CS 3354 Homework 6

Daniel Crawford

1. A fat-client model would fit this problem better than thin-client. With the fat-client model, each of the users can do some application processing on their own machines instead of entirely on the server. For this situation, each dealer wants to use this simulation in a different way. For comparison, lets say a thin-client model is used for this application. If a dealer wanted to use the system in a specific way such as looking at different kinds of stock. Then the server would need to support several features which will require more processing. On the other hand, if the users have their own system to follow some specific instructions, then the service will benefit from it in terms of performance and convenience. In this purpose, the data just needs to be sent to the user and the user can compute what exactly they’re looking for on their own machine. For this situation, it is apparent that the fat-client architecture model is simply the best fit. However, there does come a major concern with this implementation. Usually when a fat-client model used, it is because the capabilities of the user’s system is known. Because the users can be random, the fat-client model may demand more processing than a user’s computer is capable of doing in a timely manner. Besides this glaring weakness, the fat-client model is still the best choice for this program.
2. 1. By design, software as a service is built to give each user a customized experience that makes each persons profile feel like their own. Typically, when people ask for IT support, it is because the customer misunderstands the product and needs the support to teach them how to use it. As a side effect of SaaS, IT support is needed less because the user interface can be less confusing. The IT support should really just be there for serious issues within the system that the user cannot fix on their own. So, to put it as simply as possible, each user is going to have their own personal profile. This helps fit the personal requirements of each user and/or organization.
   2. The costs of software as a service narrows down to making the application configurable and scalability. While there are benefits to making the software configurable, it makes the whole process much more expensive. Now, there has to be multiple features implemented into the program or system so that it feels like it can fit different peoples requirements. Scalability is another difficulty that affects expenses. Developers need to focus on a way to develop the system so that as more users come in, they can make changes quickly to get the system working properly. This is especially important if there is a sudden influx of people coming in to use the system. Thus, primarily, configuration and scalability are the two most prevalent factors related to cost in a SaaS.
3. 1. Software oriented architectures would not fit this system well simply because it does not have a guaranteed network connection. One of the fundamentals of SOAs is that it connects to one service that connects to other services as a package for a system. By this logic, a system simply must have network connection to work inside of this architecture. The embedded system property of this idea is not the issue here, because embedded systems can benefit from a SOA. The lack of a network connection simply makes the embedded system useless in this scenario.
   2. The concern with using an SOA is its promise of giving strict response times and accurate results. In an SOA, the same service may not be used every time by the application to get its output, which basically says that the timing of the results given can be different. In a system where the timing is crucial, output needs to be given to the user within a quick time interval, and no SOA service can guarantee this because it relies on so many network services which can cause problems. Following this reasoning that there are different services being used, the results are not guaranteed to be the same from every point. In a real-time system like airline reservation systems, this is a huge concern because it means that it could let two people reserve the same seat at once because two different services have yet to decide that the seat has been taken at the same time. So to reiterate, the structure of an SOA is not built for time-crucial systems because its using multiple services which can all function differently and respond slower.