Universidad Rey Juan Carlos – Diseño y Arquitectura Software (DAS)

Práctica1: Captura y Representación de Decisiones de Diseño

Curso 2023-2024

1. **Objetivo**: Dada una especificación de un problema se precisa identificar, capturar y documentar las decisiones de diseño y construir la arquitectura software de un sistema.
2. **Roles**: Se formarán grupos de **6 personas** (*en caso de grupos con 5 miembros habrá un ASJ menos*) con los siguientes roles:
    a. **Arquitectos Software Senior (ASS)**: Dos integrantes que tengan más experiencia que el resto del grupo actuaran como las personas que toman las decisiones de diseño y las capturan.
    b. **Arquitectos Software Junior (ASJ)**: Dos integrantes con experiencia en UML u otra notación similar se encargan de modelar y documentar la arquitectura software en base a las decisiones de diseño tomadas. En aquellos grupos de 5 miembros habrá un solo ASJ.
    c. **Arquitectos Software Cognitivos (ASC)**: Dos integrantes se encargan de cuestionar las decisiones tomadas y plantear nuevos problemas de diseño.
3. **Grupos de TEST y de CONTROL**: Los alumnos de grupos de TEST y CONTROL realizarán las mismas tareas, pero los grupos de TEST deben grabar las discusiones entre los roles ASS-ASC y entre los ASS-ASJ en ficheros de voz de una dirección máxima entre 2 y 8 minutos. Al menos deberá haber 1 fichero de voz por iteración.
4. **Herramientas sugeridas***:* Cualquier herramienta de modelado UML (se recomienda MagicDraw).

    Captura de decisiones de diseño: Las decisiones se capturarán utilizan el formato MADR para capturar las decisiones de diseño. **El formato MADR se encuentra en** https://github.com/adr/madr. Documentación adicional puede encontrarse en: https://adr.github.io/madr/ así como de la herramienta ADR-Manager https://adr.github.io/adr-manager/#/ . Se deberá crear una cuenta en GitHub con acceso libre para el profesor en el que se almacenarán las decisiones de diseño y las arquitecturas. Las decisiones capturadas y las arquitecturas, así como ficheros de voz con las discusiones se almacenarán cada semana en GitHub.
5. **Tareas**: Las tareas a realizar se desarrollarán durante **4 semanas**. **Fecha de comienzo: 12/11 Fecha**
    a. **Tarea 1: Semana 1 (Iteración 1): Análisis de requisitos y estilo arquitectónico**: Todos los miembros del grupo deberán comprender el problema y realizar una fase de análisis y captura de requisitos funcionales y de dominio. Cada requisito deberá llevar un identificador (RF1, RF1.1., RF2), un nombre descriptivo y una suficientemente descripción detallada. No consideraremos requisitos no funcionales o de calidad. Además, se deben capturar las primeras decisiones que den lugar al estilo arquitectónico. Solo
    b. **Tarea 2: Semanas 2, 3 y 4 (Iteraciones 2,3,4):** Se continúa refinando la arquitectura con clases y paquetes UML hasta terminar los requisitos.

c. **Operativa interna:** Para todas las iteraciones las tareas de cada rol son las siguientes:
    i. Los arquitectos senior (ASS) comienzan a identificar los problemas de diseño y plantear las primeras decisiones de diseño tratando de seleccionar el estilo o estilos arquitectónicos más adecuados. Las decisiones se capturan con MADR y se deberá medir el tiempo (**Time in ADD**) en tomar dichas decisiones desde que se comienza a pensar en el problema de diseño hasta que se captura una decisión. Para facilitar la captura de tiempo usar unidades que terminen en 0 o en 5.
    ii. Los arquitectos cognitivos (ASC) se reúnen con los seniors para discutir y cuestionar dichas decisiones. Asimismo, pueden plantear nuevos problemas de diseño. Se mide el tiempo (**Reflection time**) empleado en el proceso de reflectividad.
    iii. Los ASS reflexionan sobre las cuestiones planteadas y elaboran de nuevo las decisiones modificadas o nuevas y se mide de nuevo el tiempo (**Time in refine ADD**).
    iv. Los arquitectos junior (ASJ) toman las decisiones capturadas y documentan la arquitectura en UML con las vistas que ellos consideren necesarias. Si no las entendieran pueden preguntar a loas ASS que las aclaren. Se mide el tiempo (**Design Time**) en modelar la primera versión de la arquitectura. Cada iteración de la arquitectura se almacenará en un fichero diferente.
    v. Rellenar la plantilla de tiempos por cada semana de trabajo.
    vi. Subir las decisiones, los ficheros de voz (grupos de control) y las arquitecturas producidas al GitHub para cada iteración.

6. **Entrega de la práctica**: La entrega de la práctica se realizará mediante el campus virtual. El nombre de fichero deberá ser "**DAS-P1-GXX (siendo XX el número de grupo)**". **Cualquier fichero que no siga esta nomenclatura será rechazado**. **La fecha límite de entrega de la memoria es el 19/11. El fichero comprimido contendrá las siguientes partes:**
    a. **Un fichero en texto plano con el enlace al repositorio GitHub que contengan las decisiones, arquitecturas y ficheros de voz (grupos TEST) organizados en carpetas**
    b. **Una memoria en Word o PDF no superior a 25 páginas con la siguiente información:**
        i. **Portada con nombre de la práctica, asignatura, lista de miembros del grupo, email del responsable. Índice generado de forma automática y numerado. Las figuras y tablas deben numerarse y llevar un pie de figura y tabla apropiado.**
        ii. **Nombres y rol de los integrantes**
        iii. **Descripción de los resultados para cada tarea incluyendo los requisitos funcionales y resultados intermedios. Las arquitecturas resultantes deben incluir un diagrama de clases y paquetes. Las decisiones**

principales deben explicarse e incluir la arquitectura resultante en cada iteración.

iv. **Conclusiones en base a lecciones aprendidas**

v. **Bibliografía adecuadamente formateada**

vi. **Anexo con todos los tiempos estimados**

Evaluación. La evaluación sobre 10 consistirá en: 90% el contenido, 10% calidad y presentación de la memoria. El contenido para los grupos de TEST se dividirá un 40% las arquitecturas producidas, un 40% las decisiones de diseño con los tiempos estimados y un 10% las discusiones en los ficheros de voz. Los grupos de CONTROL reparten los pesos en 40% y 50%. **Una práctica en la que los alumnos no hayan asumido los diferentes roles se considera suspensa**. El profesor podrá realizar preguntas a los alumnos de un grupo en cualquier momento lo cual puede influir en la nota individual de un alumno respecto de la nota de los restantes miembros del grupo. **Si algún grupo no sube cada semana con su fecha correspondiente las decisiones y las arquitecturas al GitHub la práctica se considera suspensa. Además, si la calidad de las decisiones y las arquitecturas es baja y no llega a unos mínimos, la práctica se considera suspensa.**

Plantilla de tiempos (en minutos)

Week	Iteration	Time in ADD (ASS)	Reflection Time (ASS-ASC)	Time in refined ADD (ASS)	Design Time (ASJ)
1	1	35	30	20	40
2	2				
3	3				
4	4				

Anexo I: Descripción del sistema que se pretende diseñar

Una compañía de productos alimenticios pretende migrar la arquitectura de un sistema monolítico a una basada en microservicios de manera. La arquitectura existente cuenta con una parte de clientes PC y móvil que acceden a los datos de la empresa alojados en 2 BBDD SQL (Clientes, Pedidos). La BBDD de Clientes contiene datos de clientes y pagos mientras que la BBDD de Pedidos se encarga de almacenar los datos restantes. El acceso actual se pretende que sea sustituido por protocolos HTTP/REST mediante un componente Gateway adecuado.

La lógica de negocio de la empresa cuenta con los siguientes módulos:

- **Clientes (crítico):** Este componente permite a acceder los datos de personales de los clientes.
- **Pedidos (no crítico):** Este módulo permite a los clientes realizar pedidos de los productos a la empresa. Si un cliente intenta realizar un pedido se le permite un número de intentos por determinar.
- **Reparto y rutas (Crítico):** Este componente complejo cuenta con una gran funcionalidad que es necesario desacoplar y gestiona el reparto de las flotas de transporte a los clientes y las rutas de los camiones. La gestión cuenta con 2 algoritmos de optimización que se seleccionan en función de la demora del camión.
- **Estadísticas (no crítico):** El sistema cuenta con un módulo de estadísticas que proporciona información valiosa sobre el estado de los pedidos y la situación en tiempo real de los camiones. Las estadísticas proporcionan también información de clientes.
- **Incidencias (semi crítico):** Este módulo reporta a los gestores de las rutas cualquier tipo de incidencia (camión averiado, reparto no realizado, etc.)
- **Pagos (crítico):** Este componente es una pasarela de pago externa que proporciona otra empresa para garantizar la seguridad de los pagos.

??? Funcional o no

La nueva arquitectura debe contar con los elementos software adecuados para ejecutar los microservicios. Por otra parte, en la arquitectura existe un componente denominado GestorPedidos que hace de intermediario entre los clientes, pedidos, el reparto y las incidencias comunicando dichas funcionalidades.

> **Nota**: La descripción del problema no tiene por qué estar completa. El alumno debe suponer requisitos funcionales y los datos necesarios para cada funcionalidad y ser responsable de buscar la información necesaria para tener la máxima información completa. Además, puede necesitarse algún componente adicional a los mencionados.