Ingenieria de software 1

Teoria: Lunes 11AM

Practica: Miercoles 8AM

Parciales:

Teoria 1 – 19/8/24:

Que es el software?: Instrucciones, procedimientos, reglas, documentacion y datos asociados que forman parte de las operaciones de un sistema de computación.

Características:

* Es un elemento lógico
* Se desarrolla, no se fabrica
* No se desgasta

Tipos de software:

* Genericos: Sistemas aislados producidos por organizaciones desarrolladoras de software y que se venden en un mercado abierto.
* Personalizados: Sistemas requeridos por un cliente en particular.

Clasificación:

* Sistemas
* Relación
* Científico y de ingeniería
* Integrado
* Línea de producto
* Aplicaciones web / móvil
* Inteligencia Artificial

Que es la ingeniería de software?

Disciplina de la ingeniería que comprende todos los aspectos de la producción de software desde la etapas iniciales de la especificación del sistema incluyendo la evolución de este, luego que se comienza a ejecutar.

Se aplican teorías, métodos y herramientas.

Usa métodos sistemáticos cuantificables

Dentro de tiempos y costos estimados

Para el “Desarrollo, operacion y mantenimiento” (Desde la necesidad hasta el producto final)

Introducción:

La comunicación:

Es la principal fuente de error

Es la base para la obtención de las necesidades del cliente.

Requerimientos / Requisitos:

Un requerimiento o requisito es una característica del sistema o una descripción de algo que el sistema es capaz de hacer con el objeto de satisfacer el propósito del sistema.

IEEE-Std-610:

(TEXTO IEEE)

Fuentes de requerimientos:

* Documentacion
* Stakeholders (Es cualquier persona o grupo que se vera afectado por el sistema, directa o indirectamente. (Ingenieros, Gerentes, Experto del dominio, Usuarios Finales))
* Especificaciones de sistemas similares

Tipos de puntos vista:

* Interactuadores: Personas que interactúan directamente con el sistema.
* Indirecto: Stakeholders
* Dominio: (VER PPT FALTA)

Elicitacion de requerimientos:

Es el proceso de adquirir (“Eliciting”)[sonsacar] todo el conocimiento relevante necesario para producir un modelo de los requerimiento de un dominio de problema.

Es una actividad principalmente de carácter social, mucho mas que tecnológico.

Los problemas que se plantean son mas psicológicos, no tan técnicos.

Los objetivos son:

* Conocer el dominio del problema para poder comunicarse con cliente y usuarios y entender sus necesidades.
* Conocer el sistema actual (manual o informatizado)
* Identificar las necesidades, tanto explicitas como implícitas, de clientes y usuarios y sus expectativas sobre el sistema a desarrollar.

Existen ciertos problemas de comunicación como por ejemplo:

* Cultura y vocabulario diferente
* No ser concientes de sus propias necesidades
* Dificultad para expresar claramente las necesidades
* Medios de comunicación inadecuados
* Entre otros.

Técnicas de elicitacion:

Métodos discretos (Información Estatica):

* Muestreo de la documentación, los formularios y los

Métodos interactivos (Mas de la gente):

* (FALTAN PPT)

Muestreo de la documentacion:

Para la recolección de hechos a partir de la documentación existente se usan Organigrama, que muestra el propietario y claves de usuarios.

Documetos que describen la funcionalidad del negocio (FALTA PPT)

Investigación y visitas al sitio:

* Investigar el dominio
* Patrones de soluciones
* Revistas especializadas
* Buscar problemas similares en internet
* Consultar otras organizaciones.

Teoria 2 – 26/8/24

¿Qué es un proceso de software?

Es un conjunto de actividades y resultados asociados que produce un producto de software.

Requerimientos: TIPOS:

Funcionales: Tienen que ver con la interacción del sistema y el usuario. (que es lo que el sistema tiene que hacer)

* Describen una interacción entre el sistema y su ambiente. Como debe comportarse el sistema ante determinado estimulo.
* Describen con detalle la funcionalidad del mismo.
* Describen lo que el sistema debe a ver, o incluso como NO debe comportarse.
* Son independientes de la implementación de la solución.
* Se pueden expresar de distintas formas.

No Funcionales: No se habla del comportamiento del sistema, son detalles del comportamiento

* Describen una restricción sobre el sistema que limita nuestras elecciones en la construcción de una solución al problema.
* EJ: Restricción sobre cómo vamos a construir el sistema
* EJ: Que lo hagan en cierto tiempo
* EJ: Tiempo de respuesta del sistema
* Requerimientos del producto:
  + Especifican el comportamiento del producto. (usabilidad, eficiencia, rendimiento, espacio, fiabilidad, portabilidad)
* Requerimientos organizacionales:
  + Se derivan de las políticas y procedimientos existentes en la organización del cliente y en la del desarrollador. (entrega, implementación, estándares)
* Requerimientos externos:
  + Interoperabilidad, legales, privacidad, seguridad, éticos.

Ingeniería de requerimientos:

* Es el proceso por el cual se transformar los requerimientos declarados por los clientes, ya sean hablados o escritos, a especificaciones precisas, no ambiguas, consistentes y completas del comportamiento del sistema, incluyendo funciones, interfaces, rendimientos y limitaciones.
* Es la disciplina para desarrollar una especificación completa, consistente y no ambigua, la cual servirá como base para acuerdos comunes entre todas las partes involucradas y en donde se describen las funciones que realizara el sistema.
* Se intercambian puntos de vista para recopilar y modelar lo que el sistema va a realizar.

La importancia de la ingeniería de requerimientos:

* Permite gestionar las necesidades del proyecto en forma estructurada.
* Mejora la capacidad de predecir cronogramas de proyectos.
* Disminuye los costos y retrasos del proyecto
* Mejora la calidad de del software
* Mejora la comunicación entre equipos (Tener un documento para ver los requerimientos claros y charlarlo)
* Evita rechazos de usuarios finales. (Evita la queja final de que algo no es como quería el usuario, que, teniendo el documento con los requisitos se puede mostrar)

Estudio de viabilidad:

* A partir de una descripción resumida del sistema se elabora un informe que recomienda la conveniencia o no de realizar el proceso de desarrollo.
* Es un informe que recomienda o no avanzar con el proyecto
* Se hace una mini elicitacion de requerimientos y mini especificación de requerimientos se decide si el proyecto procede o no, ya sea recursos o tiempo.

Especificación de requerimientos:

Propiedades de los requerimientos:

* Necesario: Su omisión provoca una deficiencia.
* Conciso: Fácil de leer y entender.
* Completo: No necesita ampliarse.
* Consistente: No contradictorio con otro.
* No Ambiguo: Tiene una sola implementación.
* Verificable: Puede testearse a traes de inspecciones, pruebas, etc.

Objetivos:

* Permitir que los desarrolladores expliquen como han entendido lo que el cliente pretende del sistema.
* Indicar a los diseñadores que funcionalidad y características va a tener el sistema resultante.
* Indicar al equipo de pruebas que demostraciones llevar a cabo para convencer ala cliente de que el sistema que se le entrega es lo que había pedido.

Cualidades:

* Correcta
* No ambigua
* Completa
* Verificable
* Consistente
* Comprensible por los consumidores
* Modificable
* Rastreable
* Independiente del diseño
* Anotada
* Concisa
* Organizada
* Utilizable en operación y mantenimiento

Documentos:

* Documento de definición de requerimientos: Listado completo de todas las cosas que el cliente espera que haga el sistema propuesto.
* Documento de especificación de requerimientos: Definición de términos técnicos
* Documento de especificación de requerimientos de Software IEEE Standard 830-1998 (SRS)
  + Objetivo: Brindar una colección de buenas practicas para escribir especificaciones de requerimientos de software.

Aspectos básicos de una especificación de requerimientos:

* Funcionalidad
* Interfaces externas (límite del proyecto)
* Rendimiento
* Atributos
* Restricciones de diseño

Validación de requerimientos:

Se evalúa el software para validar si cumple los requerimientos.

2 detalles importantes:

Validación: hacer el software correcto (se hace si o si con el cliente ya que es el que necesita el producto)

Verificación: hacer el software correctamente (se puede hacer sin el cliente)

Comprenden:

* Verificaciones de validez (para todos los usuarios)
* Verificaciones de consistencia (sin contradicciones)
* Verificaciones de completitud (todos los requerimientos)
* Verificaciones de realismo (se pueden implementar)
* FALTAN 2

Pueden ser manuales o automáticas.

Pueden ser formales o informales:

* Formales: el equipo de desarrollo debe conducir al cliente, explicándole las implicaciones de cada requerimiento.
* Informales: los desarrolladores deben tratar los requerimientos con tantos stakeholders como sea posible.

Antes de una revisión formal, es conveniente realizar una revisión informal.

Se construye un prototipo, que a veces los errores o detalles se ven más fácil.

Se generan casos de prueba.

**Técnicas de especificación de requerimientos:**

Estáticas:

Se describe el sistema a través de las entidades y onetos, sus atributos y sus relaciones con otros. No describe como las relaciones cambian con el tiempo.

Cuando el tiempo no es un factor mayor

Ejemplos: ----------------- faltan

Dinámicas:

Se considera un sistema en función de los cambios que ocurren a lo largo del tiempo.

Se considera que el sistema es en un estado particular hasta que un estímulo lo obliga a cambiar su estado.

Ejemplos: tablas de decisión, diagramas de transición d estas, tablas de transcion de estaods, diagramss de persianas, diagramas de transcon extendicdos.

Historias de usuario:

Descripción:

* Una historia de usuario es una descripción corta y simple de un requerimiento de un sistema, que se escribe en lenguaje común del usuario y desde su perspectiva.
* Son utilizadas en las metodologías de desarrollo agiles para la especificación de requerimientos.
* Acompañadas de las

Conceptos:

* Debe ser limitada, esta debería poder escribirse sobre una nota adhesiva pequeña.
* Son una forma rápida de administrar los requisitos de los usuarios sin tener que elaborar gran cantidad de documentos formales y sin requerir de mucho tiempo para administrarlos.
* Permiten responder rápidamente a los requisitos cambiantes
* Al momento de implementar las historias, los desarrolladores deben tener las posibilidades de discutirlas con los clientes.
* Generalmente se espera que la estimación de tiempo de cada historia de usuario se sitúe entre unas 10 horas y un par de semanas.
* Estimaciones –FALTA

Debe responder a tres preguntas:

1. ¿Quién se beneficia?
2. ¿Que se busca?
3. ¿Cuál es el beneficio?

Esquema:

Como (rol) quiero (algo) para (beneficio)

Ejemplo:

Como cliente quiero suscribirme por medio del sitio web para obtener un nuevo plan de tv por cable.

Características:

* Independientes unas de otras: de ser necesario, combinar las historias dependientes o buscar otra forma de dividir las historias de manera que resulten independientes.
* Negociables: FALTA
* FALTA
* FALTA
* FALTA
* FALTA

Criterios de aceptación:

* Un criterio de aceptación es el criterio por el cual se define si una historia de usuario fue desarrollada según la expectativa del producto maager/owner
* FALTA
* Los cripterios de aceptación son utilizados para expresa el resultado de las converasiciones del cliente con el desarrollador. El cliente debería ser quien las escriba mas que el desarrollador.
* Representan el inicio de la definición del como. No están diseñados para ser tan detallados como una especificación de diseño tradicional.
* FALTA
* FALTA

Plantilla: (que se usaba antes originalmente)

* Adelante:
  + ID: Identificador univoco de la historia expresado como texto generalmente de la forma <verbo> <sustantivo>
  + Título: descripción de la historia de la forma: como <rol> quiero <algo> para <beneficio>
  + Reglas de negocio: conjunto de reglas, normas, políticas, etc. Que condicionan el modo de operación.
* Atrás:
  + Criterios de aceptación (serian como los casos de prueba): Criterios por los cuales una historia cumple con las expectativas del cliente.
  + Formato:
    - Escenario 1: titulo del criterio
    - Dado <un contexto inicial>
    - Cuando <ocurre un evento>
    - Entonces <garantiza uno o mas resultados>
  + Así hasta N escenarios.

Beneficios:

* Al ser muy corta, esta representa requisitos del modelo de negocio que pueden implementarse rápidamente (días o semanas)
* Necesitan poco mantenimiento
* Mantienen una relación cercana con el cliente
* Permite dividir los proyectos en pequeñas entregas
* Permite estimar fácilmente el esfuerzo de desarrollo
* Es ideal para proyectos con requisitos volátiles o no muy claros.

Limitaciones:

* Sin criterios de aceptación pueden quedar abiertas a distintas interpretaciones haciendo difícil utilizarlas como base para un contrato.
* Se requiere un contacto permanente con el cliente durante el proyecto lo cual puede ser difícil o costoso.
* Podría resultar difícil escalar a proyectos grandes
* Requiere desarrolladore muy competentes

Épicas:

* Se denomina épica a un conjunto de historias de usuario que se agrupan por algún denominador común.
* FALTA
* FALTA
* FALTA

Es parecida a como se escriben las historias de usuarios pero son claramente mas grandes

Practica – 28/8/24:

Historias de usuario:

* Una historia de usuario es una descripción corta y simple de un requerimiento de un sistema, que s escribe en un lenguaje común del usuario y desde su perspectiva.
* Son utilizadas en las metodologías de desarrollo agiles.
* La historia de usuario debe responder a tres preguntas:
  + Quien se beneficia
  + Que se quiere
  + Cuál es su beneficio

Ejemplo:

Función:

* Matricularse al instituto, la persona.
* Inscribirse en los cursos que se dictan en el instituto, los matriculados.

Roles:

* La persona. (no inscripta)
* Los matriculados.

Regla de negocio:

* un DNI no puede estar registrado dos veces con diferentes matriculas.
* Cada curso tiene un cupo de 30 personas.
* Un alumno solo se puede inscribir una vez a un mismo curso.
* Todos los pagos se realizan con tarjeta de crédito.
* Banco valide el número de tarjeta y verifique que el saldo sea suficiente.

Historias de usuario:

* Matricular persona
* Inscribir matriculado a curso
* Pagar con tarjeta

Para los pagos la catedra se centra en 3 cosas:

* Pago sea correcto
* La tarjeta exista
* Que se pueda conectar al servidor

En el de los pagos tienen que estar todos los casos.

HISTORIA DE USUARIO DE MATRICULAR PERSONA:

ID: Matricular Persona

Titulo: como persona quiero matricularme

Reverso:

Generalmente se pone en el escenario 1 el exitoso.

Teoría – 2/9/24:

Casos de uso:

Refleja la interacción entre el usuario y el sistema

Proceso de modelado de las fncionalidad del sistema en termino de los eventos que interatcuan entre los usuarios y el sistema.

El uso del CU facilita la intereaccion con el usuario.

Beneficios de casos de uso:

* Herramienta para capturar requerimientos funcionales.
* Falta
* Falta
* Falta
* Falta
* Falta
* Falta
* Define una línea base para toda la documentación del sistema.
* Proporciona una herramienta para el seguimiento de los requisitos.

Componentes:

Diagramas de casos de uso: ilustra las interacciones entre el sistema y los actores

Escenarios (narración del CU): Descripción de la interacción entre el actor y el sistema para realizar la funcionalidad.

Diagrama:

Representa un objetivo (funcionalidad) individual del sistema y describe la secuencia de actividades y de interacciones para alcanzarla.

Para que el CU sea considerado un requerimiento debe estar acompañado de su respectivo escenario.

Actores en diagrama:

Un actor inicia una actividad (CU) en el sistema. (Puede haber actores que no inicien una actividad)

Representa un papel desempeñado por un usuario que interactúa (rol)

Puede ser una persona, sistema externo o dispositivo externo que dispare un evento (sensor, reloj)

Relaciones en diagrama:

* Asociaciones: relación entre un actor y un CU en el que interactúan entre sí.
* Extensiones:
  + Un CU extiende la funcionalidad de otro CU
  + Un CU puede tener muchos CU extensiones
  + Los CU extensiones solo son iniciados por un CU
* Uso o inclusión: reduce la redundancia entre dos o mas CU al combinar los pasos comunes de los CU
* Herencia.