Clase 7:

Casos de uso: conjunto de actividades que permiten obtener una funcionalidad completa, esta funcionalidad puede tener antes otros casos de uso previos denominados ‘precondiciones’.

**Un caso de uso es una funcionalidad completa**

Un caso de uso no es, por ejemplo, ingresar un producto, ingresar cantidad, etc. Tiene que ser funcionalidades completas.

Funcionalidad completa: algo que ejecuta el actor para obtener un beneficio. También puede pasar que no se pueda ejecutar por algún problema.

Qué es una precondición? Aquellos casos de uso (de mi sistema o no) que deban haberse ejecutado previamente para que mi caso de uso sea exitoso, es decir, que se obtenga un beneficio. Siempre es el caso de uso inmediatamente anterior.

Interacciones: cuando alguien externo al sistema interactúa con el.

Los actores pueden ser algo o alguien, siempre afuera del sistema.

**Usuario no es lo mismo que un actor.**

usuario: interactúa con el sistema.

Un usuario puede tener mas de 1 rol en el sistema.

Actor: usuario que se le asume un rol que cumple en el sistema. No necesariamente tiene que ser humano, puede ser un software.

Actor primario: inicia el caso de uso, solo lo puede iniciar un solo actor.

Actor secundario: ayuda al actor primario para poder completar el caso de uso. No todo caso de uso tiene actor secundario.

Caso de uso extendido: le agrega valor a caso de uso, no se tiene que ejecutar obligatoriamente.

Clase 9:

La calidad no tiene que ver con un producto sin errores, tiene que ver con satisfacer lo que el cliente necesita y esta dispuesto a pagar.

No existen los sistemas perfectos, existen con menos o mas tasa de error.

**Validación**: si es el sistema correcto -> Cliente

**Verificación**: si estamos construyendo el sistema correctamente -> con el equipo de desarrollo

Siendo del equipo de desarrollo puede haber un especialista en interpretar al cliente cuando este no puede darse a entender, son nuevos roles que nacen con las metodologías ágiles (SCRUM).

Modelo clásico -> metodologías ágiles

Calidad: cuando hablamos de validar y verificar hablamos de calidad.

Concepto de calidad: naturaleza del producto.

Satisfacer al cliente. Cuando alcanzamos la calidad alcanzamos lo que el cliente pagó.

Cuando verificamos con el equipo de desarrollo, estamos verificando que estemos construyendo el sistema correctamente. Verificamos si es consistente si es correcto y si es coherente. La verificación puede tener el visto bueno del equipo de desarrollo, pero no necesariamente es el sistema que nos pidió el cliente, eso se valida con el cliente. Cuando verificamos solo buscamos saber si el sistema esta correctamente construido

Producto Owner  Integrante del equipo de desarrollo que cumple el rol del cliente para realizar la validación, es una persona entrenada para hacer de cliente. (dato de color)

Un sistema satisface calidad cuando se acordó con el cliente sobre lo que iba a pagar

Gestión de calidad:

• Control de calidad: Realizar un esfuerzo determinado, hacer una producción y después probarlo. (comprar hierro, contratar gente y hacer tornillos, después pesar los tornillos medirlos y demás, los que están bien los guardo en una caja y los que no, los tiro, descarto los no aptos) Es muy costosa en cuanto al desecho. Es barato el proceso, pero los resultados pueden ser muy malos y costosos.

• Aseguramiento de la calidad: En cada parte del proceso se evalúa la calidad. La evaluación constante minimiza el descarte. El proceso es más costoso, pero asegura un alto nivel de rendimiento. Vamos a tener que verificar en todas las etapas.

Validación

Cuando validamos es con el cliente. Buscamos que nos diga si es el sistema correcto. Si lo que él quería/esperaba obtener del sistema que nos pidió está satisfecho. Saber si su expectativa fue cubierta.

Técnicas de validación

Aquellas metodologías y herramientas que utilizamos para que el cliente entienda que si lo que yo le propongo es lo que él necesita. Formas de mostrarle al cliente si lo que le vamos a ofrecer es lo que necesita. El cliente verifica el qué, no el cómo se está haciendo.

• Enmaquetado/Prototipo:

o Poder mostrarle al cliente un aproach, en forma física, de lo que nosotros entendimos que nos pidió. Sirve para que el cliente entienda el concepto/idea de lo que vamos a hacer. Maqueta que hacer un arquitecto en cartón, sirve para que el cliente nos diga o valide si el diseño y la idea es correcta.

• Creación de casos de prueba:

o Es el escenario donde voy a validar ese producto, ese escenario lo tiene que armar el cliente, no el programador. El que determina los casos de prueba, es el que lo va a usar, no el programador. Si no logramos que el cliente haga los casos de prueba, es que lo que estamos haciendo no es lo que el cliente necesita.

Técnicas de verificación

Aquellas metodologías y herramientas que utilizamos para verificar que los requisitos funcionales y no funcionales que obtuvimos con el cliente se cumplan. De manera interna se verifica el cómo se está haciendo, no el qué.

Estándar:

Es una norma que dice como se tienen que hacer las cosas.

Ej: Los autos en argentina tienen los volantes en la izquierda, es por ley.

Según la Norma IEEE 830:

Una especificación de requisitos debe tener un propósito, un ámbito (dominio) apartado de definiciones, acrónimos y abreviaturas. Restricciones  Requisitos no funcionales

**Gestión de calidad:**

* Modelo clásico: control de calidad **-QC-** (esfuerzo determinado, hacer la producción y después probarla)

Ej.: fabrica de tornillos. Compro material, contrato obreros, a fin de mes tengo un depósito de un millón de tornillos.

Los analizo (peso, mido, etc.), los que están bien los guardo en una caja, los que están mal los guardo en otra caja. Descarto los no aptos de la muestra. Se puede tomar una muestra.

Ej.: si nos tomasen una sola prueba al final de la universidad, en vez de tomar todas las pruebas posibles entre jardín y universidad, nos ahorramos tiempo.

* Quality Assurement (**QA**): permanentemente monitoreando/verificando fase del proceso tratando de detectar un error.

Cuando hablamos de ing. De requisitos hablamos de técnicas de verificación y validación, es decir que hay cosas que nosotros podemos hacer antes de entegar, para verificar si estamos construyendo bien nuestro producto, y segundo es validar con el cliente si está correcto.

**Sommerville tiene 3 técnicas de validación (trabajar con el cliente):**

A close-up of a document

Description automatically generated

Los casos de prueba (los crea el cliente) à donde valido.

**(INGENIERÍA DEL SOFTWARE. UN ENFOQUE PRÁCTICO PAG 132 DEL PDF)**

**Verificamos definiciones, diagramas de casos de uso, etc.**

A close-up of a page

Description automatically generated

* Al riesgo que algo se malinterprete: ambigüedad.
* Tenemos un requisito ¿Tenemos la posibilidad de saber como se generó ese requisito?
* El requisito está acotado en términos cuantitativos: que sea medible.
* Relaciones entre requerimientos (Diagrama de casos de uso vemos las relaciones).
* Violación del dominio: si se sale del dominio el cliente no puede especificar los casos de prueba.
* Definir cuál va a ser la forma de medirlo.
* ¿La especificación del caso de uso coindice con el diagrama?

Trabajos de verificación à consistencia interna.

**Negociación**: va consensuando con el cliente el grado de satisfacción.

**Estándar IEEE830 (ÍNDICE) \*\*muestra el documento de especificación de IEEE\*\***

* Introducción
  + Propósito: requisitos del sistema
  + Ámbito: dominio
  + Definiciones acrónimos y abreviaturas (Ej.: que significa despachante, renaper, etc.)
  + Visión general del documento (¿abstract?)
* Descripción general:
  + Perspectiva del producto
  + Funciones del producto: Requisitos funcionales
  + Características del usuario
  + Restricciones: REQ no funcionales
  + Suposiciones y dependencias (inclusiones, extensiones)
  + Requisitos futuros (no lo vimos)
* Requisitos específicos
  + Interfaces externas (en los casos de uso, ej.: pago de algún producto, usabas una página externa).
  + Funciones
  + Requisitos de rendimiento (requisitos no funcionales, eficiencia, seguridad, etc.)

Se generan las historias de usuario a través de una metodología ágil.

Clase 10:

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente Es la historia que nos cuenta el usuario. Es mucho mas sencillo que las especificaciones de casos de uso.

Las historias del usuario son requisitos del cliente.

Las historias de usuario están bien o no, no existe un termino medio.

A diferencia de las especificaciones de los casos de uso, que es una funcionalidad completa, las historias de usuario son mas acotadas, no son funcionalidades completas.

Smart: Es una metodología para definir objetivos. Se trata de un acrónimo del inglés a través del cuál se explican las características básicas de los objetivos SMART. Éstos deben ser Específicos (Specific), Medibles (measurable), alcanzables (Achievable), realistas (Realistic) y de duración limitada (Time-bound). No llevan mas de dos o tres semanas.

Criterios de aceptación: algo que escribe el usuario y se valida entre el equipo de analisis y el usuario mismo. Es un resultado esperado.