7 Análisis de Consistencia

7.1 Tabla Requisitos-Párrafo

Aquí describiremos los requisitos que no aparecen en el enunciado del ejercicio.

RF16 Permitir creación, modificación y eliminación de los partes de trabajo: en el enunciado del ejercicio los dos primeros puntos, creación y modificación, si aparecen, pero en lo relativo en la eliminación de los partes no se muestra explícitamente en el documento, pero presuponemos que es una función necesaria del sistema.

RF26 Mantener un listado de los empleados: no se muestra explícitamente en ninguna parte del enunciado, pero es un requisito fundamental para las peticiones que no son requeridas.

RF28 Dar la opción de aceptar y declinar presupuesto: en nuestro sistema para que una petición de trabajo sea aceptada, si el cliente ha pedido previamente un presupuesto, debe aceptarlo para que la petición de trabajo sea efectiva.

RF29 Permitir cancelar peticiones de trabajo: si el cliente tras hacer una petición de trabajo desea cancelarla en un momento no critico el sistema permitirá dicha cancelación. Esto no aparece en el enunciado.

RF32 y RF33 tienen las mismas funcionalidades que los requisitos RF28 y RF29, pero en este caso, los clientes tienen acceso al sistema directamente.

RF36 Consultas sobre los datos de los pedidos: los responsables de almacén deben acceder a los datos de los pedidos, pero esto no aparece como tal en el enunciado.

RNF1 y RNF2: estos requisitos se refieren a los datos obtenidos al realizar la entrevista con el cliente, por lo tanto, no aparecía en el enunciado.

RNF4: este requisito nos permite mayor eficiencia energética al sistema, al reducir las horas de funcionamiento de este, y nos aporta un plus de seguridad. Pero no aparece en el enunciado ni fue expuesto en la entrevista.

RNF5: el sistema nunca se actualiza en horas de trabajo para tener una disponibilidad plena, haciendo que los usuarios no realicen actualizaciones. Este requisito no aparece explícito en el enunciado ni se comentó en la entrevista.

RNF6: se envía a una persona para garantizar que cuando sea necesario realizar una actualización se haga en el menor tiempo posible.

RNF7: cuando el rendimiento del sistema no es el adecuado se procede a mejorar los equipos para obtener un mayor rendimiento del sistema y, por tanto, de la empresa.

RNF8: todas las aplicaciones del sistema se adaptarán al formato de los dispositivos sobre los que trabajen.

RNF9: muestra el contenido del sistema, adaptándose al hardware del equipo sobre el que se trabaja.

RNF10: la interfaz será intuitiva y similar tanto en la versión web como en la versión móvil.

RNF11: se dispone de un plan de adaptación para que los usuarios con mayor dificultad a la hora de utilizar el sistema puedan aprender rápida y eficazmente.

RNF12: se podrán incorporar asistentes en el sistema que permitan al usuario mayores recursos de búsqueda.

RNF13: la información del sistema se guarda en una base de datos para proteger el sistema ante un fallo en el sistema.

RNF14: si se produce algún fallo en el sistema siempre habrá personal disponible para subsanarlo.

RNF15: el software se prueba exhaustivamente para reducir al mínimo las posibilidades de error.

RNF16 y RNF17: las versiones del sistema con más funcionalidades permiten que el sistema sea más estable y fiable.

RNF18: en las versiones más baratas del sistema el sistema cuenta con menos portabilidad.

RNF19: en las versiones más caras del sistema el sistema cuenta con mayor portabilidad.

RNF20: en las versiones más baratas el uso de un framework multiplataforma proporciona una portabilidad excelente.

RNF21: la portabilidad de la base de datos es la máxima posible y cualquier equipo se puede conectar con ella mediante TCP.

RNF22: se reduce el número de funcionalidades del sistema en función del coste de este.

RNF23: se proporciona una versión con un mayor costo orientada a los **clientes** los cuales dispondrán una mayor cantidad de funcionalidades específicas.

RNF23: se proporciona una versión con un mayor costo orientada a los **técnicos** los cuales dispondrán de una mayor cantidad de funcionalidad específicas.

RFN26: el cliente java puede contactar directamente mediante TCP con la base de datos, aumentando así su interoperabilidad.

RFN27: en la versión más cara del sistema el servidor Apache se comunica con la base de datos mediante PHP, aumentando así su interoperabilidad.

RFN28: los dispositivos móviles se comunican con la base de datos mediante, aumentando así su interoperabilidad.

RFN29: se proporcionará una mayor estabilidad al sistema utilizando patrones de diseño a la hora de implementar el código.

RFN30: se proporcionará una mayor estabilidad además de mantenibilidad al sistema utilizando técnicas de Programación Orientada a Objetos.

RFN31: el sistema tendrá un margen de usuarios conectados para que en caso de sobrecarga el sistema no colapse. Esta información se obtuvo a través de la entrevista.

RFN32: se ofrecen planes de mantenimiento de distinto coste.

RFN33, RFN34, RFN35, RFN36, RFN37: todos son requisitos referentes a la seguridad del sistema, de los usuarios y de la forma de acceso de dichos usuarios al sistema.

Se han añadido requisitos relativos al lanzamiento de ofertas especiales, las cuales pueden aplicarse durante unas fechas concretas o a unos determinados clientes por motivos especiales. Estos requisitos son gestionados por parte del coordinador, quien los crea pero es el sistema el encargado de supervisar que se apliquen correctamente a las facturas implicadas.

TABLA EXCEL

7.2 CONSISTENCIA EN EL DIAGRAMA DE DISEÑO DE CLASES

Hemos creado diversos diagramas de clases, uno para cada una de las aplicaciones que existirán y se coordinarán en nuestro sistema. Dichos diagramas nacen inspirados por las entidades del modelo de dominio, del que deducimos las clases de trabajo principales. Pensando en cómo coordinaremos las distintas aplicaciones del sistema entre sí con la base de datos, llegamos a la conclusión de que íbamos a necesitar otras clases para nuestro diagrama.

7.2.1 CONSISTENCIA ENTRE DIAGRAMAS DE CLASE

A la hora de implementar dichos diagramas de clases, la forma más sencilla ha sido partir de un diagrama de clases general en el que se encontraban la mayoría de clases utilizadas en los distintos diagramas. La forma de crear dichos diagramas de clases ha sido partir del diagrama general, copiarlo en un nuevo modelo y a partir de ahí añadir las clases necesarias para dicho diagrama específico. De esta manera conseguimos actualizar las clases automáticamente en todos los diagramas cuando modificamos una clase en cualquiera de ellos, a la vez que tenemos los mismos valores de multiplicidad de las asociaciones entre las clases.

7.2.2 CONSISTENCIA ENTRE DIAGRAMAS DE CLASE Y DIAGRAMAS DE SECUENCIA

Para crear los distintos diagramas de secuencia tenemos que crear un nuevo diagrama de secuencia (una vez creado el diagrama se podrá vincular a su caso de uso correspondiente). Para añadir el actor simplemente arrastramos el actor desde el paquete de actores, para añadir la clase sistema (en nuestro caso interfaz de usuario) hacemos el mismo procedimiento, pero con el paquete de clases en vez de actores. Añadiremos los mensajes entre el actor y la clase mediante las herramientas de diagramas y seleccionaremos el método correspondiente con dicho mensaje. Seleccionamos message y lo arrastramos del actor a la clase y encima click derecho y en features en el apartado realization seleccionamos el método correspondiente.

Una vez hemos creado el diagrama de secuencia, procedemos a vincularlo con el caso de uso correspondiente. Para ello clicamos sobre el caso de uso, seleccionamos add new > diagrams> referenced sequence diagram y seleccionamos el diagrama de secuencia al caso de uso que le corresponde.

7.2.3 CONSISTENCIA ENTRE DIAGRAMAS DE CLASE Y DIAGRAMAS DE COLABORACIÓN

Para crear los distintos diagramas de colaboración, al igual que los diagramas de secuencia, tenemos que crear un nuevo diagrama de colaboración. Para añadir las clases simplemente tendremos que arrastrar dichas clases desde su paquete correspondiente al modelo sobre el que estemos trabajando y las relaciones entre las clases las añadiremos mediante las herramientas de diagramas. A la hora de insertar los métodos tendremos que usar las herramientas *link message* o *reverse link message*. Para elegir el método que queremos usar, hacemos click sobre el mensaje y seleccionaremos el método correspondiente de la clase con la que es enlazada. De esta manera tendremos una correspondencia plena entre los diagramas de clases y los diagramas de colaboración.