Software Architectures

Assignment 3: Service Oriented Architectures - Web Services

Assistants: Kennedy Kambona, Janwillem Swalens
Mail: {kkambona, jswalens}@vub.ac.be
Office: {10F732, 10F719}

Deadline: 23rd April 2015, 23:59

Software developers often reuse well-established external services within their applications instead of reimplementing the functionality. This of course has repercussions on the architecture of their applications.
In this exercise you will experiment, investigate and analyse some typical technologies used in *Service Oriented Architectures*. You will use a web service orchestration language BPEL (Business Process Execution Language) to compose results from two simple web services, and use the result in a third web service which will be called by the Web Portal application from the previous assignment (# 2)[1].

NOTE: This assignment uses a number of tools and technologies you have most likely not used before. We strongly recommend that you **start early** with the technical part!

Assignment

Like the previous assignment, this one consists of an implementation and a reporting part. Your implementation artifacts (project files) and a brief report have to be handed in.

Deadline: 23rd April 2015 at 23:59. The deadline is fixed and will not be extended

Deliverables: The report and implementation should be handed in as a single ZIP file. Submit the ZIP file on the Software Architectures course page in PointCarré, by clicking on *Assignments (Opdrachten) > Assignment 3*.
The file should follow the naming schema `(Firstname-Lastname_)*3.zip`, for example: `Kennedy-Kambona_Janwillem-Swalens_3.zip`.

[1]If you did not work on assignment 2 to its fruition, you have to use the original version of the Web portal application.

Team work You are allowed to work in a team of two. It is preferrable that you continue with your previous team-mate. Only one of you should submit the report on PointCarré, but be sure to mention **both names** in the report.

Grading The exercises will be graded and can be subject to an additional defense.

Context

This assignment **reuses** the scenario of assignment 2. You are asked to extend the web portal with functionality to query the `LibrarySearch` web service, which itself will query two other web services and compose their results.

Similar to the last assignment, you have to change the web portal implementation to provide yet another data backend. The new backend should only be restricted to search for books. It has to be implemented as an addition to the existing layers, i.e., in addition to searching for books using the web service, the search also has to use the configured sql database layer or your json data source to query for books.

The `LibrarySearch` web service orchestrates two existing library web services using a BPEL process. The two web services are the library web service of the Soft lab (`SoftLabLibrary`), and the national library of Belgium (`NationalLibrary`). Both web services are provided as Java web applications, and should not be changed. However, the BPEL process has to be optimized by you.

Exercise 1: BPEL Processes

Modify the existing LibrarySearch BPEL process[2]. Currently it invokes the two web services in a sequential manner, see Figure 1.

- Using BPEL constructs, modify the BPEL process to invoke the two library web services **in parallel**. The different structured activities in BPEL are documented in the WS-BPEL 2.0 specification[3].
  - Provide a small summary of your changes in the report. Indicate BPEL constructs you used, any errors you observed in the initial BPEL file (if any), and document changes you made in any other files as well (if necessary).
  - Make it clear why you chose to parallelize certain activities and not others. This means, describe briefly which data dependencies between the different actions need to be taken into account when parallelizing requests.
  - BPEL provides other constructs that can be used to optimize the invocation of multiple services apart from parallelization. State one, and how this could conceptually be done in your BPEL process.

[2]LibrarySearch/LibrarySearch.bpel
[3]`http://docs.oasis-open.org/wsbpel/2.0/OS/wsbpel-v2.0-OS.html#_Toc164738514`

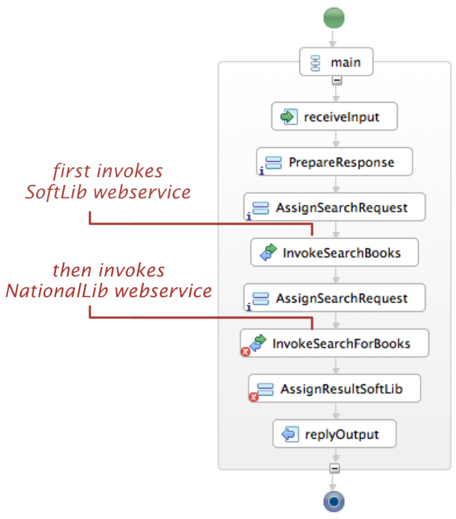

Figure 1: Sequential invocations of the LibrarySearch WS

- Using the WSDL querying tools indicated in the Tools section (Interacting with the Web services), query your webservice and observe the result.
  - State what data differences you observe from the results of the two web services NationalLibrary and SoftLibrary.
  - Observe the BPEL code when it is collecting the results (around line 140-148). Give a brief explanation for how the results are being processed. What are the advantages and disadvantages of this approach?

Exercise 2: Integration with Legacy Software

Your web portal application from assignment 2 only has access to a local database of books at the moment. To allow users to access a larger number of books, especially the books in the Soft library and in the national library of Belgium, you will integrate your `LibrarySearch` web service into the web portal.

- You will generate the web service client code required to call the web service in your workspace and then check if it works. See the tutorials in the Tools section 4 below (in case of trouble, join the help session or email the assistants with the specific problem).
- Add this client as another data backend, with only the functionality to search for books (not user login etc). In the end, searching in the web app for books should combine local results and the external web service results. Thus, the web portal should be able to use either of the two database layer implementations from the last assignment, and the new backend which queries the BPEL Web service i.e. (the

original (`db.sql AND db.webservice`) OR your new implementation (`db.json AND db.webservice`))

- Describe your implementation decisions in the report:
  - What did you change in the web portal application? How did your work from the last assignment help or hinder you with this task?
  - Which pattern or architectural style did you use?
  - Trying to query the LibrarySearch service with different query strings yields the same result always. Briefly explain this unexpected behaviour. (Hint: Have a look at the imported `NationalLibrary` and `SoftLabLibrary` projects)

Exercise 3: Architecture

Give an overview of the architecture of your new system of web service and local result mashups.

- Choose an appropriate visualization, for example an UML or just a block diagram (state which type of diagram you used). The diagram should allow to see the connections between the relevant elements in this architecture, e.g., the involved web services, and the relevant parts of the web portal application. It should minimally depict the architecture which results from your design choices in exercise 2.
- **Briefly** describe your diagram and name the relevant concepts/architectural styles. **Do not** go into the details for responsible classes etc., the architecture has to show logical components, not primarily implementation details. Often you will have no control over available web services, so treat the library web services as if they where provided by external providers.

Package the Result

Upload **a single ZIP-archive** to PointCarré "Assignment" option (Assignment 3). It should contain **your report** as a PDF, the changed `LibrarySearch` web service (packaged as an archive), and the web portal (exported as an archive from eclipse). Make sure that the uploaded file use this naming schema:
First-LastName_First-LastName-3.zip

Tools Setup

To manually install the tools and configure the project you have sets of instructions to follow. After this is done, proceed to the next section (Generating a Web Service Client).

1 Eclipse and Tomcat

This assignment was designed using **Eclipse for Java EE Developers** and **Apache Tomcat 6.0.43**. If you do not have these two software for your project, follow instructions in the Assignment 2 pdf file from Pointcarre in the section 'Preparing Eclipse and Tomcat'. Also, follow the steps described in "Preparing the Web Application" from the previous assignment Either way, before proceeding make sure that the Tomcat server runs the web portal application on localhost.

2 Preparing the Project in Eclipse

These steps indicate how to load and configure the project for this assignment in Eclipse.

1. Download the files for this assignment from PointCarré.
2. Import the WAR archive for the NationalLibrary and the SoftLabLibrary
   a) Choose File > Import... > Web > WAR file
   b) Select one of the WAR files, the target runtime should be the same Tomcat instance from last time.
   c) Finish the wizard, and repeat the same steps for the second WAR file[4].
3. Add the two applications to the Tomcat server. In the Servers-view right click on Tomcat > Add and remove...
4. The web portal application should also be configured to run on this Tomcat instance to be able to debug it.
5. For the BPEL process we will need another Tomcat instance, thus, this Tomcat in Eclipse has to use different ports. Double click on the Tomcat server in the Servers-view > on the right hand side, change the ports like follows:
   - Tomcat admin port: 8105
   - HTTP/1.1: 8180
   - AJP/1.3: 8109
6. Now, the Web services and the portal application should be deployed and you can start the server (possibly in debugging mode).
7. The web portal should now be reachable via `http://localhost:8180/web_portal/`.

[4]If you get "Loading Descriptor" errors, it is because you have an older Java version. Luckily, these errors can be ignored and the services will still deploy.

3 Deploying the BPEL process

To deploy and test the BPEL process you will modify for exercise 1, you will use the Apache ODE BPEL engine deployed in Tomcat. This environment does not provide a lot tooling, but it is simple to setup and use to deploy.

1. You can download the WAR distribution of Apache ODE from `http://ode.apache.org/getting-ode.html` (tested version: 1.3.6).

2. Extract the ZIP archive and copy the ode.war to the webapps folder of your copy of Tomcat (created from assignment 2).
   Sidenote: The Tomcat instance started from Eclipse uses only the binaries of Tomcat, but the actual working directory is somewhere in Eclipse workspace. Thus, it is safe to reuse it.

3. Navigate to the Tomcat folder and start it from the command line.
   You may need to change permissions of the .sh files if it doesnt work the first time.

4. Copy the content of the LibrarySearch archive into the ODE processes folder: tomcat/webapps/ode/WEB−INF/processes/LibrarySearch

5. The process will be deployed automatically and can be seen at
   `http://localhost:8080/ode/`

6. You can get the WSDL for the web service in the URL
   `http://localhost:8080/ode/processes/LibrarySearchService?wsdl`

7. To test the BPEL process, delete the LibrarySearch folder and the LibrarySearch.deployed file, and then copy the changed version from the Eclipse workspace into the processes folder.

For NetBeans users, the GlassFish server also includes standard debugging support for BPEL processes, including breakpoints.[5].

3.1 Interacting with the Web services

You can use tools to interact with the two NationalLibrary and SoftLibrary web services (or even your composed web service after doing Ex.1) and investigate their result data. The easiest way is through browser extensions Wizdler for Chrome and SOA Client for Firefox. An alternative standalone tool is soapUI[6]. We outline how to use Wizdler below, but the process is similar in the other tools.

1. For Chrome: Install the Wizdler extension[7]

[5] `http://soa.netbeans.org/soa/`
[6] SoapUI: `http://sourceforge.net/projects/soapui/`
[7] Wizdler: `http://bit.ly/wizdler`

2. Make sure the web_portal application and the `NationalLibrary` and `SoftLabLibrary` services are running
3. Visit any of the URLs of your web services
   - `http://localhost:8180/NationalLibrary/services/LibraryService?wsdl`
   - `http://localhost:8180/SoftLabLibrary/services/soap?wsdl`
4. To activate Wizdler, click on the new icon that appears in the browsers URL input field (next to the bookmark star icon).

4 Generating a Web Service Client

For the second exercise you will generate the client code automatically for your new `LibrarySearch` webservice from the first exercise, to call the service from your web portal app. We recommend using Eclipse, which provides facilities to generate the client code for Web services. For a brief tutorial visit: `http://www.craigsprogramming.com/2011/03/tutorial-consume-any-web-service-using.html`.

Depending on your strategy for handling the result data, you might want to adapt the BookList type in LibrarySearchArctifacts.wsdl which is part of the `LibrarySearch` Web service. This file is also the input file you want to use for generating the client code.

Beside Eclipse, there are also command-line tools to achieve the same results, e.g., see `http://axis.apache.org/axis2/java/core/docs/userguide-creatingclients.html`.