ARQUITECTURA DE DIAGRAMAS

Álvarez Menéndez, David UO288705@uniovi.es

Fernández Vior, Ángel David UO287998@univovi.es

Jirout Cid, Alfredo UO288443@uniovi.es

Introducción

Los diagramas son una parte imprescindible para realzar un proyecto, siempre será necesario mirarlos y son la base del proyecto. Son un soporte para los desarrolladores, están hechos para consultarlos en cualquier momento y sobre lo que estamos hacerlo, por eso deben de siempre legibles por toda la gente del proyecto.

A lo largo de los años, algunas personas les han cogido algo de “manía” dado a que se hay gente que siente que es algo antiguo hoy en día.

Distinción de tipos de diagramas

Podemos distinguir dos tipos de diagramas principalmente, tenemos los diagramas de corta duración, que están hechos en reuniones cortas, donde lo que queremos es reflejar las ideas que tenemos y transmitirlas al resto de los participantes sin entrar en muchos detalles y en un formato excesivamente ornamentado, por otro lado, tenemos los diagramas de larga duración, estos son más formales, mejor estructurados y se incluyen en el código fuente de nuestro proyecto, nos llevaran más tiempo crearlos

Notación y tipo de diagrama

Lo más importante cuando se va a realizar un diagrama en arquitectura del software no es ni la notición, ni el tipo que se utiliza; lo realmente importante es que este transmita la idea correcta del proyecto y tenga sentido.

Respecto a UML, hay 15 tipos de diagramas diferentes, aunque en realidad para Ashley Peacock solo son importantes 2 o 3 de ellos. Destaca:

-Diagramas de clases: resulta fácil para representar la estructura del código y las interacciones entre clases

-Diagrama de secuencia: sus favoritos, permite entender como interactúa el usuario con el sistema.

Mermaid

Es una herramienta muy útil, ya que, permite utilizar “actores” para representar usuarios, el navegador o servicios involucrados con la aplicación. Además, permite añadir números a las interacciones, facilitando así el entendimiento de la secuencia.

Por último, Mermaid ofrece diagramas de flujo que son versátiles fuera del contexto UML, como en el modelo C4.

Miro

Esta herramienta es muy útil para sesiones más distendidas como brainstorming o primeras reuniones de colaboración. Consiste en una pizarra en línea que facilita la visualización y edición en tiempo real. Antes se solía hacer en pizarras, pero era un medio en el que se solía perder información.

Presente y Pasado

Cuando se empezó a hacer los diagramas se hacían en papel, lo cual llevaba mucho tiempo y en caso de una modificación, había que rehacer el diagrama entero, perdiendo mucho tiempo por el camino, hoy en dia, contamos con multitud de herramientas que permiten generar diagramas a través de código, algunas de estas son: Microsoft Visio, Draw.io Mermaid

Modelo C4

Es una forma de modelar la arquitectura creado por Simon Brown la cual consiste en cuatro diagramas:

- Diagrama de contexto del sistema
- Vista del contenedor
- Vista de componentes
- Vista de componentes más relevantes

Conclusión de Ashley Peacock

Los 2 modelos claves son: modelo de dominio y Modelo C4.

Es muy importante documentar bien el código, aunque hay que tener cuidado ya que es algo cambiante.

Arquitectura de diagramas software

Dada la importancia de los diagramas en el sector del software, estos son algunos de los aspectos clave para garantizar la calidad de estos según el experimentado arquitecto de software Mark Richards:

- A pesar de que pueda parecer obvio, un buen diagrama precisa de un título de la arquitectura que trata de representar para ofrecer un contexto inicial a quien vaya a utilizarlo.
- Los nombres de los diferentes componentes deben ser suficientemente representativos como para que una persona ajena al proyecto pueda comprender su propósito
- Evitar las relaciones mediante flechas bidireccionales. En su lugar utilizar dos flechas unidireccionales de manera que se pueda indicar que tipo de mensaje se emite y recibe entre dos entidades
- Siguiendo el punto anterior, las relaciones deben tener un nombre en base al tipo de comunicación entre los componentes
- La orientación es importante en un diagrama software ya que puede ser útil para hacer énfasis en una parte concreta del sistema que se considere importante. En ocasiones reorganizar un diagrama puede hacerlo más comprensible
- Las diferentes formas y colores deben ser coherentes entre sí. Además, se deben tener presentes las posibles discapacidades visuales de la persona que vaya a interpretar el diagrama
- Finalmente, una leyenda es muy conveniente para establecer un significado concreto para cada símbolo utilizado en el diagrama, facilitando así la comprensión de este y evitando ambigüedades.

Referencias

- SE Radio 566: Ashley Peacock on Diagramming in Software Engineering
- https://youtu.be/wgpSdpny-0c?si=zQAGovar4EhCYASO
- About Mermaid | Mermaid
- Modelo C4 - Wikipedia, la enciclopedia libre