CSCE 742 - Assignment 2

Due Date: Thursday, October 25th, 11:59 PM (in PDF format, via Dropbox)

Question 1 (20 Points)

REST consists of several important elements bundled together to create an architectural style. One of them is client-server, so the communication paradigm is always: client issues requests to server, and server responds synchronously; the server does not initiate requests.

1. What limitations (if any) does this place on the kinds of applications that can be created? How does this choice affect performance?
2. Suppose servers were allowed to issue requests to clients along an existing HTTP connection. Describe a few benefits and a drawbacks of this not-quite-REST protocol.

Question 2 (20 Points)

Some languages and systems use *implied* interfaces rather than *explicit* interfaces.

For example: *http://www.shindich.com/sources/patterns/implied.html*, *https://en.wikipedia.org/wiki/Duck_typing*

1. Explain the difference between implied vs. explicit interfaces.
2. What are some advantages of implicit interfaces for architectural design? What are some disadvantages?

Do not simply quote from the example articles. Provide your own answer. It is recommended that you read additional sources.

Question 3 (22 Points)

Object-oriented design is often used to design the low-level classes of a system. Similarly, component-oriented design is often used to design large-scale architectures, and is particularly relevant to the functional view.

For example: https://en.wikipedia.org/wiki/Component-based_software_engineering, https://www.oreilly.com/library/view/programming-net-components/0596102070/ch01s02.html

1. What's the difference? Describe three ways in which components differ from objects.
2. Describe why components might be better suited to constructing large-scale systems than classes/objects.

Do not simply quote from the example articles. Provide your own answer. It is recommended that you read additional sources.

Question 4 (20 Points)

We have considered *data-oriented* and *computation-oriented* interfaces as two mechanisms by which one can conceptually communicate between components.

1. Define each type. What is the difference?
2. Provide and justify an example architectural style using each interface type.
3. Show how it is possible to construct a computation-oriented interface from a data-oriented one, and vice versa.
4. Since each mechanism can be expressed in terms of the other, comment on why we conceptually might like to keep them separate.

Question 5 (18 Points)

Consider a widely used Web site, such as Amazon or eBay. Identify one performance, one availability, and one usability scenario that you think would be necessary for that site.