CSE445/598 Assignment/Project 3 (100 Points)

Fall 2013

Part 1 Due by 11:59pm of October 19, 2013, in Blackboard

Part 2 Due by 11:59pm of October 26, 2013, in Server and in Blackboard

---

Introduction

The aim of this assignment is to make sure that you understand and are familiar with the concepts covered in the lectures, including service development, service registration, service deployment, service hosting, service proxy, service binding, service invocation, and application building using your own services and public services. By the end of the assignment, you should have applied these concepts in programming your services, deploying your services, and have used your own services and public services to compose your SOC applications.

This project is partly an **individual project** (80%) and is partly a **group project** (20%). Each project team will consist of two, three, or four members, based on the team building exercise. Although a team works together to complete the project in a collaborative and coordinated manner, a large part of the project will be done individually. A declaration must be given, which specify the portion of individual efforts in the group part of the project. A percentage of contribution of each member (e.g., 35%, 35%, and 30%) must be given for the joint part of work, which will be used to **scale** the assignment grades of the group part. Each service must have a single author associated with it. The entire project must be deployed into the given virtual server: WebStrar or V-Lab. Each member must submit your own services into the Blackboard in part 1. In part 2, each team member must submit your part into the Blackboard, and one of the team members must also submit the joint part of the project into the Blackboard after the project has been deployed into the server. When submitting the services and service test pages into the blackboard, you must submit the folder (folders) with the source code (We will read the source code from Blackboard and test the code from the server). For the server deployment, you can submit folders with the source code or with the precompiled files in the folder.

Section I Practice Exercises (No submission required)

No submission is required for this part of exercises. However, doing these exercises can help you better understand the concepts and thus help you in quizzes, exams, as well as the assignment questions.

1. Reading: Textbook Chapter 3 and Appendix C.

2. Answer the multiple choice questions 1.1 through 1.16 of the text section 3.10. Study the material covered in these questions can help you prepare for the class exercises, quizzes, and the exams. Answer keys to the questions can be found in the course web page. To better learn these concepts, you should do the exercises based on your understanding, and then check the answer keys.

3. Study for the questions 2 through 16 in text section 3.10. Make sure that you understand these questions and can briefly answer these questions. Study the material covered in these questions can help you prepare for tests and understand the homework assignment.

4. Questions 17 through 19 are largely covered by the assignment questions in Part II.

Section II Assignment/Project Questions (Submission required)

The purpose of this project is to exercise service development, service deployment, service discovery, remote binding, and application composition using your own services and external public services. Some of the services to be developed can be synthetic, e.g., banking service, while others can be realistic, e.g., en/decoding, en/decryption, and product catalog. However, you should make your services and overall application as realistic as possible.

In this project, each team should develop a requirement document that outlines the overall project, a service directory, and the architecture of the Web application, and a set of services.

1 Main Page (group assignment question) [15 points]

This assignment question must be done jointly by the entire group. The main page (inkling the requirement document) can be an html page, an aspx page, or a page that can be opened by a standard Web browser. The main page must be named use the default convention, e.g., index.html or default.aspx, so that the page can be accessed without using its directory name. The main page must contain the following contents:

1.1 The project name, team name, and team member names, the link (URL) to the deployed main page in the server (WebStrar or V-Lab). [1]

1.2 Description of the service-oriented computing system that your team plans to develop: The requirement document. [3]

1.3 A diagram showing the overall system design, its layers, components, and the connections among the components. A sample diagram is given in figure 1. Your team must come up with your own system. You will not implement the entire system outlined here in this assignment. The focus of this assignment is the service development and deployment. [3]

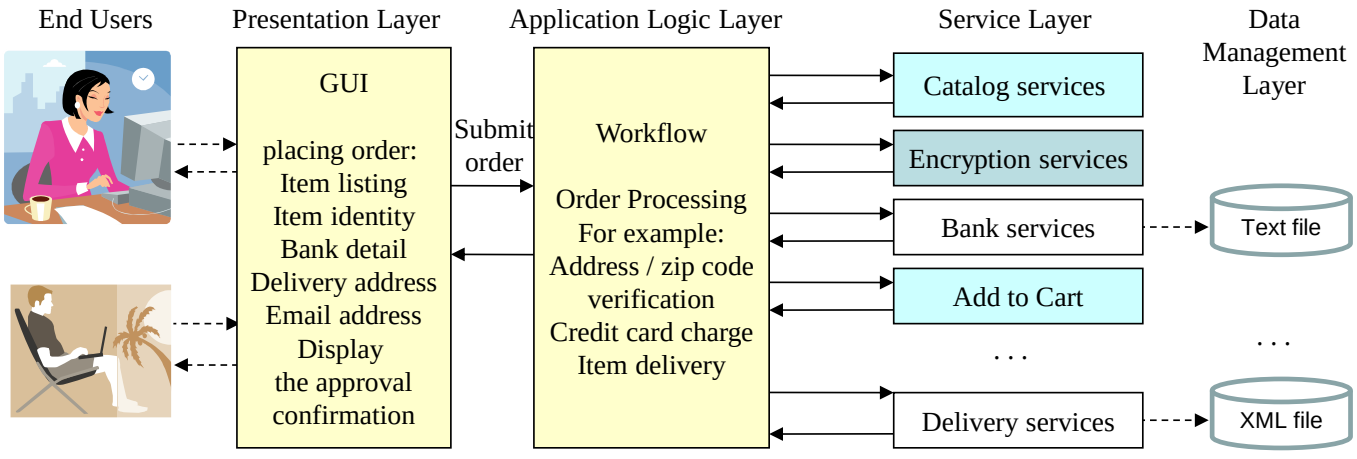


Figure 1. A sample of a four-tier service-oriented computing system

1.4 Service directory (a table) listing the services and links to the test page and services. The schema and a list of example services are shown in the table below, which is similar to the service list in ASU Repository in text Appendix C. [8]

Provider name	Service name and deployed URL in server	Service description	Server link to test page	Language and Platform
John Doe	*Encryption* *Webstrar URL*	*Cipher encryption and decryption*	*Try It*	*WCF SVC*
John Doe	*SolarPower* *Webstrar URL*	*Output annual KW number for given panel size*	*Try It*	*WCF Workflow*
Jane Miller	*findStore* *V-Lab URL*	*Use an existing online service or API to find the provided storeName*	*Try It*	*WCF SVC*
…				
Eric Smith	*sendText* *V-Lab URL*	*Send a text to a cell phone*	*Try It*	*Java Eclipse*
…				

(1) Provider: Name of the service developer (your name). One name per service is allowed.

(2) Service Name and link: The URL of the deployed service in WebStrar or in V-Lab to the service must be given.

(3) Service description: Describe the function of the service, including the input and output.

(4) Link to test (*Try It*): The link to the test page of the service must be given behind TryIt. The TryIt page must test the services deployed to the server, not the local host service.

(5) Language and Platform: The programming language and the development environment used to develop the service. Only C# and Java are allowed in this assignment. WebStrar (v-lab) supports IIS and C# only. If you use Java, you must find your own hosting server.

2 Required Service Development (individual assignment question) [40 points]

Note: Question 2 is the Part 1 and must be submitted by the first due date specified.

This is an individual task within the group assignment. Each team member must independently implement and deploy their services. There are two types of services to develop: required services and elective services.

2.1 Required Services [30 = 15+15]

A set of required services and their requirements are listed in a separate document named "List of Required Services". Each team member must choose and implement **two** services from the given list. The members in the same team must choose **different services** from the given list.

2.2 TryIt Pages of the Required Services [10]

For each service and operation that you developed, you must develop a TryIt test page to allow the human user to test the service. The TryIt test page must contain the following contents:

(A) A sentence to describe the functions of each service (operation);

(B) Part 1 submission: Localhost URL of the WSDL file,

(C) For Part 2 submission, the service URL of each service (or its WSDL address) used in the TriIt page must be the URL in the server. no localhost address can be used. For the required services, you will develop WSDL services. For the elective services in the next section, you will be asked to develop at least one RESTful service.

(D) Method names, with parameter type list and the return type for each endpoint.

(E) Text boxes for entering inputs

(F) Invoke buttons to call the services

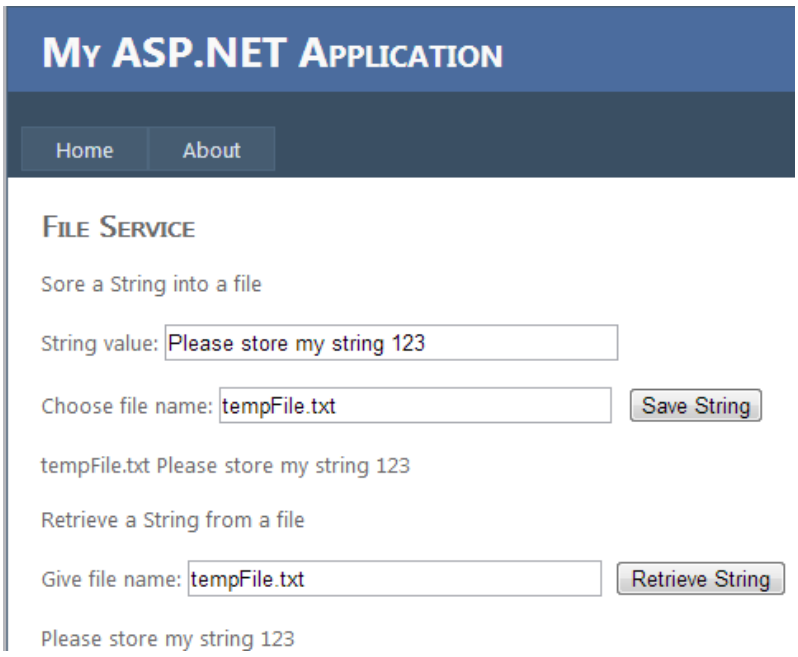
(G) a place (e.g., label) to display the service response (output)

The two services and the TryIt page will be tested on the localhost in part 1 submission. To make sure the TA can test your services and TryIt pages on their computer, you must specify the localhost port number statically in the following steps:

(1) In Solution Explorer, click the name of the application.
(2) In the Properties panel, click the down-arrow beside Use dynamic ports and select False from the dropdown list. This will enable editing of the Port number property.
(3) In the Properties panel, click the text box beside Port number and type in a port number.
(4) Click outside of the Properties panel. This saves the property settings. Each time you run a file-system Web site within Visual Studio, the ASP.NET Development Server will listen on the specified port.

You can combine the test pages of multiple services and operations into one test page. You must make sure that you have a GUI to test every service operation that you developed for credit. An example of 3.E, 3.F, and 3.G is given in the figure and in the link below:

http://venus.eas.asu.edu/wsrepository/Services/FileServiceTryIt/



More examples of test page are at:

http://venus.eas.asu.edu/WSRepository/AjaxIn/Default.aspx

http://venus.eas.asu.edu/WSRepository/Services/ImageVerifier/Tryit.aspx

http://venus.eas.asu.edu/WSRepository/Services/RandomString/Tryit.aspx

3 Elective Service Development (individual assignment question) [35 points]

3.1 **Elective Services**: Each member must develop at least three elective services (or service operations). For the elective services, the team members must discuss what services are needed based on the scope defined in question 1 and who should develop what services. All team members must define and implement different elective services. The elective services should be related to the scope defined in question 1. One can choose the required services as elective services as long as no other team members are implementing them. If a member chooses to implement the services in the list of required services, this service does not have to be related to the application scope that your team plans to develop. At least one service must be converted to the RESTful service. For the rest of the services, you can choose to develop WSDL services or RESTful services. For developing RESTful service, please follow Lecture Slides L10 or textbook section 7.3.3.2. [25 points]

The difficulty level of the elective services (operations) that you developed will be rated by the instructor and the teaching assistant into one of the three difficulty levels:

(1) Easy: The method (operation) in this service implements a simple math function and can be done using less than 50 lines of code, for example, Fahrenheit and Celsius temperature conversion. [5 each]

(2) Moderate: There are algorithmic issues to address and the code for each method will be at least 50 lines, for example, encryption/decryption service, efficient sorting, and equation system solving. If a service operation can be done in less than 50 lines, but you use more than 50 lines, it will be counted as an easy service. [10 each]

(3) Challenging: Services that will use states, such as creating a simulated (synthetic) banking service that allows users to sign up, create an account, deposit fund, spend fund, etc; or services that make use of multiple available services (operations) or APIs provided by other providers, such as Microsoft services (e.g., Bing map service), Google code's APIs, Amazon's services, or the ASU services. These services and APIs may or may not have WSDL interface. Your services must provide WSDL interfaces. The data received from other services should be processed and combined before returning to the clients. The given required services are examples of challenging services: If you choose to implement these services, they count as challenging services. Database is not allowed in this assignment. If you need to store states, you can either a text file (See Chapter 3 L12 Slides) or an XML file (Read Text Chapter 5, Section 5.4). [15 each]

To obtain the full point 25 in question 3.1, you need to develop at least two services and at most four services. It implies that you cannot write five easy services in this question. If you write easy services only, the maximum points you can obtain in this question is 20 out 25. On the other hand, you can obtain 25 points at most, even if you develop more services and more difficult services than required. From the format point of view, you can either define these services methods in one big service, or define them as separate services. Each service and methods must be commented in detail, including the functionality, parameters, types, and the return value type.

3.2 TryIt Pages for Elective Services [10]

As described in question 2.2, you must develop TryIt test page for elective services. Notice that, after you have deployed the services into the server, you must change the localhost-based URL to the server URL, so that your test page can be accessed from the Web.

4 Server Deployment [10 points]

All services, TryIt GUI pages, and the main page must be deployed into the server (WebStrar or V-Lab) in part 2.

Services and TryIt GUI pages deployment (individual)	[5]
The main page deployment (group)	[5]

Submission Requirements

Part 1: Due: October 19, 2013. Each member must submit the two required services and the TryIt pages in to the Blackboard.

Part 2: October 26, 2013. There are multiple submissions:

(1) Each member must submit the **code** of the elective services and their TryIt pages in to the Blackboard. We will read the code from this submission. Do not submit the code of required services and their TryIt pages, even if you have updated them. We will not re-grade your required services.
(2) Each member must upload the latest required services, the elective services and their TryIt pages in server. The TryIt pages must have the global links of the services. If you still have the localhost link in the TryIt pages, you will lose points, because they would not work.
(3) One member must submit the main page (question 1) into the blackboard. We will use this submission to find where your main page is in the server.
(4) One member must upload the main page (question 1) into the server. We will use this server page to test your entire application and services. We will test your TryIt pages and services in the server.

Grading

We will grade each program or service following these steps:

(1) We will read the code and give points based on the points allocated to each component, 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.

(2) Compile the code. If it does not compile, 40% of the points given in (1) will be deducted. For example, if you are given 20 points in step (1), your points will become 12 if the program fails to compile.

(3) If the code passes the compilation, we will execute and test the code. This can be on local machine or on the Web, depends on the assignment specification. If, for any reason, the program gives an incorrect output or crashes for any input, 20% of the points given in (1) will be deducted.

Please notice that we will not debug your program to figure out how big or how small the error is. You may lose 40% or 20% of your points for a small error such missing a comma or a space!

Late submission deduction policy:

- No penalty for late submissions that are received within 24 hours of the given deadline;
- 1% grade deduction for every hour after the first 24 hours of the grace period!