CSE446/598 Project 2 (100 Points)

Spring 2020

Assignment 3 Due: February 22, 2020 by 11:59pm, plus a one-day grace period

Assignment 4 Due: February 29, 2020 by 11:59pm, plus a one-day grace period

---

Introduction

The aim of this assignment is to make sure that you have read the text and the slides, and you have understood the software integration and workflow concepts in Workflow Foundation and in BPEL covered in the lectures and in the text. By the end of the project, you should have a good understanding of the concepts and have applied these concepts in developing operational software in Workflow Foundation. Both parts of the project are individual work. No collaboration is allowed. All students must submit independently developed applications. You cannot repeat the examples given in the book or lecture slides. You can follow the examples, but significant changes must be made.

Part 0 Practice Exercises (No submission required)

1. Reading: Textbook chapters 7 and 8.
2. Practice exercises: Questions at the end of Chapters 7 and 8.
3. Follow the lectures 2-2 and 2-3 and the text chapter 7 to implement the Image Verifier service using workflow.
4. Write a Web site application to consume the Image Verifier service.
5. Implement a new workflow service based on the Image Verifier service.
6. Download the Mortgage example and test the mortgage example.
7. Explore the ASU Repository at: neptune.fulton.ad.asu.edu/WSRepository/repository.html
8. Follow the lectures 2-4 through 2-6, the text chapter 8, and the BPEL tutorials in the assignment folder to implement a BPEL process.
9. Write a client to consume the BPEL service.

Note: Workflow Foundation module may not be included in your Visual Studio 2017. In this case, you cannot create a new Workflow Foundation project. However, you can use your Visual Studio to open a Workflow Foundation. Then, Visual Studio will add Workflow Foundation module automatically.

Assignment 3 Workflow Composition Project (Submission Required)

Due: February 22, 2020 by 11:59pm, plus a one-day grace period

In this part of the project, you will develop a Web service and a sensible application based on your own ideas. You can use the challenging services that you implemented in CSE445/598 Project 3 or any services with similar level of complexity. A copy of CSE445/598 Project 3 document is placed in the "Barrett Honor's Project" Folder for your reference. You **cannot** use the number guessing game (application and its services) that you used in assignment 1 in this assignment. You cannot use the same example found in the lecture slides. For the application, you can use any type of human interface: console, Windows Forms, ASP .Net website, HTML5, or MVC1/2. In the application, you must access the service that you develop in this project, and a service in the ASU Service Repository or one that you discovered from public repository, as shown in Figure 1.

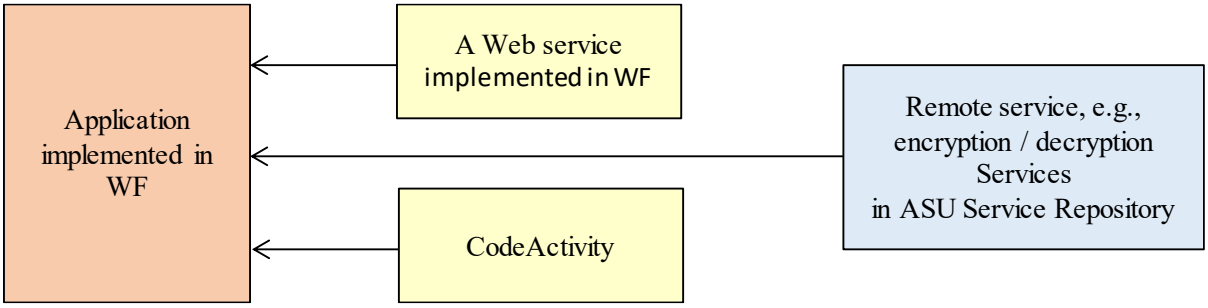

Figure 1. Workflow-based application

You must use the **Workflow Foundation** to develop **both** the application and the Web service. You must also use at least one **CodeActivity** in one of the workflows (in the application workflow or in the Web service workflow). The application must combine at least the two services to deliver the integrated functionality. If needed, you can use more services and other components. For all services, you can choose to use either SOAP or RESTful services. The assignment is broken down into the following tasks.

1. Use Workflow Foundation and flowcharts to develop a service that provides a sensible function to be used in your application. [10]

2. Use Workflow Foundation to develop a sensible application using at least two services (or service operations). You can use either interface-first or workflow-first template. One service is developed in question 1 of this assignment. The other service is a remote service from a service repository, such as those in the ASU Service Repository. For the human user interface, you can use any type of interface. Your service will be hosted on the localhost / IIS Express. [20]

3. Develop a sensible CodeActivity within the workflow application (from question 2). [10]

4. User Manual: You must write a mini user manual in an html web page. The main components include: (1) Introduction to the application and the Web service that you have developed. (2) A description of the interface. (3) A diagram illustrating the application logic, workflow, and the code activity that you have implemented. (4) Testing instruction (How the TA can test your application and services) and at least two test cases (inputs and outputs) and screenshots you used to test your software. [10]

Assignment 3 Canvas Submission:

Zip all the following items into a single file folder for submission.

1. The Visual Studio Solution folders with all the files, developed in questions 1, 2 and 3.
2. The user manual in question 4.

Assignment 4 BPEL Composition and Integration (Submission Required)

Due: February 29, 2020 by 11:59pm, plus a one-day grace period

In this part of the project, you will develop a BPEL process and its WSDL interface for a similar project that you developed in assignment 3 of the project. The architecture is shown in Figure 2.

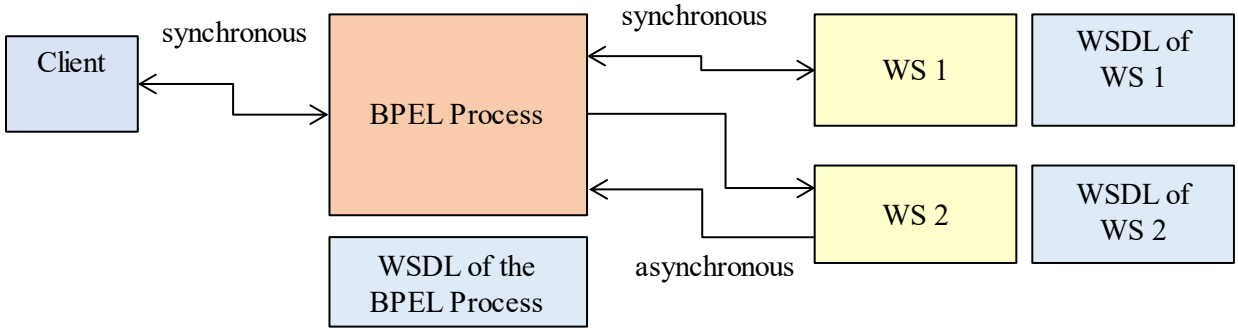

Figure 2. BPEL application

In this assignment, you can take one of the two options:

(1) You can also directly use an XML editor to write the BPEL process and the part of the WSDL required in this assignment. You do not need to create the operational executable software.

(2) You can use Oracle SOA Suite and JDeveloper (or Eclipse BPEL Designer) to create the process in XML and to generate the WSDL file.
Note: Oracle SOA Suite and JDeveloper are large enterprise software packages. It may take significant effort to install and to learn using it. It may take significantly longer time to take option (2). You take this option if you have intrinsic motivation for learning BPEL and Oracle SOA Suite programming.

The project is broken down into the following tasks.

1. Use BPEL process (in XML language) to model the application that you developed in Part 1. You will consider the application in assignment 3 as the BPEL process that offers a WSDL service. You will write a simple client to call this BPEL-based service. In this question, you will define the complete BPEL process in XML, including partnerLink to all the Web services that you use, variables, receive-reply for input and output, and the workflow that performs the application's function. [30]

2. Provide the WSDL file for your BPEL process. You do not have to provide the complete WSDL file. You just need to provide the roles and the port types to the **client** and to the **two services** that the process uses. [15]

3. Your BPEL process must access at least two Web services, one in synchronous mode and one in asynchronous mode. [5]

Note, you do not need to provide the operational software in this assignment. However, the process, the WSDL interfaces must be complete and correct, as we discussed in the lecture and in the test. The main purpose is to exercise the BPEL-based workflow concepts and the partners' definitions.

Assignment 4 Canvas submission:

Zip all the following items into a single file folder for submission.

1. The BPEL process in XML file.
2. The WSDL file for your BPEL process.
3. WSDL files of the two Web services.

General Submission Requirement

All submissions must be zipped into a single file electronically submitted to the Canvas assignment folder. All files must be zipped into a single file for submission.

For programming assignments, the entire solution folder with all the files must be included. To make sure that you have all the files in the zip file, unzip the file on a different machine or in a different directory, and test if you can run the project in a different location.

Submission notice: A programming assignment typically consists of multiple distributed parts. They may be stored in different locations when you create them. You must copy these projects into a single folder for Canvas submission. To make sure that you have all the files included in the zip file and they work together, download your own submission from the Canvas. Unzip the file on a different machine, and test it and see if you can run the solution in a different location because the TA will test your application on a different machine.

Grading and Rubrics

Each sub-question (programming tasks) has been assigned certain points. We will grade your programs following these steps:

(1) Compile the code. If it does not compile, 50% of the points given for the code under compilation will be deducted. Then, we will read the code and give points between 50% and 0, as shown in right part of the rubric table.

(2) If the code passes the compilation, we will execute and test the code using test cases. We will assign points based on the left part of the rubric table.

In both cases (passing compilation and failed compilation), we will read your program and give points based on the points allocated to each sub-question, the readability of your code (organization of the code and comments), logic, inclusion of the required functions, and correctness of the implementations of each function.

Please notice that we will not debug your program to figure out how big or how small the error is. You may lose 50% of your points for a small error such missing a comma or a space!

We will apply the following rubrics to **each sub-question** listed in the assignment. Assume that points assigned to a sub-question is *pts*:

Rubric Table

Major	Code passed compilation				Code failed compilation		
Points	pts * 100%	pts * 90%	pts * 80%	pts * 70% - 60%	pts * 50% - 40%	pts * 30% - 10%	0

Each sub-question	Meeting all requirements, well commented, and working correctly in all test cases	Working correctly in all test cases. Comments not provided to explain what each part of code does.	Working with minor problem, such as not writing comments, code not working in certain uncommon boundary conditions.	Working in most test cases, but with major problem, such as the code fail a common test case	Failed compilation or not working correctly, but showing serious effort in addressing the problem.	Failed compilation, showing some effort, but the code does not implement the required work.	No attempt

Late submission deduction policy:

- Grace period (Sunday): No penalty for late submissions that are received within 24 hours of the given due date;
- 1% grade deduction for every hour after the first 24 hours of the grace period!
- No submission will be allowed after Tuesday midnight. The submission link will be disabled.