTP REQUEST

```
POST /StockQuote HTTP/1.1
Host: www.stockquoteserver.com
Content-Type: text/xml; charset="utf-8"
Content-Length: nnnn
SOAPAction: "Some-URI"

<SOAP-ENV:Envelope
  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
  SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
   <SOAP-ENV:Body>
       <m:GetLastTradePrice xmlns:m="Some-URI">
           <symbol>DIS</symbol>
       </m:GetLastTradePrice>
   </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

[FIG 21] SOAP REQUEST

SOAP MESSAGE EMBEDDED IN HTTP RESPONSE

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset="utf-8"
Content-Length: nnnn

<SOAP-ENV:Envelope
  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
  SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"/>
   <SOAP-ENV:Body>
       <m:GetLastTradePriceResponse xmlns:m="Some-URI">
           <Price>34.5</Price>
       </m:GetLastTradePriceResponse>
   </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

[FIG 22] SOAP RESPONSE

14. ACHIEVEMENTS/DEFEATS

14.1 ACHIEVEMENTS

Through this task we earned a lot of information about SOA. So we learned how to create, deploy and use a service with SOA. We also learned how to create a client and handle complex data structures as return type.

14.2 DEFEATS

GENERATING THE CLIENT STUB

At the first time we didn't know how to generate the client stub. So we decided to access the service using http sessions for requests and responses but then we found the axis2 standard distribution which contains useful tools. We generated the client stubs by using the wsdl2.java tool. After that we had all the needed classes for implement the client, who is accessing and using the services.

WRONG WSDL FILE USED

We generate the client stub using the address to the wsdl file as following:

- `http://localhost:47270/axis2/services/sucheEintrag?wsdl`

The problem was that the axsis tool wsdl2java used a new generated wsdl file, which contained a wrong description of the service such as the return type for the service "sucheEintrag". We passed this problem by giving the local path to the wsdl file for creating successfully the client stub.

AXIS2 PLUGIN FOR NETBEANS

We wanted to use the Netbeans plugin called axis2, but we realise that the plugin only was supported for an older version of the Nerbeans IDE. So we had to install an older version of Netbeans(6.8) to use the plugin successfully. For the BPEL implementation we used another Netbans distributed called "Netbeans OpenESB".

COMPLEX RETURN TYPE IN AXIS2

We first tried to return a multi-dimensional String array. But that didn't seem to work.

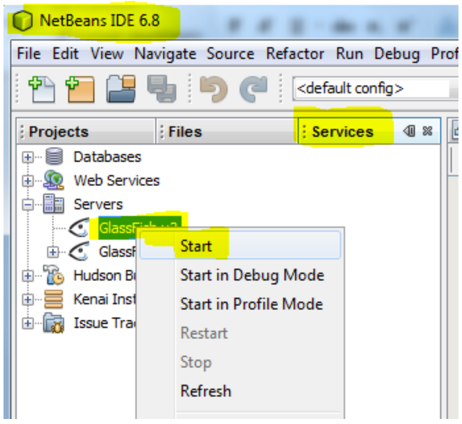
After we generated the client stub using the axis2 tool called "wsdl2java" we realise that axis2 is not able to handle such complex types. After some time we found a solution in the internet. We had to create a class which handles the content of the complex type. In our case this was the title of the article and its content. To access and set the attributes we write setter and getter methods into that class. Then we used as return type an array of this class.

15. APPLICATION DEMONSTARTION

We will show the usage of this project with the Netbeans IDE, which contains a server where we can deploy our services.

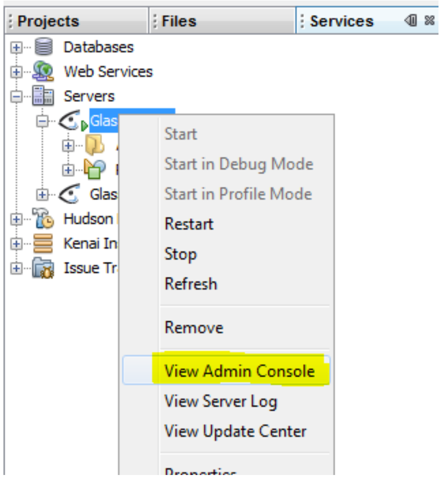
STARTING THE GLASSFISH SERVER

Open the Netbeans IDE and switch to the service tab. Then right click on the Glassfish server and press the button start:



[FIG 23] STARTING THE GLASSFISH SERVER

After a while the server starts up and you can access the admin page to be sure that the server has started successfully. To check this out just do a right click on the server and select the “View Admin Console”:

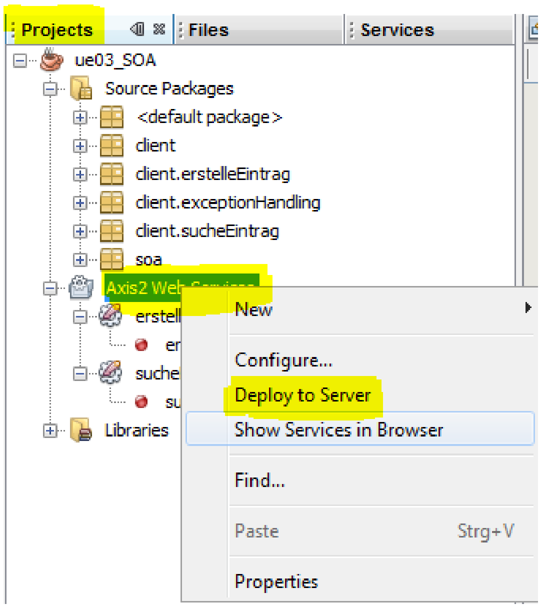


[FIG 24] OPENING THE ADMIN CONSOLE

After that a browser opens and you can see the admin page of the Glassfish server.

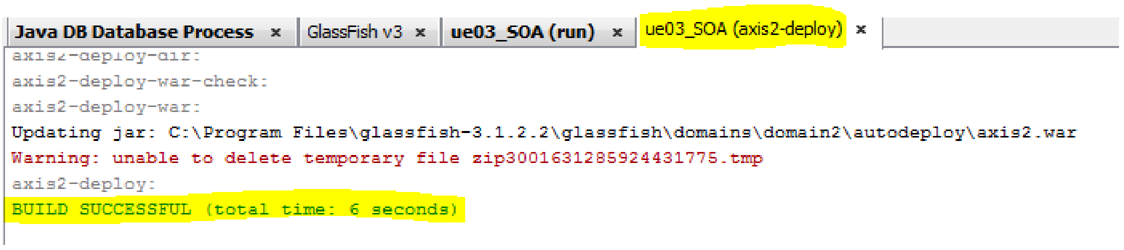
DEPLOYING THE SERVICES TO THE SERVER & STARTING MYSQL

Now we have to deploy our services to the server. We do this by right click on the axis2 web services and select the “Deploy to server” entry.



[FIG 25] DEPLOYING THE SERVICES

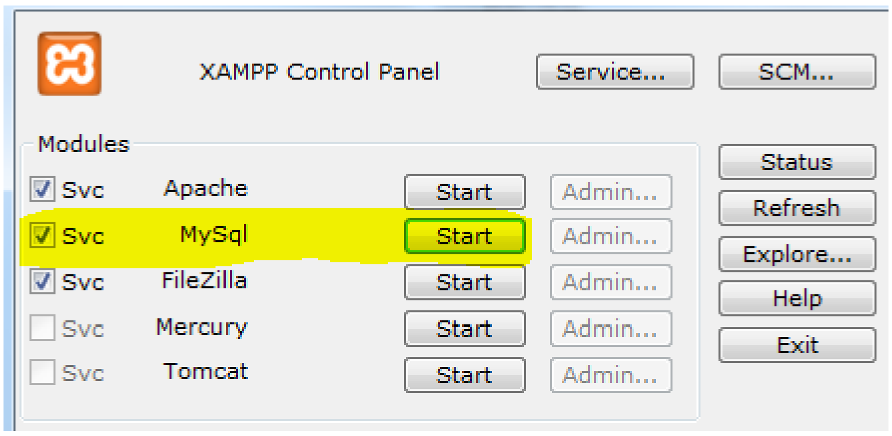
After a few minutes we get a success message like this:



```
axis2-deploy-dir:
axis2-deploy-war-check:
axis2-deploy-war:
Updating jar: C:\Program Files\glassfish-3.1.2.2\glassfish\domains\domain2\autodeploy\axis2.war
Warning: unable to delete temporary file zip300163128592443l775.tmp
axis2-deploy:
BUILD SUCCESSFUL (total time: 6 seconds)
```

[FIG 26] SUCCESS MESSAGE FOR THE DEPLOYING

Now we have to start the mysql service, where our services will be written. Open XAMPP and hit the start button for mysql.


XAMPP Control Panel
Service...
SCM...
Modules
Svc Apache Start Admin...
Svc MySql Start Admin...
Svc FileZilla Start Admin...
Svc Mercury Start Admin...
Svc Tomcat Start Admin...
Status
Refresh
Explore...
Help
Exit

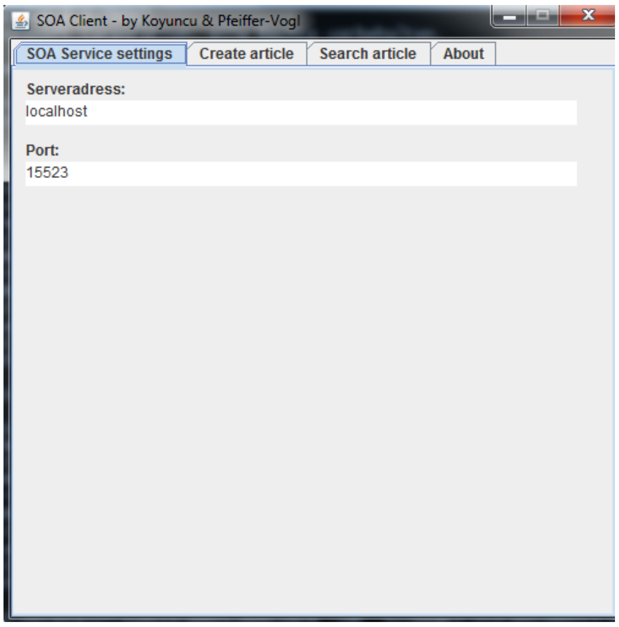
[FIG 27] STARTING MYSQL SERVICE

CLIENT USAGE

The clients can be started with the following command through the command line:

- java –jar client.jar

Then a GUI appears and looks like this:


SOA Client - by Koyuncu & Pfeiffer-Vogl

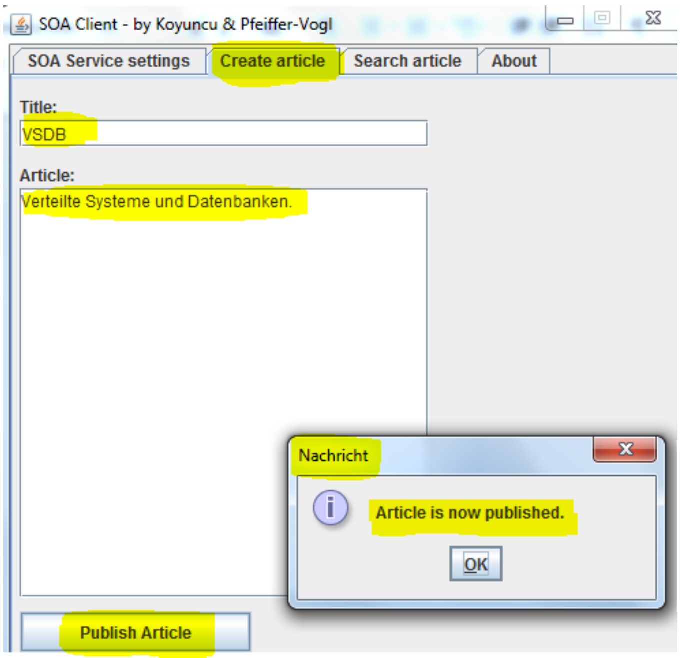
Serveradress: localhost

Port: 15523

[FIG 28] GUI OF OUR SOA CLIENT

One the first tab called "SOA Service settings" should be defined the location of the server, where the services are running.

On the second tab the user can write a new article and publish it. When he presses the "publish article" button it should appear a success message when everything went well.


SOA Client - by Koyuncu & Pfeiffer-Vogl

Title: VSDB

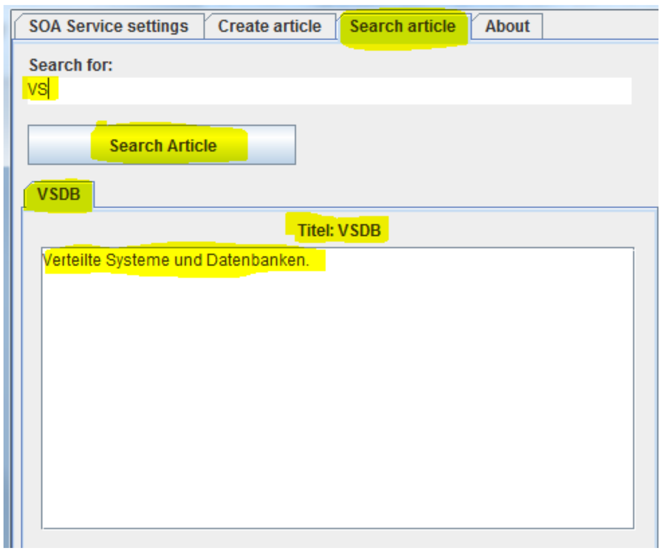
Article: Verteilte Systeme und Datenbanken.

Nachricht

Article is now published.

[FIG 29] PUBLISHING AN ARTICLE

In the third tab the user have the option for searching any articles, which contains a keyword. If any result exists then they will be shown to the client:



Verteilte Systeme und Datenbanken.

[FIG 30] SEARCHING ARTICLES

16. TABLE OF FIGURES

17. SOURCES

[1] ServiceOrientation.com, [Online], URL: http://serviceorientation.com/index.php/whatissoa/p10, [accessed on 17-Oct-2012]

[2] Simple Object Access Protocol , [Online], URL: http://www.w3.org/TR/2000/NOTE-SOAP-20000508/, [accessed on 25-Nov-2012]

[3] Welcome to Apache Axis2/Java, [Online], URL: http://axis.apache.org/axis2/java/core/, [accessed on 25-Nov-2012]

[4] Java™ Platform, Standard Edition 6 API Specification, [Online], URL: http://docs.oracle.com/javase/6/docs/api/index.html, [accessed on 25-Nov-2012]

[5] WinRar, [Online], URL: http://www.rarlab.com/ [accessed on 20-Nov-2012]

[6] OpenESB, [Onlien], URL: http://wiki.open-esb.java.net/Wiki.jsp?page=OpenESBIntroductionTutorial, [accessed on 20-Nov-2012]