Universidad Abierta Interamericana

Campus Virtual UAIOnline

METODOLOGÍA DE DESARROLLO DE SISTEMAS I

UNIDAD 4

CASOS DE USO Y REQUISITOS

Autor de contenidos:
Carlos Neil

PRESENTACIÓN

En la Unidad 3 hemos visto las nociones generales del modelo estructurado. En la misma se presentaron elementos que intervienen en este modelado como así también las diagramas que la componen como son el diagrama de flujo de datos (DFD), el diccionario de datos(DD) y la especificación de procesos (EP) dejando para la unidad 6 el diagrama entidad relación (DER).

En esta unidad N°4, estudiaremos ***las herramientas actuales de modelado***. Estas herramientas se han impuesto sobre paradigmas anteriores como el estructurado que presentamos en la unidad anterior.

Para una mejor organización de los ritmos de estudio, hemos organizado esta unidad en tres partes:

- Unidad 4.1: Casos de Uso y Requisitos.
- Unidad 4.2: Introducción al Paradigma OO.
- Unidad 4.3: Análisis OO: Modelo de Dominio y DSS

Iniciamos esta unidad con el estudio de los Casos de uso y Requisitos.

1. INGENIERIA DE REQUISITOS

La Ingeniería de Requisitos direcciona el proceso de elicitación (inducir), definición, modelado, análisis, especificación y validación de los requisitos de un sistema y de su software, basado en un enfoque sistemático, separando el "qué" del "cómo" del diseño.

2. REQUISITOS

Los requisitos son los Flujo de trabajo fundamentales cuyo propósito esencial es orientar el desarrollo hacia el sistema correcto. El conjunto de todos los requisitos forma la base para el desarrollo del sistema o el componente de Sistema.
El conjunto de los requisitos es el acuerdo entre el cliente (usuario) y los desarrolladores del sistema acerca de lo que el sistema debe hacer y lo que no. De aquí la importancia de comprender en forma clara la necesidad del usuario.

3. REQUISITOS DE SOFTWARE

Un requisito de software es la capacidad que tendrá el sistema para resolver un problema o alcanzar un objetivo que servirá a los usuarios del mismo para la toma de decisiones.
Una capacidad del software que debe ser reunida o poseída por un sistema o componente del sistema para satisfacer un contrato, especificación, estándar u otra documentación formal. En

síntesis, un requisito de software es una descripción completa del comportamiento del sistema que se va a desarrollar y que le servirá de guía para los desarrolladores.

4. TÉCNICAS DE ELICITACIÓN

Para poder obtener los requisitos funcionales de cualquier sistema debemos acceder a personas o materiales que nos permitirán recopilar los mismos.
Existen técnicas para obtener estos requisitos como:

- Entrevista de comienzo y final abierto
- Entrevistas estructuradas
- Brainstorming (Lluvia de Ideas)
- Prototipos
- Escenarios
- Observación

5. TIPOS DE REQUISITOS

Existen varios tipos de requisitos como lo son:

- **Requisitos de Usuarios:**

  Necesidades que los usuarios expresan verbalmente.
- **Requisitos del Sistema:**

  Son los componentes que el sistema debe tener para realizar determinadas tareas.
- **Requisitos Funcionales:**

  Son los sservicios que el sistema debe proporcionar. Estos requisitos especifican una acción que debe ser capaz de realizar el sistema, sin considerar restricciones físicas. Describen la funcionalidad o los servicios que se espera proveerá el sistema.
- **Requisitos no funcionales:**

  Son restricciones que afectan al sistema. Especifican restricciones físicas sobre un requisito funcional (rendimiento, plataforma, fiabilidad). No se refieren directamente a las funciones específicas que entrega el sistema, sino a las propiedades emergentes de éste como la fiabilidad, la respuesta en el tiempo y la capacidad de almacenamiento. A Son restricciones a nivel hardware y Redes.

6. POSIBLES ACTORES EN LOS REQUISITOS DE INGENIERÍA

Usuario Final:

Son personas que utilizarán el sistema en desarrollo. Quienes utilicen las interfaces y manuales de usuario.

Usuario Líder:

Son las personas que comprenden el ambiente del sistema y el dominio del problema. Proporcionan información al equpo técnico (detalles de los requerimientos, interfaces, etc)

Personal de mantenimiento:

Son quienes administran los cambios y mejoran los procesos

Analistas y Programadores:

Son los responsables del desarrollo del producto.

Personal de pruebas:

Son quienes elaboran y ejecutan los planes de pruebas.

Administradores de proyectos:

Personas que planifican, organizar y controlan un proyecto informático. Gestina todos los recursos necesarios para logran el fin perseguido.

7. REQUISITOS Y CASOS DE USOS

El Proceso de Desarrollo de Software está formado por un conjunto de actividades organizadas y secuenciadas para transformar los requisitos del usuario en un sistema/software.

Este proceso se caracteriza por:

- Estar dirigido por casos de uso
- Estar centrado en la arquitectura
- Ser iterativo e incremental

Los **casos de uso** representan los requisitos funcionales y guían el diseño, la implementación y la prueba. Basándose en los casos de uso, los desarrollares crean modelos de diseño e implementación.

Se dice que el proceso está centrado en la **arquitectura** porque es una vista de diseño con las características más importantes, dejando los detalles de lado.

Es **iterativo** ya que el desarrollo se organiza en una serie de mini proyectos de duración fija, llamados iteraciones (con duraciones entre 2 a 6 semanas).

El resultado de miniproceso es un sistema que puede ser probado, integrado y ejecutado. El sistema crece incrementalmente a lo largo del tiempo, iteración tras iteración.

El proceso unificado tiene una serie de **disciplinas** que son el conjunto de actividades y artefactos vinculados en un área determinada: requisitos, diseño, implementación, etc.

También cuenta con una serie de **fases**. Cada fase es el periodo de tiempo entre dos hitos principales de un proceso de desarrollo: inicio, elaboración, construcción y transición.

Para profundizar el estudio de este contenido, le solicitamos la lectura del siguiente material:

LOS REQUISITOS FUNCIONALES DE LOS CASOS DE USOS

Los casos de uso son una técnica para la **especificación de requisitos funcionales** propuesta inicialmente por Jacobson en 1992. Esta técnica es ampliamente aceptada y existen múltiples propuestas para su realización.

Los requisitos funcionales presentan una ventaja sobre la descripción textual.

Los Casos de Uso se utilizan para documentar los requerimientos funcionales de un sistema desde el punto de vista de los usuarios que responden a la pregunta: *¿Qué se supone que el sistema debe hacer para los usuarios?*

Los Casos de Uso se han vuelto muy populares para establecer el esquema de la lógica de negocios. En el desarrollo se representa la **secuencia de eventos** que suceden entre el actor y el sistema como si fuera una caja negra, donde se muestra el "qué" y no el "cómo". Los casos de usos **guían el proceso de desarrollo.**

Es importante aclarar que los Casos de Uso no son "orientados a objetos".

ACTORES DE LOS CASOS DE USOS

Un actor es alguien o algo que interactúa con el sistema (una persona, una organización, un programa o sistema de hardware o software). Un actor estimula al sistema con algún evento o recibe información del sistema y este siempre externo al mismo.

Un actor cumple un rol definido por Ejemplo un Cliente, un Banco o un empleado.

Tipos de Actores:

Actores primarios: utilizan las funciones principales del sistema.

Actores secundarios: efectúan tareas administrativas o de mantenimiento.

COMO ENCONTRAR LOS ACTORES DE UN SISTEMA

Las preguntas que debemos hacernos son:

- ¿Quién usa el sistema?
- ¿Quién instala o mantiene al sistema?
- ¿Qué otros sistemas utilizan al sistema?
- ¿Quién recibe información del sistema?
- ¿Quién le provee información al sistema?

CASOS DE USOS Y ESCENARIOS

Los casos de uso describen un conjunto de secuencias para un Flujo principal y para Flujos Alternativos.

Un ejemplo podria ser el caso de uso Contratar Empleado podría tener variantes:

a) contratar externos (Escenario más frecuente) y traslado dentro de la misma empresa
b) contratar extranjeros.

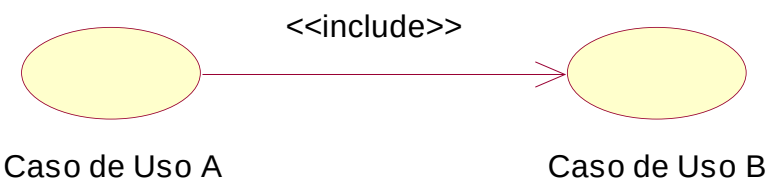
Cada una de estas variantes se puede expresar en una secuencia diferente.

Un escenario es una secuencia específica de acciones entre los actores y el sistema (es una instancia de un caso de uso).

RELACIONES DE INCLUDE

Un caso de uso base incorpora explícitamente el comportamiento de otro caso de uso en el lugar establecido.

El caso de uso base no podría cumplir su objetivo sin el caso de uso incluido



El caso de uso B es parte escencial del CU A

Las ventajas de esta asociación son:

- Las descripciones de los casos de uso son más cortas y se entienden mejor.
- La identificación de funcionalidad común puede ayudar a descubrir el posible uso de componentes ya existentes en la implementación.

Las desventajas son:

- La inclusión de estas relaciones hace que los diagramas sean más difícil de leer, sobre todo para los clientes.

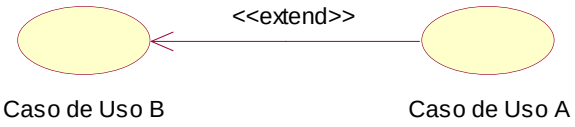
Cuando debemos usar "Include"

- Al incorporar explícitamente el comportamiento de otro caso de uso.
- Para evitar repeticiones de descripción de flujos de eventos.
- Cuando distintos eventos se pueden factorizar en un nuevo caso de uso
- Cuando un caso de uso es muy extenso y difícil de leer.

RELACIONES DE EXTEND

Un caso de uso base incorpora implícitamente el comportamiento de otro caso de uso en el lugar especificado por el caso de uso. Estos se ven los dos flujos como uno solo

La funcionalidad de un caso de uso incluye un conjunto de pasos que ocurren sólo en algunas oportunidades.



El caso de uso B (caso de uso base) incluye en su funcionalidad (opcionalmente) el caso uso A

- Extiende su funcionalidad gracias al agregado implícito de comportamiento
- **Depende de puntos de extensión** y **una condición**.
- ¿Quién nombra a quién?
- Desde el punto de vista del usuario: los dos flujos parecen uno.
- El CU base normalmente no depende del caso de uso que extiende para cumplir su objetivo

Se entiende que se agregan pasos a un Caso de Uso existente, y esto se hace creando un nuevo Caso de Uso, que enriquece al existente, pero ***no lo modifica****.*

Cuando debemos usar "Extend"

- Cuando se desea describir una variación del comportamiento normal de un caso de uso.
- Para conjuntos de eventos que son ejecutados solamente en ciertos casos.
- Cuando la sección de flujos alternativos de un caso de uso se torna muy grande y difícil de leer, es conveniente crear nuevos casos de usos a partir de estos flujos alternativos que extiendan al caso de uso original.