Apuntes Ingeniería del Proceso Software

Contenido

Conceptos generales

Proceso: **actividades que se realizan**. Por debajo de los procesos están las técnicas.

Método/Practica: **forma de hacer algo**. El **proceso especifica qué hacer** mientras que la **práctica indica cómo** hacerlo

Metodología: **conjuntos de prácticas, técnicas, procedimientos y reglas usados por sujetos que trabajan en una determinada disciplina.**

Procesos genéricos:

- **Análisis** (que hacer)
- **Diseño** (como hacerlo)
- Codificación/ **desarrollo** (hacerlo)
- **Prueba** (encontrar fallos y depurarlos)
- **Mantenimiento** (mejorar, adaptar, corregir)

Este proceso no es secuencial, en cada punto se pueden volver atrás.

Las metodologías a menudo **difieren** en **cómo organizar los procesos en el tiempo**.

Tipos de requisitos

- **Funcionales:** servicios y funciones que tendrá el producto. Cómo se debe comportar.
- **No funcionales:** Restricciones de cómo se implementará el producto. Capacidad de almacenamiento, seguridad, herramientas para utilizar durante el desarrollo…

SCRUM

Describe como se gestionan y organizan los procesos, en lugar de describir los procesos en sí. Es un **framework** (marco de gestión y trabajo) más que una metodología**. Métrica separa los procesos** de desarrollo y gestión mientras que Scrum lo une todo.

Se basa en ciclos cortos (1-4 semanas) denominados **Sprints**. **Al final** de cada Sprint se realizará una **entrega operativa** del software.

Necesita que los **equipos** sean **multifuncionales**, autoorganizados, autogestionados y altamente colaborativos.

Se utiliza bastante en la industria, especialmente **en proyectos** cuyos **requisitos** no están prefijados (son **volátiles**) y requiere **gran confianza y colaboración entre el cliente y el equipo**.

Scrum no impide detallar, solo indica que no se especifique todo al principio, sino que se realiza de forma incremental**. Se debe especificar a medida que se profundiza en el sistema.**

Roles

El equipo está formado por:

Dueño del producto (Product Owner)

Representa al **negocio** y a los **clientes**. **Interactúa con el resto de partes interesadas** (otros usuarios y clientes finales). Tiene **total autoridad** en la toma de **decisiones** (producto, alcance, presupuesto, prioridades…) **Interactúa** activa y frecuentemente **con el equipo**. Mide las **prioridades**.

Gestor de Scrum (Scrum master)

Experto en Scrum, ayuda a aplicar la metodología. Intenta **solucionar bloqueos e impedimentos** y actúa de **intermediario cuando hay conflictos**. Puede ser parte del equipo de desarrollo, pero se desaconseja. **No es el jefe del equipo**, ya que los Scrum teams son autogestionados. Se adapta a la experiencia del equipo.

Equipo de Scrum (Scrum team)

Se encargan de todos los procesos de desarrollo. Son **equipos autogestionados** de **7±2** personas, con un alto grado de **autonomía**, **responsabilidad** y **compromiso**. Son **multifuncionales**, **aunque tenga expertos** en temas concretos esto **no debería suponer un impedimento.**

Elementos de SCRUM

Product Backlog (pila de producto)

Lista priorizada de funcionalidades (y mejoras, historias de investigación, correcciones…). **Las estimaciones las suministra el equipo**, mientras que **el dueño del producto asigna la prioridad y el valor de negocio**. Scrum **no indica como estimar ni como detallar el backlog**, aunque se aconseja **hacerlo en base a las historias de usuario.**

En el **Backlog va todo lo que se va a hacer**, aunque no otorgue valor. Lo que no está en el backlog no existe. En el backlog **también se introducen cosas técnicas**. En el backlog **se añaden las funcionalidades con una prioridad y una estimación del esfuerzo** para poder decidir qué hacer y qué no hacer en un sprint.

Historias de Usuario

Descripciones claras y concisas de la funcionalidad en términos del valor que aporta al usuario final del producto. Son **independientes**. Su especificación no está completamente detallada. Son **pequeñas**, que **se puedan estimar y probar.**

Estructura típica:

Como [rol] quiero [característica] para [valor de negocio]

Los **detalles adicionales** se consiguen a través de **conversaciones** con el cliente. Se pactan unos **criterios de aceptación que definen cuando está completa.**

En un primer paso podemos dejarnos de detalles, ya que Scrum es incremental y no hace falta hacerlo todo al principio. Las historias de usuario se iran desarrollando en el momento que sea necesario.

Es importante identificar los roles, es decir, los interlocutores de la aplicación. Las historias expresan una función y deben realizarse en un sprint.

EPIC: Historia de usuario demasiado grande y que deben dividirse en historias más pequeñas.

Las historias de usuario deben cumplir las características **INVEST**:

- Independent
- Negotiable
- Valuable
- Estimable
- Small
- Testeable

Criterios de aceptación

La **historia de usuario es solo una frase que describe una funcionalidad**. Los **criterios** de **aceptación** son los **detalles de su realización**. Son los **requisitos que debe cumplir el producto para ser aceptado.** A veces al describir los criterios de aceptación se descubre que una historia es demasiado grande y se desglosa en historias de usuario más pequeñas.

Un criterio de aceptación **puede tener una o varias pruebas de aceptación**, dependiendo del criterio. Junto a los criterios de aceptación **puede haber vistas**, donde se pone **aquello que se va a tratar más tarde.**

Sprint Backlog

Lista de historias de usuario que se ha acordado realizar durante el sprint, así como sus **tareas asociadas** y otras **tareas necesarias** que **no** están **asociadas a historias** (mejoras, historias de investigación…). El resultado del Sprint debe ser un **producto potencialmente estable**.

Planificación de cada Sprint

Un **sprint** es un **bloque de tiempo entre 1-4 semanas** durante el cual **se lleva a cabo un incremento del producto** y al final se obtiene un producto **potencialmente estable**.

Sprint Planning Meeting

Consta de dos reuniones de medio día cada una.

Reunión 1

Se **define qué se va a hacer en ese sprint** y se intenta **entender que quiere el dueño del producto**. El equipo, junto al dueño del producto, **revisa y discute la pila incluyendo historias de usuario y criterios de aceptación**. Se llega a un **compromiso de qué se realizará en el sprint.** El **dueño del producto** decide las **prioridades** y el **equipo** decide **el volumen de tareas** a realizar.

Reunión 2

Se define cómo se va a hacer. Se deja de pensar en términos económicos y se piensa en términos técnicos. **Se planifica detalladamente cada tarea** (información adicional, diseño de pantallas, cambios en la BBDD…)

Tras haber medido el coste de cada tarea**, se decide que va a entrar en el sprint**. El **dueño del producto puede no asistir**, pero debe estar disponible.

Se fija quien es **el responsable de cada tarea**, aunque en última instancia el responsable es el equipo. **Se reparten los elementos de la pila hasta acabar el tiempo disponible** durante el sprint.

Cambios durante el sprint

- **No se debe cambiar la estimación de tiempo** fijada al principio del Sprint
- **El Dueño del producto puede cambiar el backlog antes de empezar el sprint**
- **Cualquier cambio** se deja **para otro sprint**
- **Las tareas pueden evolucionar** durante el sprint

Con esto se busca que **el equipo esté protegido durante el sprint** y que **el DP priorice correctamente.**

¿Qué significa hecho?

En general, **cuando algo es potencialmente entregable**. Puede ser que pase las pruebas, cumpla la normativa… **Debe definirse que se considera hecho**, ya que lo que no se considere hecho no se puede mostrar al cliente.

Sprint diario (Stand-up meeting)

Reunión corta de todo el equipo en la que **cada miembro informa de lo que ha hecho desde la última reunión y lo que tiene pensado hacer antes de la siguiente**, así como **cualquier** tipo de **bloqueo** o **impedimento** que tenga. No hay discusiones y es **aconsejable** que **no acudan** los **jefes**.

Refinamiento de la pila de producto

Facilita la planificación de Sprints posteriores. Se realiza **durante un sprint y toma el 5-10% del tiempo** del sprint, cerca del final.

- Análisis de requisitos
- División de elementos grandes en pequeños
- Estimación de nuevos elementos

Cierre del Sprint

Reunión 1 -> Revisión del Sprint

Se realiza una **demo y conversación en profundidad entre el equipo y el dueño del producto**. Se realiza **una evaluación del producto obtenido** para adaptarlo en posteriores Sprints y **obtener feedback del cliente**.

Reunión 2 -> Retrospectiva del Sprint

S**e inspecciona y adapta el proceso. Se analiza que fue bien**, que no, **que puede mejorar**. No hace falta que esté el DP.

Se actualiza el backlog y la gráfica del trabajo restante del producto.

Otras prácticas

Planificación de entregas

Se eligen que historias se incluirán en la siguiente versión y cuando (incluirá varios Sprints)

- Alcance fijo -> fecha variable
- Fecha fija -> Alcance variable
- Fecha y alcance fijos -> Problemas

Mapeo de Historias (Story mapping)

Damos al backlog estructura bidimensional

- **Filas**: Las **historias de usuario** con sus prioridades
- **Columnas**: Agrupa los **procesos/usuarios**

Pruebas

Son parte integral y esencial de todos los métodos agiles

TDD: Test-Driven Development

Primero se definen las pruebas y se implementa el código para que no fallen las pruebas.

BDD: Behaviour-Driven Development

Evolución del TDD. Las historias de usuario se convierten en pruebas.

Otros métodos ágiles

- **Extreme programming**
- **Concepto de Lean:** Se elimina todo lo innecesario que no aporta valor.
- **Kanban:** Limita el trabajo en proceso. No hay reasignación de tareas, empezar una al acabar otra
- **Scrumban:** En vez de estimar cada ítem del backlog, tener una estimación del promedio de cada uno. Se reduce la planificación de las iteraciones.

Métrica V3

Metodología de planificación, desarrollo y mantenimiento de sistemas de información. Define **procesos** que se estructuran en **actividades**, y estas en **tareas**. **El plan de trabajo es el que nos indica el orden de realización de los procesos.** Estas metodologías no abarcan solo el desarrollo, sino también cosas transversales importantes, sobretodo en aplicaciones grandes.

Se divide en procesos:

- **PSI** – **P**lanificación de **S**istemas de **I**nformación
- **DSI** – **D**esarrollo de **S**istemas de **I**nformación
    - **EVS** – **E**studio de **V**iabilidad del **S**istema
    - **ASI** – **A**nálisis del **S**istema de **I**nformación
    - **DSI** – **D**iseño del **S**istema de **I**nformación
    - **CSI** – **C**onstrucción del **S**istema de **I**nformación
    - **IAS** – **I**mplantación y **A**ceptación del **S**istema
- **MSI** – **M**antenimiento de **S**istemas de **I**nformación

DSI - Desarrollo del Sistema de Información

EVS – Estudio de Viabilidad del Sistema

Analiza las necesidades del cliente/usuario para proponer una solución a corto plazo. Como **resultado** se definen uno o varios **proyectos**. Nos sirve para **delimitar** el **alcance** del **proyecto**.

ASI – Análisis del Sistema de Información

Su **objetivo es obtener una especificación detallada del sistema de información que satisfaga las necesidades de información de los usuarios y sirva de base para el posterior diseño del sistema**. Transforma los requisitos del usuario en requisitos del sistema.

Está dividido en 8 actividades:

- ASI 1 – **Definición del Sistema**
- ASI 2 – **Establecimiento de los requisitos**
- ASI 3 – **Identificación de Subsistemas de Análisis**
- ASI 4 – **Análisis de los casos de uso**
- ASI 5 – **Análisis de clases**
- Las ASI 6 y 7 no las tenemos en cuenta
- ASI 8 – **Definición de interfaces de Usuario**

ASI 1 – Definición del sistema

Su objetivo es **realizar una descripción del sistema**, así como **definir su alcance**, las **interfaces con otros sistemas** e identificar a **los usuarios representativos**. Esto está desarrollado en parte en el EVS.

Tareas

1. Determinación del Alcance del Sistema
2. Identificación del entorno tecnológico
3. Especificación de estándares y normas
4. Identificación de usuarios participantes y finales

ASI 2 – Establecimiento de requisitos

Definición, análisis y validación de los requisitos, así como completar el catálogo de requisitos obtenido en el paso anterior.

Tareas

1. Obtención de requisitos
2. Especificación de los casos de uso
3. Análisis de requisitos
4. Validación de requisitos

Especificación de los casos de uso

Suelen especificarse mediante diagramas que representan una visión general y jerarquizada del dominio. El caso de uso es la función a realizar, el diagrama muestra los actores y los casos de uso generales.

Partes

- **Actor** : representa un stakeholder que interacciona con el sistema
- **Caso de uso:** representa una funcionalidad encaminada a conseguir un objetivo
- **Escenario:** descripción del caso de uso. Describe detalladamente el requisito.

Lo importante es la descripción de la funcionalidad, **el escenario**. Este ese el contenido del caso de uso.

En SCRUM**, las historias de usuario junto con los criterios de aceptación y las vistas equivaldrían al escenario en Métrica.**

La principal **diferencia entre el escenario y las historias de usuario** reside en que el **escenario muestra los pasos a realizar en forma de secuencia**, mientras que **los criterios de aceptación** de las historias **nos indican las características que debe tener**, en lugar de como funciona.

Análisis y validación de los requisitos

Revisar los requisitos con el usuario para determinar su validez y confirmar el catálogo de requisitos.

En Métrica está todo más detallado, ya que el análisis y la validación pueden ser realizados por un equipo distinto. En SCRUM inicialmente no está tan detallado.

ASI 3 – Identificación de Subsistemas de Análisis

Descomponer el sistema en subsistemas para facilitar el análisis. División en base a procesos similares o que trabajan sobre los mismos datos.

ASI 4 – Análisis de casos de uso

Identificar las clases cuyos objetos se necesitan para realizar un caso de uso y describir su comportamiento mediante interacción de dichos objetos. Se hace un análisis para cada caso de uso.

ASI 5 – Análisis de clases

Describir cada una de las clases. Analizar asociaciones para descubrir su tipo, añadir la cardinalidad correcta. Buscar generalizaciones o especificaciones (herencia)

Resultado: Modelo de Dominio.

ASI 8 – Definición de Interfaces de usuario

Especificación de las interfaces entre el usuario y el sistema.

IMPORTANTE: Identificar grupos de usuarios.

Técnicas: Prototipado, diagrama de transición de estados…

Tareas

1. Especificación de principios generales de la interfaz
2. Identificación de perfiles y diálogos
3. Especificación de formatos individuales de la interfaz
4. Especificación del comportamiento dinámico de la interfaz
5. Especificación de formatos de impresión

DSI – Diseño del Sistema de información

Su **objetivo** es definir la arquitectura del sistema, el **entorno tecnológico**, **describir** los **componentes** del sistema.

- DSI 1 – **Definición de la arquitectura del sistema**
- DSI 2 – **Diseño de la arquitectura de soporte**
- DSI 3 – **Diseño de casos de uso reales**
- DSI 4 – **Diseño de clases**
- DSI 6 – **Diseño físico de datos**

DSI 1 – Definición de la arquitectura del sistema

Tareas

1. Definición de niveles de arquitectura
2. Identificación de requisitos de diseño y construcción
3. Especificación de excepciones
4. Especificación de estándares y normas de diseño y construcción
5. Identificación de subsistemas de diseño
6. Especificación del entorno tecnológico
7. Especialización de requisitos de operación y seguridad

RUP – Proceso unificado

Metodología intermedia entre una basada en procesos (métrica) y **una ágil** (Scrum). Es un **proceso iterativo e incremental.**

Estructura

- Fases
- Workflows (disciplinas, se corresponderían con los procesos de métrica 3)

Organización: **cada fase se organiza en iteraciones** y **cada iteración incluye diferentes disciplinas** (con mayor o menor intensidad).

Disciplinas

1. **Modelado de negocio:** Conocer la organización y sus necesidades
2. **Requisitos:** análisis del sistema, basado en casos de uso
3. **Diseño:** arquitectura, interfaces, datos, programas
4. **Implementación:** codificación
5. **Prueba :** planificar, diseñar y ejecutar pruebas a distintos niveles
6. **Despliegue:** integrar e instalar el software en la organización
7. **Gestión de cambios y de configuración**
8. **Gestión del proyecto:** identificar riesgos, planificar iteraciones, seguimiento
9. **Entorno:** soporte del proceso de desarrollo mediante procesos y herramientas

Fases

Comienzo/Incepcion

Tiene como propósito **definir y acordar el alcance del proyecto con las partes interesadas**.

- **Visión:** organiza y resume los requisitos y principales restricciones
- **Casos de uso críticos:** define el alcance y los principales casos de uso
- **Business case:** describe el contexto del negocio, objetivos y criterios de éxito
- Arquitectura candidata
- Evaluación inicial de riesgos
- Plan de proyecto inicial

Elaboración

Selección de casos de uso que permitan definir la arquitectura base del sistema. Identifica todos los casos de uso, describiendo la mayoría. **Plan de proyecto incluyendo todas las iteraciones**

Construcción

Refinar requisitos y diseño. **Programación del sistema y realización de pruebas en cada iteración**

Transición

Asegurar que el software esté disponible para usuarios finales. Ajustar errores u defectos, capacitar a los usuarios y proveer soporte técnico. **Verificar que el producto entregado cumpla las especificaciones.**

Iteraciones e incrementos

Desarrollo incremental

Mejora el proceso de desarrollo dividiéndolo en partes, en cada una de las cuales se incrementa la funcionalidad del sistema a desarrollar

Desarrollo iterativo

Mejora el producto, ya que **se reescriben partes del producto cuando sea necesario.**

Extreme programming (XP)

Proceso altamente iterativo (1-3 semanas/iteración) útil para proyectos arriesgados, con requisitos muy dinámicos y con el cliente muy implicado (siempre disponible).

Documentación

Incluir **aquello que nosotros quisiéramos ver si nos diesen el proyecto y tuviésemos que realizar su mantenimiento**. La documentación **debe aportar valor.**

Especificación

Normalmente se tienen unos requisitos de usuario y de sistema

- **Requisitos de usuario:** en forma de lista jerárquica
- **Requisitos del sistema:** dependiendo de la metodología
    - **Métrica V3:** Lo visto aquí con los casos de uso, modelos de datos del dominio
    - **SCRUM:** El backlog con los criterios de aceptación. Story mapping. Modelo de datos del dominio.

Errores comunes en la especificación

- **No seguir ninguna metodología** e intentar forzar todo para generar documentación que no aporta valor.
- Hacer los requisitos del sistema que incluyen menor nivel de detalle que los de usuario
- **Forzar el uso de una plantilla para casos de uso** donde ocupa más espacio la plantilla que la información útil

Diseño

- **Diferenciar claramente la arquitectura** de otros aspectos de diseño de detalle.
- **Describir muy claramente los conceptos de la arquitectura y frameworks utilizados.** Utilizar gráficos
- En el diseño de detalle es importante **describir los conceptos sobre cómo funciona la aplicación**, no limitarse a poner gráficos.

Errores comunes en el diseño

- **Sobresimplificar** la arquitectura
- **Incluir demasiado detalle** sin describir los conceptos y principios generales del diseño