**SOA Module Assignment**

**Introduction**

The assignment is to design a Service Oriented Architecture and approach for a given domain. You must show a good understanding of Service Oriented principles. In addition you must show knowledge and understanding of specific SOA techniques, practices and approaches in the design.

**Assessment objectives**

This assignment is being assessed. Like other modules, you will pass or fail dependent on demonstrating certain things. In this case the main criteria for passing is that you understand and can apply SOA concepts, principles and approaches for reasonably complex systems. This means that you must address key issues such as governance, security, description and discovery in your assignment. Your assignment must also show good service decomposition and a good understanding of why to use services, where to use services, and what makes a good service.

**Domain – Patient Records**

The UK has had a failed top down attempt to create a single IT system that would allow any hospital or medical practice to access any patient’s medical record, securely and reliably[[1]](#footnote-1). Recent news shows that the government is changing the architecture to utilize a lot of open source tools instead of proprietary models[[2]](#footnote-2).

In this assignment, we are going to explore a completely different alternative: one in which each patient has complete responsibility for their own data. Patients can either (if they are technically savvy) run their own medical record service, or they can choose a provider.

If they choose a provider, the provider must ensure that the records are secure and meet the relevant privacy, security and protection rules. Patients must be able to delegate rights to various healthcare providers – for example, to allow their doctors to read and update their medical record. They might allow a hospital instant access to read their records, with a limited time. They should be able to put time limits on access. You might want to think about how this gets delegated – for example you may want a specific doctor or department in the hospital to see your data but not another.

Like mobile phone number portability, each provider must provide a facility to securely transfer their medical record to another provider and to ensure that all data is deleted after successful transfer.

This is a large domain problem with multiple solutions, so it is up to you to choose and approach and justify it. Issues of security, identity and reputation are key in this model, and open APIs spring to mind as being an essential aspect.

You are not expected to provide a complete solution, and you are not expected to solve all the security and privacy issues of this problem, but you are expected to think significantly about these, given the privacy issues inherent in this problem.

**Tasks**

1. Schemas and message formats  
     
   A good Service Oriented Architecture has a clear set of message formats. For this assignment, document at least some of the relevant message formats and/or media types that your solution will use. These may be defined in XML Schema, as an XML infoset definition, UML, informal XML or JSON schema (as used in many standards) or using other schema approaches (such as JSON Schema). Discuss any decisions you had to make (including your choice of description language) and why you chose your approach, as well as outlining what other formats you might define.
2. Services  
     
   Create some well-documented clear service definitions. These may use WSDL, WADL, Swagger or another approach, but must clearly convey how to use the service such that a third-party with domain knowledge should be able to construct an application that interacts with that service. You do not need to create descriptions for all the services, but you do need to create at least one complete service description. Ensure that there is a human readable presentation (e.g. a graphical view, or clear document) as well as the XML.
3. Service interactions  
     
   Capture some service interaction flows. You may use BPEL, BPMN, EIP, ESB flows or sequence diagrams to indicate how different services interact and if there are workflows, ESB flows, etc., how those orchestrate and mediate the services.
4. Security and other non-functional aspects  
     
   How is the system secured? Which services need security and what is the most appropriate technology for that? What directories, token servers, key management, etc does the solution require? What other non-functional aspects are important? Does the solution use policies and if so what is described in those policies?
5. SOA design and concepts   
     
   Did you choose a WS-\* or a RESTful approach, or a mix of the two? Did you utilize asynchronous flow or event distribution? Is there an ESB, API management system, a registry, or a business process manager in your solution? How did you decide the granularity of your services? Are there components that are not available as services and how did you decide which components to expose and which to hide? What other SOA concepts such as compensation did you apply?

**Deliverables**

* Write a document of between 3000 and 5000 words, plus diagrams and code samples, outlining your design and architecture. Do not submit a document longer than 5000 words.
* You are not expected to completely implement a system! A real-life solution is out-of-scope.
* Clearly document any assumptions you make.
* You do not need to explore every aspect or every service. However, some well-chosen examples, explored to a good level of detail will demonstrate the understanding you have.
* You (and the examiner) must be confident that the there are no major flaws in the design and that it is implementable.
* Prototyping some parts of the system in a common toolkit such as JAX-RS, JAX-WS or .NET is highly recommended.
* The examiner appreciates references.

**Assessment Criteria**

Assessment will be according to the following criteria:

* Have you understood the principles and design characteristics of a service-oriented architecture? By undertaking a design activity, can you show that you appreciate the strengths and weaknesses of the approach?
* Can you implement and deploy simple services using a development platform?
* Are you able to define and design applications as combinations of services, and be able to discuss the emergent properties of those composite services?
* Have you addressed the discovery, description and governance issues of the system you have designed?
* Have you addressed security challenges around your solution?
* Do you understand the challenges, emerging work and tradeoffs between different approaches? In particular, can you articulate clearly why different SOA technologies are better or worse for certain tasks?

*Paul Fremantle, November 2013*

*paul@fremantle.org*

1. NHS Spine [↑](#footnote-ref-1)
2. <http://www.theregister.co.uk/2013/10/10/nhs_drops_oracle_for_riak/> [↑](#footnote-ref-2)