

Centro de Enseñanza Técnica Industrial

Plantel Colomos

Ingeniería en Desarrollo de Software

Nombre Alumno: José Rafael Ruiz Gudiño

Registro: 20110374

Arquitectura de Software

Actividad 4 - Antipatrones

5°P

T/M

06/03/2022

Architecture by implication

El error es tener desarrollado un sistema sin una documentación de su arquitectura. La clave de este punto es que hay que favorecer el desarrollo sobre la documentación la cual debe ser sencilla y asequible para todas las partes implicadas.

Información crucial que se debería tener

Integraciones con otros componentes

Requisitos de rendimiento y mantenibilidad

¿Es nuestra solución factible dentro de nuestros límites de

tiempo y presupuesto?

Requisitos de seguridad

Previsiones de escalabilidad

Witches brew

Consiste en juntar a tantísimos roles (stakeholders) en la definición de la arquitectura, que el resultado final es una mezcla de ideas sin mucho sentido ni visión clara, lo que bloquea el desarrollo de forma inevitable.

¿Cómo remediar esto?

Se necesita, en primer lugar, una clara estructura de equipo, donde los ingenieros puedan tomar la última palabra.

Además, las primeras fases de desarrollo deberán centrarse en el descubrimiento técnico, así como una revisión conjunta de dichas decisiones.

Gold plating

Se da tanto en fases de arquitectura como en fases de desarrollo, en el cual al realizar tareas llega un momento en que añadir mayor esfuerzo no genera una cantidad de valor razonablemente acorde con el esfuerzo realizado.

Desde la arquitectura, esto puede ocurrir si se comienzan a añadir demasiados detalles a los documentos de definición. Esto puede generar un estado de parálisis del proyecto.

Tenemos un producto dependiendo de otro que condiciona enormemente nuestro diseño.

La solución a este problema es utilizar patrones de diseño como Adapter, para desacoplar al máximo nuestra aplicación del producto externo.

Otra solución puede ser el uso de colas de mensajes como punto de conexión entre ambos sistemas.

Consiste en definir todos los detalles de la arquitectura al principio del proyecto (Big Design Up Front), que es el momento en que menos información tenemos.

Big Bang Architecture

Vendor King

Lo correcto es definir únicamente la arquitectura básica adecuada para arrancar el desarrollo, e ir refinando sucesivamente.

Para que nuestro proyecto concluya de forma exitosa será necesario que diseñemos una arquitectura flexible y adaptable al cambio, así como respetar al máximo buenas prácticas de implementación.

Armchair Architecture

Esto ocurre cuando bocetos dibujados en fases iniciales pasan al equipo de desarrollo para ser implementados sin validación ninguna.

Para solventar el problema de las fotos está claro que es necesario establecer un proceso claro de normalización de estos bocetos.

Los equipos deben estar implicados de principio

a fin en todas las fases de desarrollo.

Antipatrones

Incorporar nuevas tecnologías a nuestro sistema sin verificar la conveniencia de su uso.

Playing With New Toys

Antes de añadir cualquier tecnología, deberíamos responder las siguientes preguntas:

Spider Web Architecture

Es una arquitectura plagada de servicios distribuidos, donde no nos hemos parado a pensar en todos los problemas que esto va a generar, empezando por el mantenimiento de cada uno de ellos por separado.

Los microservicios están bien, pero pensar bien si es lo que necesitas cuando vayas por esa vía.

Infinity Architecture

Consiste en crear arquitecturas extremadamente flexibles, para soportar requerimientos que seguramente tendremos en el futuro. El problema es que muchísimas veces estos requerimientos no llegan a materializarse, y nos vemos manejando unos niveles de abstracción que complican cualquier mínimo cambio o la solución de bugs.

Groundhog Day

Ocurre cuando las decisiones de arquitectura tomadas no son comunicadas de forma efectiva, lo que lleva a discutirlas de nuevo en futuras reuniones o en nuestras mesas de trabajo.

¿Cómo evitarlo?

- ¿Qué valor añade?
- ¿Es una tecnología probada y contrastada para el problema que tratamos de resolver?
- ¿Somos de los primeros en utilizar esta tecnología?
- ¿Existe soporte suficiente en la comunidad?
- ¿Existe ya algo en nuestro sistema que proporciona esta funcionalidad o parecida?
- ¿Tenemos las habilidades necesarias en nuestro equipo, o deberemos considerar un tiempo adicional de aprendizaje?

Debemos centralizar todas estas decisiones en una Wiki o similar, de manera que ante la duda podamos remitir a nuestro interlocutor a ese lugar para finiquitar cualquier discusión innecesaria.

Antipatrones

Conclusiones

Los antipatrones son todas aquellas prácticas que, como desarrollador, se deben tratar de evitar, que, aunque al principio pueden sonar como buenas prácticas de implementar en nuestro proyecto en realidad lo que se ocasiona es que los procesos se retrasan, y en momentos se pueden llegar a tener parones, por mala implementación o redefinición de ciertos puntos. Por esto mismo siempre es recomendable siempre llevar buenas practicas de desarrollo y arquitectura.