CSE445/598 Assignment/Project 3 (100 Points)

Spring 2014

Part 1 Due by 11:59pm of October 4, 2014, in Blackboard. One submission per team.

Part 2 Due by 11:59pm of October 18, 2014, in Blackboard. Individual Submissions.

Part 3 Due by 11:59pm of October 25, 2014, in Server and in Blackboard. Individual Submissions.

Part 4 Due by 11:59pm of October 25, 2014, in Server and in Blackboard. One submission per team.

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** (75%) and is partly a **group project** (25%). Each project team will consist of two or three 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 final project must be deployed into the given Web server: WebStrar. 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. The project will be completed in following parts.

Part 1 Requirement Document (group assignment question) [10 points]

Part 1 Due by 11:59pm of October 4, 2014. One team member submits the requirement document in Word or PDF into Blackboard.

1 The requirement document will contain the following contents.

1.1 Description of the service-oriented computing system that your team plans to develop. [2]

1.2 A diagram showing the overall system design, its layers, components, and the connections among the services. A sample diagram is given in Figure 1. Your team must come up with your own system. You will not implement the application logic layer of system outlined here in this assignment. It will be implemented in assignment 5. The focus of this assignment is to develop and deploy the **services** outlined in this requirement document. Thus you must pay more attention to the services in this assignment. [2]

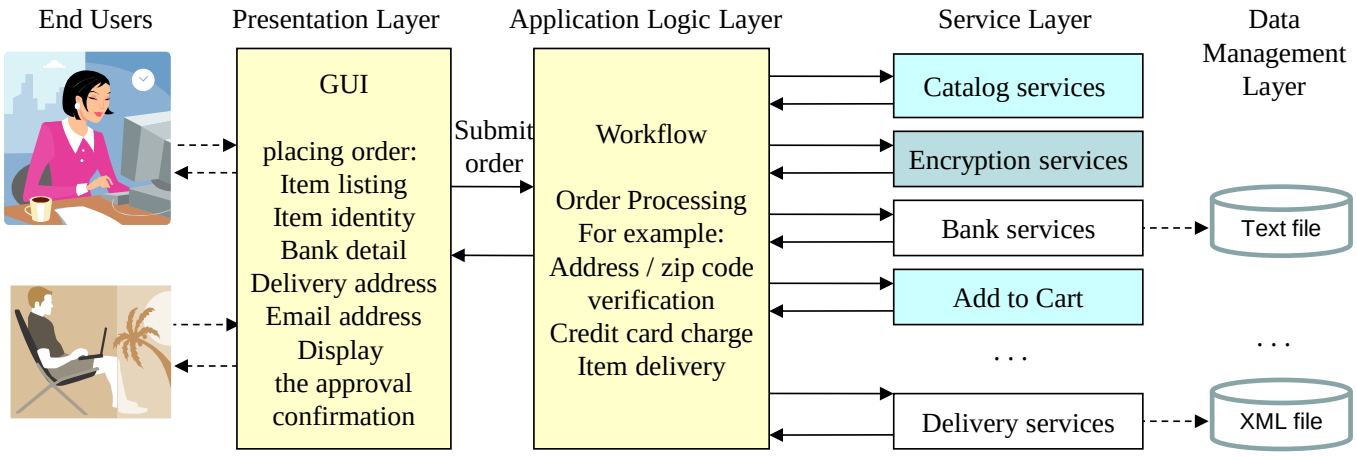


Figure 1. A sample of a four-tier service-oriented computing system

1.3 Create a service directory (a table) listing the services that your team and each member plan to develop. Each member must be responsible for three to five services in this table. Each service must be assigned to one and only one team member, as the service provider. Each service must be described in such detail that shows the feasibility of implementing the service in a reasonable amount of time, e.g., 10 hours. All the services must be related to the application that you outlined in the question above and support the composition of the application. The contents (test in *italic*) of the table below are the examples and you need fill the table based on your work. You can leave the TryIt column blank for now and fill it out in the final submission. [6]

This page is deployed at: To fill out the address in later submission.				
This project is developed by: Put your team name here.				
Provider name	Service name, with input and output types	TryIt link	Service description	Planned resources need to implement the service
Member 1's name	Encryption and decryption: Input: String Output: String	TryIt	Cipher encryption and decryption	Use library class and local component to implement the service
Member 1's name	SolarPower Inputs: zip code and size Output: integer	TryIt	Output annual KW number for a given panel size at a given zip code location	Retrieve information from national database at: http://graphical.weather.gov/xml/
Member 1's name	findStore Input: zipcode Output: list of string	TryIt	Use an existing online service or API to find the locations of a given store name	Use the service from Yelp site at: http://www.yelp.com/
Member 2's name	…		…	…
Member 3's name	textToPhone inputs: string Account, string Password, string Receiver, string Subject, string Body Output: boolean	TryIt	Send a text to a cell phone and return if the message is sent successfully	Use google gmail account and carrier services: phone_number@tmomail.net phone_number@messaging.sprintpcs.com phone_number@vtext.com phone_number@txt.att.net
…	…		…	…

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

Part 2 Due by 11:59pm of October 18, 2014. Individual submissions: Each team member must submit one zip file into Blackboard. The zip file must contain all the **required services** and their TryIt pages developed in this part of the assignment.

2 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. The required services will be developed Part 2 and the elective services will be developed in Part 3.

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. For the elective services in the next section, you will be asked to develop at least one RESTful service.

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) The URL of the service

1. For Part 1 submission: use the Localhost URL of the WSDL file.
2. For the final submission in Part 3, the service URL of each service (or its WSDL address) must used in the TryIt page must be the URL in the server. No localhost address can be used. For the required services, you will develop WSDL/SOAP services.

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

(D) Text boxes for entering inputs

(E) Invoke buttons to call the services

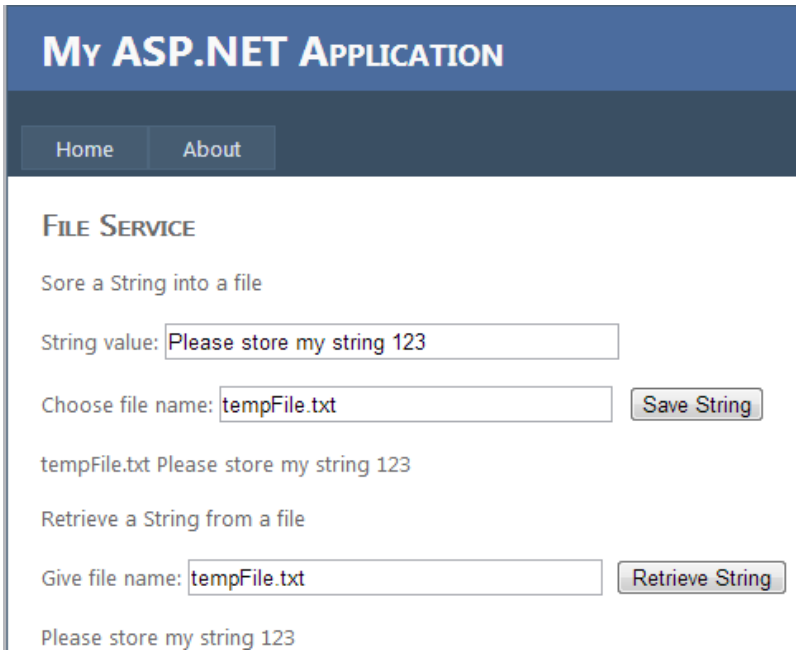
(F) a place (e.g., label) to display the service response (output)

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 TryIt 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

Part 3 Elective Services Development (individual assignment questions) [35 points]

Part 3 Due by 11:59pm of October 25, 2014, in Blackboard and in the WebStrar server.

Individual submissions: Each team member must submit one zip file into Blackboard. The zip file must contain all the **elective services** and their TryIt pages developed in this part of the assignment. All the services (required and electives) must be deployed into the WebStrar server.

3 In this part, you will develop the elective services and their TryIt pages.

3.1 **Elective Services**: Each member must develop at least two 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 for 3.1]**

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 points 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 points 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 points 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. 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 Development of TryIt Pages for Elective Services [5 points for 3.2]

You must develop TryIt test page for elective services. The requirement is the same as defined in question 2.2, but all the URLs must be based on the server URLs because the services have been deployed to the server. You must also have the URL of the Main page (See next question) linked to the test pages, so that one can return to the main page from the TryIt page.

3.3 Deployment of Required Services, Elective Services, and TryIt pages [5 points for 3.3]

All the services (required and elective) and their TryIt pages must be deployed into WebStrar server. You must deploy the services first. Before deploying the TryIt pages, you must change all the service references from localhost addresses to the server addresses, so that the services can be tested from the server.

Part 4 Main Page and Server Deployment (group assignment question) [15 points]

Part 4 Due by 11:59pm of October 25, 2014, in Blackboard and into the WebStrar server. One team member submits the requirement document in Word or PDF into Blackboard.

4 This assignment question must be done jointly by the entire group. The main page must be an html page. The main page must be named index.html, so that the page can be accessed through the link: http://webstrarX.fulton.asu.edu/index.html, where X is your site number. The main page must contain the following contents:

4.1 Service directory (a table) listing the services and links to the test pages (TryIt pages) and services. The schema and a list of example services are shown in the table below, which is similar to the table in the requirement document, except that the TryIt pages must be linked into the page. [8]

This page is deployed at: http://webstrarX.fulton.asu.edu/index.html (change X to you site number)				
This project is developed by: Put your team name here.				
Provider name	Service name, with input and output types	TryIt link	Service description	Actual resources used to implement the service
Member 1 name	Encryption and decryption: Input: String Output: String	TryIt	Cipher encryption and decryption	Use library class and local component to implement the service
Member 1 name	SolarPower Inputs: zip code and size Output: integer	TryIt	Output annual KW number for a given panel size at a given zip code location	Retrieve information from national database at: http://graphical.weather.gov/xml/
Member 1 name	findStore Input: zipcode Output: list of string	TryIt	Use an existing online service or API to find the locations of a given store name	Use the service from Yelp site at: http://www.yelp.com/
Member 2 name	…		…	…
Member 2 name	textToPhone inputs: string Account, string Password, string Receiver, string Subject, string Body Output: boolean	TryIt	Send a text to a cell phone and return if the message is sent successfully	Use google gmail account and carrier services: phone_number@tmomail.net phone_number@messaging.sprintpcs.com phone_number@vtext.com phone_number@txt.att.net
…	…		…	…

Make sure that you have the deployment URL in the first row of the table!

4.3 Deploy the table in html into WebStrar server, in the root directory outside the pre-created directories, so that the page can be accessed using the address: [7 points]

http://webstrarX.fulton.asu.edu/index.html (change X to you site number)

This page must also be deployed into Blackboard by one of the team members.

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: For each part of the submission,

- 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!