**Decisiones de arquitectura.**

Nuestra aplicación en un principio fue hecha sobre una arquitectura monolítica específicamente en JAX-RS. Luego de presentar una serie de problemas y de identificar necesidades que se ajustaban de mejor manera a otro tipo de arquitectura, decidimos migrar la aplicación por completo a Play! Además de esto, Play! Puede ayudar de manera gradual y mayor, a características en la implementación de funcionalidades que de alguna manera ayuden al cliente directo de la aplicación. Por otra parte Play! Puede contribuir de manera exponencial en la medida de obtención de mejoras en atributos y escenarios de calidad tales como la Disponibilidad y el Desempeño.

Para la primer entrega nuestra base de datos no corría junto con el servidor de la aplicación, esto aliviaba completamente la competencia por los recursos y ayudaba de manera efectiva a cada uno de los procesos. Al cambiar la arquitectura y manejar la base de datos se esta manera pudimos evidenciar una notable mejora, en ese momento obtuvimos mejores resultados en cuanto a tiempos de respuesta y latencia.

Por otra parte, una de las tácticas implementadas fue la táctica de introducción de concurrencia. Esta táctica fue implementada con el fin de que se procesaran una mayor cantidad de peticiones mediante la creación de hilos. Con esto, el tiempo de bloqueo por petición se puede reducir, si las mismas se procesan en paralelo.

Para esta segunda entrega, debido a la cantidad de requerimientos especificados tantos por los clientes directos de la aplicación, así como por la empresa y sus funcionalidades básicas, tuvimos que encontrar una mejora sustancial en la medida de obtención de respuestas y disponibilidad para el cliente. De esta manera, implementamos una herramienta satisfactoria como un “Balanceador de carga”. Este mismo se encarga directamente de un manejo efectivo de los procesos y funcionalidades de la aplicación, direccionándosela a diferentes tipos de recursos.