Adding a Database to a Service in Genesis II
Tutorial

Karolina Sarnowska 06/05/2007

This tutorial will outline the implementation framework for adding a database resource to a service in Genesis II. The connection between xml, wsdl, and java files will be discussed.

I. Step One

The XML documents

Three XML documents should be created in the >wsdl folder:

a. .xsd - here the elements that will be used are specified. In the *addition* example there is a complex element encapsulating the two integer values to be added and a complex type element for the result.
b. .gwsdl – the .gwsdl file contains information about the messages that will be exchanged and the port through which they will pass.
c. .build xml – this is the build file for the service. Basically all you need to do to change this file is to replace the name of the old service with the name of your own. Just make sure that the paths are correct.

The easiest way is to copy-and-paste from an already existing service and adjusts the code. Some services to look at are *queue* or *rns*.

After the three files are created:

1. The mappings.properties file needs to be changed. A mapping has to be entered for the new service
2. The build.xml file needs to be changed.
   a. an entry should be made in the *imports;*
   b. a *target* name should be added
3. Run the build.xml file. This generates code in the genned-src folder. There should be a folder with the name of your service in edu.virginia.vcgr.genii

If there are any errors usually they'll show during the build. When everything works properly the service can be seen at: https://localhost:18080/axis/services

Bellow are the three files as they should look for a web-service performing a simple addition of two elements:

addition.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema
        xmlns="http://vcgr.cs.virginia.edu/genii/addition"
        xmlns:xsd="http://www.w3.org/2001/XMLSchema"
        xmlns:wsa="http://www.w3.org/2005/08/addressing"
        xmlns:wsbf="http://docs.oasis-open.org/wsrf/bf-2"
        targetNamespace="http://vcgr.cs.virginia.edu/genii/addition"
        elementFormDefault="qualified"
        attributeFormDefault="unqualified">

        <xsd:import namespace="http://docs.oasis-open.org/wsrf/bf-2"
                schemaLocation="./basefaults.xsd"/>

        <xsd:import
                namespace="http://www.w3.org/2005/08/addressing"
                schemaLocation="./ws-addr.xsd"/>

//A complex type element is created. It contains the two elements to be added. Specified is that both
elements (firstAdditionElement and secondAdditionElement) should occur exactly once.

        <xsd:complexType name="AdditionType">
                <xsd:sequence>
                        <xsd:element
                                name="firstAdditionElement" type="xsd:integer"
                                minOccurs="1" maxOccurs="1"/>
                        <xsd:element
                                name="secondAdditionElement" type="xsd:integer"
                                minOccurs="1" maxOccurs="1"/>
                </xsd:sequence>
        </xsd:complexType>
        <xsd:element name="AdditionType"
                type="AdditionType"/>

//This is the complex type element for the result of the addition. The result should also occur once.
        <xsd:complexType name="ResultType">
                <xsd:sequence>
                        <xsd:element
                                name="result" type="xsd:integer"
                                minOccurs="1" maxOccurs="1"/>
                </xsd:sequence>
        </xsd:complexType>
        <xsd:element name="ResultType"
                type="ResultType"/>

</xsd:schema>
```

addition.gwsdl

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

<wsdl:definitions
        name="Addition"
        xmlns="http://vcgr.cs.virginia.edu/genii/addition"
        targetNamespace="http://vcgr.cs.virginia.edu/genii/addition"
        xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
        xmlns:addition="http://vcgr.cs.virginia.edu/genii/addition"
        xmlns:wsa="http://www.w3.org/2005/08/addressing"
        xmlns:genii-common="http://vcgr.cs.virginia.edu/genii/2006/12/common"
        xmlns:genii-resource="http://vcgr.cs.virginia.edu/genii/2006/12/resource"
   xmlns:genii-ext="http://vcgr.cs.virginia.edu/genii/2006/12/wsdl-extensions"
        xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">

        <wsdl:import
                namespace="http://vcgr.cs.virginia.edu/genii/2006/12/resource"
                location="./resource.wsdl"/>

        <wsdl:import
                namespace="http://vcgr.cs.virginia.edu/genii/2006/12/resource-factory"
                location="./resource-factory.wsdl"/>

        <wsdl:import
                namespace="http://vcgr.cs.virginia.edu/genii/2006/12/resource-lifetime"
                location="./resource-lifetime.wsdl"/>

        <wsdl:import
                namespace="http://vcgr.cs.virginia.edu/genii/2006/12/resource-attrs"
                location="./resource-attributes.wsdl"/>

        <wsdl:import
                namespace="http://vcgr.cs.virginia.edu/genii/2006/12/resource-authz"
                location="./resource-authz.wsdl"/>

   <wsdl:import
      namespace="http://vcgr.cs.virginia.edu/genii/2006/12/common"
      location="./common-generated.wsdl"/>

   <wsdl:import
      namespace="http://vcgr.cs.virginia.edu/genii/2006/12/notification"
      location="./notification.wsdl"/>

        <wsdl:types>
                <xsd:schema
                   xmlns:xsd="http://www.w3.org/2001/XMLSchema"
                   attributeFormDefault="unqualified"
                   elementFormDefault="qualified"

//The targetNamespace is specified
                   targetNamespace="http://vcgr.cs.virginia.edu/genii/addition">
                   <xsd:import
                           namespace="http://vcgr.cs.virginia.edu/genii/addition"
                                   schemaLocation="./addition.xsd"/>

                </xsd:schema>
        </wsdl:types>
```

```
//Here you specify the types of messages to be exchanged. In the current example there are two messages – AdditionRequest and AdditionResponse.

    <wsdl:message name="AdditionRequest">
                <wsdl:part name="AdditionRequest"
                        element="addition:AdditionType"/>
        </wsdl:message>

        <wsdl:message name="AdditionResponse">
                <wsdl:part name="AdditionResponse"
                        element="addition:ResultType"/>
        </wsdl:message>

// This is the portType to be used. The operations to be performed are specified in the portType definition. Two messages can pass through that port – a request for addition and a response.

        <wsdl:portType name="AdditionPortType">
                <genii-ext:extend portType="genii-common:GeniiCommon"/>
                <wsdl:operation name="addition">
                        <wsdl:input message="addition:AdditionRequest"/>
                        <wsdl:output message="addition:AdditionResponse"/>
                </wsdl:operation>

        </wsdl:portType>

</wsdl:definitions>
```

addition.build.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<project basedir="." default="gen.addition" name="addition.build">

	<condition property="gen.addition.uptodate">
		<and>
			<common.source.files.uptodate
				targetfile="${codegen-dir}/wsdl/addition.flag"/>
			<uptodate targetfile="${codegen-dir}/wsdl/addition.flag">
				<srcfiles dir="${wsdl-dir}">
					<include name="addition.xsd"/>
					<include name="addition.gwsdl"/>
				</srcfiles>
			</uptodate>
		</and>
	</condition>

	<target name="gen.addition" unless="gen.addition.uptodate">
		<copy todir="${codegen-dir}/wsdl">
			<fileset refid="common.source.files"/>
			<fileset dir="${wsdl-dir}">
				<include name="addition.gwsdl"/>
				<include name="addition.xsd"/>
			</fileset>
		</copy>

		<echo message="Normalizing gwsdl."/>
		<genii-gwsdl-normalizer source="${codegen-dir}/wsdl/addition.gwsdl"
			target="${codegen-dir}/wsdl/addition-generated.wsdl"/>

		<echo message="Generating Service gwsdl."/>
		<genii-service-generator
			source="${codegen-dir}/wsdl/addition-generated.wsdl"
			target="${codegen-dir}/wsdl/addition-service.wsdl"
			portType="{http://vcgr.cs.virginia.edu/genii/addition}AdditionPortType"/>

		<antcall target="gen.stub-generation">
			<param name="stub.gen.target-dir" value="${codegen-dir}/addition"/>
			<param name="stub.gen.source-wsdl"
				value="${codegen-dir}/wsdl/addition-service.wsdl"/>
			<param name="stub.gen.target-flag-file"
				value="${codegen-dir}/wsdl/addition.flag"/>
		</antcall>
		<move file="${codegen-dir}/addition/edu/virginia/vcgr/genii/addition/deploy.wsdd"
			tofile="${wsdd-dir}/addition.deploy.wsdd"/>
		<replace file="${wsdd-dir}/addition.deploy.wsdd"
			token="edu.virginia.vcgr.genii.addition.AdditionSOAPBindingImpl"
			value="edu.virginia.vcgr.genii.container.addition.AdditionServiceImpl"
			summary="yes"/>
		<copy file="${wsdd-dir}/addition.deploy.wsdd"
			todir="${services.dir}"/>
	</target>

</project>
```

II Step two

Creating the Container

The container should be created in the >src package/>edu.virginia.vcgr.genii/>container

1. create a package for the container
2. add a java class named *nameOfTheService*ServiceImpl. For the addition example it's *AdditionServiceImpl.java*
    a. the class needs to inherit from the base container class **edu.virginia.vcgr.genii.container.common.GenesisIIBase**
3. in this class file specify what the service will actually be doing

After that the two folders in the >configuration folder should be changed:

1. the service should be added to the server-config.xml file

```
<genii:service name="AdditionPortType">
        <genii:resource-provider name="basic-db-resource-provider"/>
</genii:service>
```

2. the service should be added in the <genii:locator-registry> in the client-config.xml

```
package edu.virginia.vcgr.genii.container.addition;

import java.rmi.RemoteException;

import edu.virginia.vcgr.genii.addition.AdditionPortType;
import edu.virginia.vcgr.genii.addition.AdditionType;
import edu.virginia.vcgr.genii.addition.ResultType;
import edu.virginia.vcgr.genii.client.security.authz.RWXCategory;
import edu.virginia.vcgr.genii.client.security.authz.RWXMapping;
import edu.virginia.vcgr.genii.container.common.GenesisIIBase;

public class AdditionServiceImpl extends GenesisIIBase implements AdditionPortType
{
	public AdditionServiceImpl() throws RemoteException
	{
		super("AdditionPortType");
	}

	//Make sure that you specify that this can be executed

	@RWXMapping(RWXCategory.EXECUTE)
	public ResultType addition(AdditionType additionRequest)
			throws RemoteException
	{
		java.math.BigInteger number1 = additionRequest.getFirstAdditionElement();
		java.math.BigInteger number2 = additionRequest.getSecondAdditionElement();

		java.math.BigInteger sum = addingMyNumbers(number1, number2);
		ResultType result= new ResultType();
		result.setResult(sum);
		return result;
	}


	public java.math.BigInteger addingMyNumbers (
			java.math.BigInteger firstAdditionElement,
			java.math.BigInteger secondAdditionElement)
	{
		java.math.BigInteger sum =firstAdditionElement.add(secondAdditionElement);
		return sum;

	}
}
```

III Step Three

Writing the Client

In order to have a command-promt interactive client, the code for the client should go into >src / >edu.virginia.vcgr.genii/ >client/ >cmd/ >tools

A new tool should be created for the purpose and the easiest way again is copy-and-paste from an already existing tool.

The client-config.xml file in the >configuration folder has to be changed once again. The new tool has to be added to the <genii:tools>

```
package edu.virginia.vcgr.genii.client.cmd.tools;

import java.math.BigInteger;
import edu.virginia.vcgr.genii.addition.AdditionPortType;
import edu.virginia.vcgr.genii.addition.AdditionType;
import edu.virginia.vcgr.genii.addition.ResultType;
import edu.virginia.vcgr.genii.client.cmd.InvalidToolUsageException;
import edu.virginia.vcgr.genii.client.cmd.ToolException;
import edu.virginia.vcgr.genii.client.comm.ClientUtils;
import edu.virginia.vcgr.genii.client.rns.RNSPath;
import edu.virginia.vcgr.genii.client.rns.RNSPathQueryFlags;

public class AdditionTool extends BaseGridTool
{
	static private final String _DESCRIPTION ="Adds two numbers.";
	static private final String _USAGE ="add <num1> <num2> <adder-service-path>";

	public AdditionTool()
	{
		super(_DESCRIPTION, _USAGE, false);
	}

	@Override
	protected int runCommand() throws Throwable
	{
		add(new BigInteger(getArgument(0)),
			new BigInteger(getArgument(1)),
			getArgument(2));
		return 0;
	}

	@Override
	protected void verify() throws ToolException
	{
		if (numArguments() != 3)
			throw new InvalidToolUsageException();
	}

// This is basically the code on the client side. The function add() takes three parameters. The third one is the service path.
	private void add(BigInteger num1, BigInteger num2, String servicePath) throws Throwable
	{
//get the current path
		RNSPath path = RNSPath.getCurrent().lookup(servicePath,
RNSPathQueryFlags.MUST_EXIST);

//create a port
		AdditionPortType adder = ClientUtils.createProxy(AdditionPortType.class,
path.getEndpoint());

//get the result from the container
		ResultType rs = adder.addition(new AdditionType(num1, num2));
		System.err.println(num1 + " + " + num2 + " = " + rs.getResult());
	}
}
```

IV Step Four

Running the service

1. Start up a container and a client
2. Bootstrap the grid
3. Go to Containers/BootstrapContainer/Services; the service should be there
4. A resource should be created in order to test the service. The syntax for the ***create-resource*** command can be seen using the ***help*** command.

```
vcgr:$>help create-resource
create-resource [--rns | --url] <service-path-or-url> [<new-rns-path>]

vcgr:$>create-resource --rns AdditionPortType /MyAdder
```

5. then the service can be executed:

```
vcgr:$>add 4 5 /MyAdder
4 + 5 = 9
```