Práctica 2:

Especificación de requisitos

Ingeniería del software

2º grado en ingeniería informática

Curso 2020/2021

Aurora Ramírez Quesada

Departamento de informática y análisis numérico

Índice

1. Organización de la práctica

Este documento contiene la información necesaria para realizar la segunda práctica de la asignatura. Esta práctica tiene una duración de tres semanas, en las cuales se realizará la especificación y análisis de requisitos sobre el problema propuesto. A continuación se detallan los objetivos y organización de la práctica:

Semana 1: extracción y análisis de requisitos

- Fecha: 05/06/08 (según grupo) de octubre de 2020.
- Objetivos:
  1. Aprender a extraer requisitos a partir de la entrevista con el cliente.
  2. Identificar, refinar y documentar los distintos tipos de requisitos del sistema.
- Preparación: se recomienda buscar información sobre el dominio de aplicación y traer planteadas posibles preguntas para la entrevista (ver los apartados "entrevista" y "descripción del problema" de la sección 2).
- Seguimiento y evaluación: se realizará una simulación de entrevista con el cliente, a partir de la cual los distintos equipos de trabajo deberán extraer y formular los requisitos. En esta semana no se realiza evaluación.

Semana 2: historias de usuario

- Fecha: 12/13/15 de octubre de 2020. Las clases del 12 de octubre se trasladan al 14 y 15 de octubre en horario de tarde (consultar calendario y aula en Moodle).
- Objetivos:
  1. Utilizar historias de usuario para crear la lista de producto según la metodología *Scrum*.
  2. Comenzar a planificar y priorizar funcionalidades de cara a su futura implementación.
- Preparación: los requisitos del sistema deben estar previamente especificados.
- Seguimiento y evaluación: cada equipo deberá crear la lista de producto y las tareas de planificación en su proyecto de *YouTrack*. En esta semana no se realiza evaluación.

Semana 3: casos de uso

- Fecha: 19/20/22 de octubre de 2020.
- Objetivos:
  1. Aplicar la técnica de casos de uso de UML para detallar el análisis de requisitos funcionales. Uso de *Visual Paradigm*.
  2. Validar los requisitos con el cliente para resolver dudas surgidas durante la toma de requisitos.
- Preparación: los requisitos del sistema deben estar previamente especificados.
- Seguimiento y evaluación: se trabajará en el aula para completar la especificación de requisitos y resolver dudas. Tras esta sesión, los estudiantes dispondrán de unos días adicionales para realizar la entrega de la práctica en Moodle. Dicha entrega deberá ser un documento formal con la especificación de requisitos y los diagramas de caso de uso (ver plantilla en Moodle). Las historias de usuario y la planificación de los *sprints* quedarán reflejadas en el proyecto de *YouTrack*.

Esta práctica está directamente relacionada con el contenido teórico de la asignatura, en concreto los temas 4 (análisis de requisitos) y 5 (técnicas de especificación y modelado). Por tanto, es deber del estudiante consultar y repasar dicho material durante la elaboración de la práctica. En el resto del documento se resumen solo aquellos aspectos esenciales para el desarrollo de la práctica y el uso de las herramientas.

2. Extracción de requisitos

La entrevista

La entrevista es una de las técnicas para realizar la extracción de requisitos [1]. En el ámbito del desarrollo de software, suelen realizarla los ingenieros de requisitos con el cliente, con el fin de conocer cómo funciona la organización cliente, el alcance del problema y las características del sistema software a desarrollar. Puede haber varias entrevistas a lo largo del proyecto, de forma que al principio se traten temas más generales con directivos o gerentes. Más adelante, pueden plantearse entrevistas con los propios usuarios para conocer su entorno de trabajo y las necesidades que debe cubrir el sistema.

Con las entrevistas se logra una mayor interacción con el cliente, haciéndole partícipe del proceso. Pueden ser más o menos informales en función de si existe una lista predefinida de preguntas que el cliente debe responder (entrevista cerrada) o si la entrevista

se desarrolla sin un guión preestablecido (entrevista abierta). En la práctica, es habitual combinar ambos tipos. Antes de la entrevista, el cliente puede enviar un resumen de temas a tratar y objetivos a alcanzar, así como cualquier otro documento que pueda ser útil para situar a los ingenieros de requisitos. Tras la entrevista, se puede elaborar un resumen para que ambas partes tengan constancia de la misma.

El éxito de la entrevista depende en gran medida de su preparación por parte del equipo del proyecto y sus habilidades interpersonales a la hora de comunicarse con el cliente. Los entrevistadores (ingenieros de requisitos) deben tener una mentalidad abierta, no tener ideas preconcebidas y no esperar que el cliente sea capaz de contestar de forma precisa qué es lo quiere. Los requisitos de dominio suelen ser los más complejos de obtener por medio de la entrevista, dado que los ingenieros no conocen los detalles del negocio y a los clientes les resulta complicado expresarlos claramente.

Para conducir la entrevista, los ingenieros pueden plantearse una serie de preguntas y establecer un orden para abordarlas. La tabla 1 presenta una posible plantilla para realizar la entrevista [2]. Dado que la entrevista a realizar en clase será de corta duración y que el sistema a desarrollar es de pequeña magnitud, muchas de estas preguntas no serán necesarias. No obstante, se recomienda a los equipos plantear una serie de preguntas concretas y establecer pautas para realizar la entrevista (orden de las preguntas, quién intervendrá en cada momento, etc.)

Tipos de requisitos

Durante la entrevista deberán identificarse los siguientes tipos de requisitos:

1. Requisitos funcionales. Conjunto de funcionalidades o servicios que el sistema debe ofrecer. Sirven para expresar ***qué*** debe hacer el sistema, es decir, cómo debe reaccionar ante determinadas entradas.
    - Requisitos de información. Detallan qué información necesita manejar el sistema y cómo se organiza.
2. Requisitos no funcionales. Permiten indicar restricciones al sistema que pueden afectar a la calidad de servicio (fiabilidad, tiempo de respuesta, etc). Por tanto, se centran más en expresar ***cómo*** debe comportarse el sistema. También pueden hacer referencia a consideraciones sobre el desarrollo del sistema (lenguaje de programación, interoperabilidad, dependencias externas).

Descripción del problema

Nuestro cliente es una empresa de gestión de parques naturales en Andalucía que necesita un sistema de gestión de la información de estos entornos. En concreto, la empresa cliente se encarga de registrar los espacios que gozan de este reconocimiento, garantizando su mantenimiento y conservación, así como programar rutas por sus senderos.

Información sobre el cliente	Nombre Compañía/departamento Cargo/rol
Identificación de problemas	¿Cuál es el problema? ¿Por qué es un problema? ¿Cómo se resuelve actualmente? ¿Cómo le gustaría que se resolviese en realidad?
Entorno de usuario	¿Quién utilizará el sistema? ¿Qué nivel de experiencia tienen los usuarios? ¿Qué plataformas tecnológicas utilizan actualmente? ¿Existen otros sistemas con los que se vaya a interactuar?
Identificar soluciones	Plantear soluciones alternativas Establecer prioridades
Identificar requisitos no funcionales	Conocer expectativas Preguntar sobre la instalación, configuración y mantenimiento Recopilar información sobre aspectos de regulación o legales que puedan influir
Resumen del problema	Describir las necesidades extraídas Describir los problemas de la solución actual, si la hay

Tabla 1: Plantilla para una entrevista

3. Historias de usuario

Introducción

En las metodologías ágiles, como *Scrum*, no existe una fase de análisis de requisitos como tal. El concepto más similar son las llamadas **historias de usuario** (*user stories*) [3]: descripciones en lenguaje no técnico que expresan un objetivo a cumplir desde el punto de vista del usuario del software. En *Scrum*, las historias de usuario se añaden a cada *sprint* para guiar la planificación del tiempo y contextualizar el trabajo que debe realizar el equipo de desarrollo. No se trata de definir la función a implementar, sino de expresar qué debe lograrse para solucionar un problema que plantea el usuario. Su descripción debe responder a la siguiente estructura:

Como [rol de usuario], quiero [objetivo], para [beneficio/valor]

- El **"como"** nos ayuda a pensar en el usuario concreto, sus capacidades y necesidades.

- El **"quiero"** sirve para expresar la intención que se persigue. No debe contener detalles de la implementación, ya que eso forma parte de la solución.
- El **"para"** nos permite visualizar el valor que esto aporta al usuario en el contexto general del proyecto.

Plantilla

Una historia de usuario suele escribirse en formato tarjeta y puede contener varios campos según las características del proyecto. Recordemos que en *YouTrack* disponemos de un panel *Scrum* para organizar las historias de usuario durante los *sprints*. La información que debe rellenarse en cada tarjeta será la siguiente:

1. Título: nombre por el que nos referiremos a esta historia de usuario. En *YouTrack* utilizaremos el campo "resumen".
2. Identificador: código único para identificar la historia de usuario. *YouTrack* lo asigna automáticamente.
3. Descripción: texto que explica la historia de usuario siguiendo la estructura explicada anteriormente. Para diferenciarlo de otro texto, se utilizará la negrita al escribirlo en el campo "descripción" de la tarjeta en *YouTrack*.
4. Responsable: persona encargada de completar esta historia de usuario.
5. Criterios de validación: aspectos adicionales que deban tenerse en cuenta y hayan sido consensuados con el cliente. Se detallarán como una lista en el campo "descripción" de la tarjeta.
6. Prioridad, estimación en días (*ideal days* en *YouTrack*) y *sprint*.

La figura 1 muestra cómo se rellenaría una historia de usuario en *YouTrack*. Las historias de usuario deberán ubicarse en el panel de desarrollo del proyecto y servirán para planificar los *sprints* durante la implementación.

4. Casos de uso

Para finalizar la práctica, se realizará la especificación de los requisitos funcionales mediante la técnica de casos de uso de UML [4]. A modo de resumen, se recuerdan los pasos fundamentales a seguir:

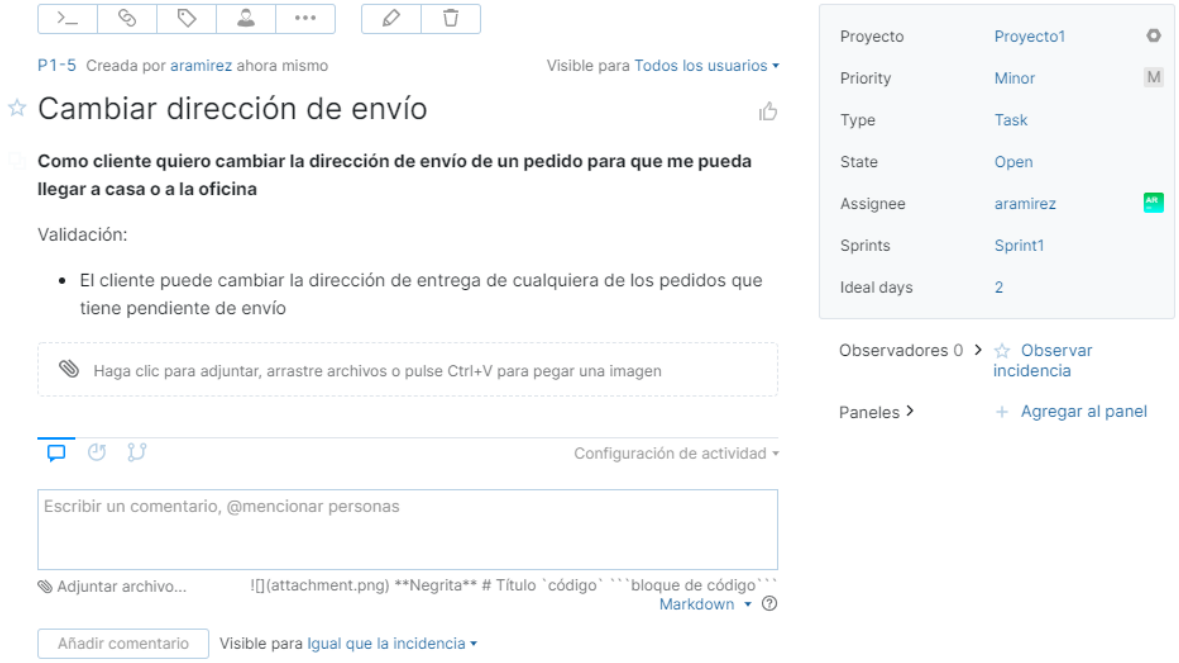

Figura 1: Tarjeta que describe una historia de usuario en *YouTrack*

1. Identificar a los actores. Durante la entrevista se habrán mencionado distintos tipos de usuarios, así como sus responsabilidades y permisos. Esta información es clave para decidir quiénes son los actores que interactuan con el sistema.

2. Identificar los principales casos. Para cada actor identificado, deben recopilarse las acciones de las que son responsables a un nivel de abstracción elevado.

3. Identificar nuevos casos de uso. Refinar la lista de casos de uso para reflejar el resto de funcionalidades, establecer relaciones, etc.

4. Describir los casos de uso. Crear una descripción detallada del caso de uso, incluyendo su escenario principal y los alternativos (extensiones).

Visual Paradigm

Para la elaboración del diagrama de casos de uso se utilizará la herramienta *Visual Paradigm CE (community edition)* [1]. El programa está disponible en las máquinas de la universidad, si bien se recomienda instalarlo para el seguimiento de las clases virtuales. Tras descargar y ejecutar el instalador correspondiente a nuestro sistema operativo, se nos pedirá registrar un correo electrónico para obtener un código de activación. También se proporciona acceso a tutoriales en línea.

[1] `https://www.visual-paradigm.com/download/community.jsp`

Una vez activado el código, podemos usar la herramienta para modelado UML (otras funciones no están disponibles en la edición CE). La figura 2 muestra un ejemplo de diagrama de casos de uso. Para crear este tipo de diagrama, deben seguirse los siguientes pasos:

1. Crear un nuevo proyecto, que nos servirá para almacenar todos los diagramas. En la pestaña *Project*, pulsar la opción *New*, escribir un nombre para el proyecto y comprobar que el conjunto de tipos de datos es *UML*. A continuación confirmar con la opción *Create blank project*.

2. En la zona de trabajo, seleccionar *System Modeling*. Ir a la columna *UML* y pulsar el botón +.

3. En el cuadro de diálogo, seleccionar el tipo de diagrama, *Use Case Diagram* en este caso. A continuación, seleccionar la opción *Blank* para crear un diagrama vacío. Introducir un nombre para el diagrama.

4. Tras estos pasos, se nos mostrará el editor de diagramas de casos de uso. En el panel lateral tenemos los diferentes elementos que pueden aparecer en este tipo de diagrama (actor, caso de uso, relaciones, etc.)

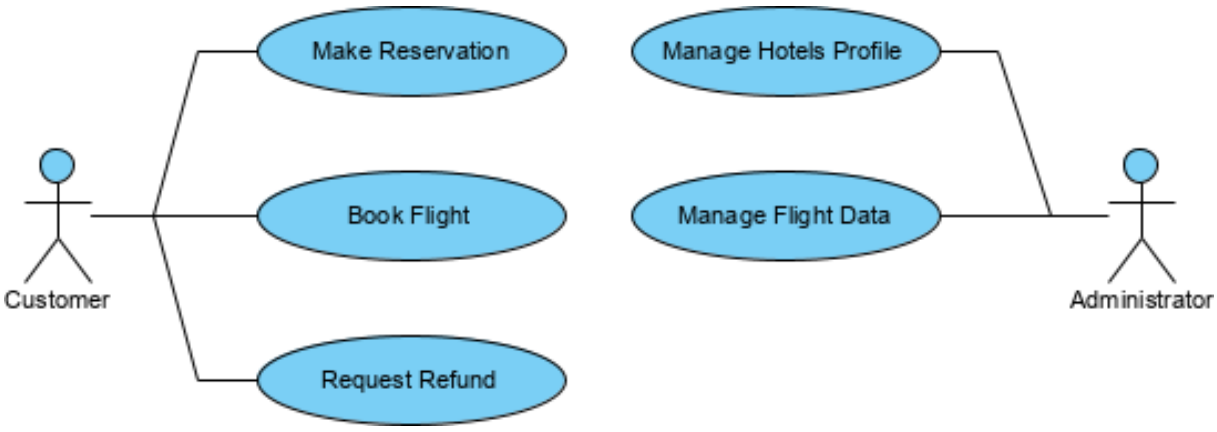

Figura 2: Ejemplo de diagrama de casos de uso en *Visual Paradigm* [5]

Para más información sobre la herramienta, se recomienda consultar la guía rápida [6] y el tutorial sobre casos de uso [5].

Plantilla

Para la descripción de los casos de uso se utilizará la plantilla mostrada en la tabla 2, que se corresponde con la estudiada en clase de teoría.

Caso de uso	Nombre del caso de uso
Objetivo	Frase corta que describe el caso de uso
Identificador	Código único que identifica al caso de uso
Contexto	Frase más larga para detallar condiciones, contexto, etc.
Actor principal	Actor involucrado en el caso de uso
Escenario principal	Pasos numerados que detallan el proceso esperado
Extensiones	Acciones alternativas a algunos de los pasos

Tabla 2: Plantilla para describir los casos de uso

Referencias

[1] I. Sommerville, *Software Engineering*. Pearson, 10th ed., 2016.

[2] ReQtest, *How to Use Interviews to Gather Requirements*, 2012. Disponible en: `https://reqtest.com/requirements-blog/how-to-use-interviews-to-gather-requirements/`.

[3] Atlassian, *User stories*. Disponible en: `https://www.atlassian.com/es/agile/project-management/user-stories`.

[4] J. Arlow and I. Neustadt, *UML 2*. Anaya, 2006.

[5] V. Paradigm, *Tutorial sobre casos de uso*, 2016. Disponible en : `https://www.visual-paradigm.com/tutorials/writingeffectiveusecase.jsp`.

[6] V. Paradigm, *Guía rápida*, 2017. Disponible en : `https://cdn-images.visual-paradigm.com/quickstart/quickstart.pdf`.