Technical University of Denmark
DTU Informatics
H. Baumeister
Associate Professor

Fall 2015
Homework week 5
September 23, 2015

02267: Software Development of Web Services

Exercises week 5 + 6

This weeks and next weeks exercises are designed to take you step by step through the different possibilities BPEL offers, starting with receiving a request and replying to it and ending with calling other Web services concurrently. In week 5, those BPEL features that are needed for 5.1 — 5.3 will be covered; in week 6, the remaining features.

- Note: The idea of the Ham and Eggs exercises is to change the Ham and Eggs process/projects through the exercises with the new requirements instead of each time starting from scratch.

- Note that the new exercises are not a conservative extension of the previous ones in that the behaviour of the old exercises can change in the requirements for the new exercises. Think of this as an example of an agile software development process, where you incrementally add new functionality to the system, but also change existing functionality as the system evolves.

5.1 Ham and Eggs in BPEL (part 1)

Create a BPEL process that offers an operation hamAndEggs with the number of used eggs as a parameter and returning true.

1 Implement and deploy the BPEL process (cf. the instructions on "Creating a BPEL Process in NetBeans" on the course Web page)

2 Test the process using a JUnit test

5.2 Ham and Eggs in BPEL (part 2)

Create a BPEL process that offers an operation hamAndEggs as in the previous exercise, but this time return

```
<hamAndEggs>
  <eggs>n</eggs>
</hamAndEggs>
```

where n is the number of eggs used in the preparation of the dish.

1 Implement and deploy the BPEL process

2 Test the process using a JUnit test

5.3 Ham and Eggs in BPEL (part 3)

- Create a BPEL process that offers an operation hamAndEggs as in the previous exercise. This time the BPEL process should use two other Web services (Cleaner and Cook) to prepare the dish. The Cleaner Web service offers two operations washDishes and ventilateAir and the Cook Web service offers the operations breakEgg and bake.

- The washDishes, breakEgg take as argument the number of dishes to wash and the number of eggs to break, respective. The bake operations takes as an integer argument the number of seconds to bake the omelette. All operations return true when they are done. The ventilateAir operation takes as argument the number of seconds to ventilate and returns immediately (is a one-way operation). Here we assume that the ventilator is switched on for the amount of seconds passed as an argument and then switched off automatically.

- The process of cooking starts with washing the dishes, then breaking the eggs, then backing the omelette and finally ventilating the air.

- Tasks

  a) Implement the Web services Cook and Cleaner with their respective operations as Java classes. Each of the operations should write **a message** to **System.out** when they are called. **The print outs appear in the log file of the Glassfish server**. Deploy the Web services.

  b) Implement the hamAndEggs process as a BPEL process and deploy it. The process should return true when finished.

  c) Test your process using a JUnit test.

Technical University of Denmark
DTU Informatics
H. Baumeister
Associate Professor

Fall 2015
Homework week 6
September 23, 2015

02267: Software Development of Web Services

6.1 Ham and Eggs in BPEL (part 4: using a for-loop)

This is an extension of the exercises of week 5. Instead of breaking all eggs at once, break each egg individually by calling the break egg operation for each egg (i.e. number of egg times).

6.2 Ham and Eggs in BPEL (part 5: using exception handling)

- Starting from the last exercise, change the break egg operation of the cook service, so that a fault can be generated when an egg smells bad. The XML for the detail element of the fault message should be

  ```
  <eggSmellsBad/>
  ```

- Implement the break egg operation of the cook service such that the fault that an egg smells bad is generated every second time
- Implement a fault handler in the ham and egg process that calls the ventilate operation of the cleaner service each time an egg smells bad
- Return the number of good eggs used in the omelette in the result, e.g. if one calls the hamAndEggs operation with 4 eggs and one of them smells bad, return the XML

  ```
  <hamAndEggs>
    <eggs>3</eggs>
  </hamAndEggs>
  ```

6.3 Ham and Eggs in BPEL (part 6: using the flow construct)

Based on the previous exercise, implement the ham and eggs process so that the operations bake and ventilate air are called concurrently

6.4 Ham and Eggs in BPEL (part 7: using correlation sets)

The previous process has proved very sucessfull with people, so the idea is to make this a commercial product. The customer will have to pay when the hamAndEggs operation returns successful. This is done by implementing an operation `pay` which has to be called after the hamAndEggs operation has replied to the customer. In addition, the argument to the hamAndEgg operations contains customer information, i.e., a customer number and an order id (which is specific to each customer). Note that the customer is responsible for managing his own order id's. The hamAndEggs returns the amount of money the customer has to pay, e.g. $20 * number\_of\_good\_eggs\_used$, and the pay operation includes the customer and oder id plus credit card infomation (to keep things simple, this is just the card number). The pay operation returns true to indicate that payment was successful.