**Faye Van Roekel  
Evolutionary Architecture at Amazon  
Web-430 – January 24, 2020**

Amazon’s architecture transformation is very well known and has been the most studied by others. Amazon started in 1996 on a monolithic application which ran on a web server which communicated with a database behind it. The application’s name was Obidos. The application contained business and display logic and all of the system’s functionality.

One of the problems that Obidos faced was it was very hard to scale up individual pieces as Amazon grew. Since the application was becoming too complex, it was becoming useless. Amazon realized that in order to be viable in the future and to keep up they needed to make some significant changes. After some self-analysis at the company, amazon decided that they needed to have a service-oriented architecture. By having this it would give them the level of isolation they needed to allow numerous software components to be built rapidly and independently.

Service-oriented architecture (SOA) is a style of software design where services are provided to the other components by application components. This happens through a communication protocol which goes over a network. Some of the basic principles of service-oriented architecture are being independent of vendors, products and technologies. If Amazon was independent of these it would allow components to be designed and services could be reused more effectively. By doing this it would reduce development time and also reduce some costs.

By having SOA services it allows for testing, debugging and maintence. The SOA services are self-contained and complete programs. There are several advantages to SOA. Elimination of complexities associated with interacting services within an environment is one of the advantages. By updating, upgrading, and maintaining the services in the SOA environment there are no complications resulting from interactions with other connected and interacting services. In environments with complex components, this makes things much easier.

Changing Amazon architecturally took around five years (2001-2005). This decision took Amazon from a two-tier monolith to a fully distributed, decentralized, services platform serving many different applications. Since Amazon was the first to take up this type of a challenge, there were three major lessons that were learned at Amazon during this change in their architecture.

1. Isolation is achieved when there is a strict service orientation which in turn causes ownership and control. Services are able to be structured the way they are needed.
2. In order to help security in the application and to increase speed for the user, database access by clients needs to be prohibited. This makes performing scaling and reliability improvements to your service state possible without involving your clients.
3. In order to create a strong customer focus, switching to service orientation in the development and operational processes allows for faster innovation. Each service is assigned to a team who then is responsible for that particular service. The team is responsible for everything from scoping out the functionality to architecting, building, and operating it.

The decision that Amazon made to make these massive changes was a great direction for them to go and it defined the path their application would take in the future and made them viable. Service-oriented architecture is a very powerful way do design applications.

**References:**

Kim, G., Debois, P., Willis, J., Humble, J., & Allspaw, J. (2017). *The DevOps handbook: how to create world-class agility, reliability, and security in technology organizations*. Portland, OR: IT Revolution Press, LLC.

Denning, S. (2019, July 11). How Amazon Became Agile. Retrieved January 24, 2020, from https://www.forbes.com/sites/stevedenning/2019/06/02/how-amazon-became-agile/#1563553f31aa

SarthakGargCheck. (2018, August 2). Service-Oriented Architecture. Retrieved January 24, 2020, from https://www.geeksforgeeks.org/service-oriented-architecture/