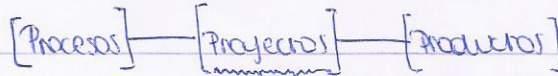


REVISAR Cronograma regularmente

Clase 17/8

Técnica para requerimientos funcionales

PROCESOS EMPÍRICOS



se adapta al proceso
Pero viene definido de fábrica

PERSONAS
Parte fundamental de la Ingeniería de Software
Profesión humano intensiva

Planificación y control

Procesos - definidos - PDB y RUP
Intentan ser "completos", todo lo necesario para el desarrollo de software.
definido de antemano distinto de la persona que lo implementa.

empíricos - Ágil (agile) - surgió en el software. manifiesto ágil ¡LEER!
(2001)

inspección
adaptación
transparencia

Lean (liviana) - Toyota system
• el que toma las decisiones es el que hace el trabajo.
• basados en la experiencia (may que conseguiria.)
• los ciclos de desarrollo son cortos e iterativos.

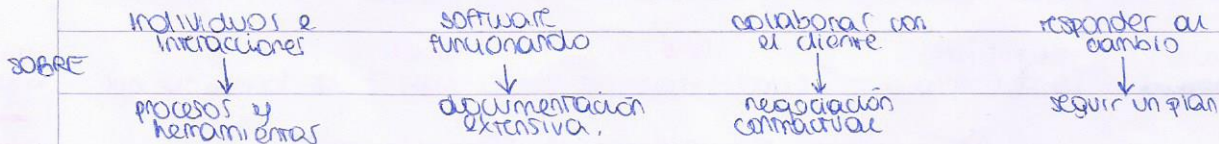
3 PILARES para la exp. basados en retroalimentación

Inspección
Adaptación
Retroalimentación: experiencia → se gana con el equipo y sus particularidades.
transparencia - visible para todos, información de todos, si hay errores lo ven todos.

La mejor comunicación es la de cara a cara.

VALORES ÁGILES

(de manera frecuente)



una versión del producto cada poco tiempo.

Product owner: Responsable de las user stories
Alguien que conoce al producto y el negocio
Los "no técnicos": del negocio
sabe lo que necesita el negocio. PRIORIZA.

PRINCIPIOS del Manifiesto ágil

Equipos auto-organizados - donde no hay jefe

↓
surgen las mejores arquitecturas y requisitos

(casi el 50% de los req.)

Requerimientos emergentes - surgen en el momento de la creación del producto

Requerimientos desconocidos - no nos damos cuenta que existen, y eran necesarios.

Entregar software de forma frecuente en entornos cambiantes. involucrar al cliente.

ÁGIL - no es metodología o proceso

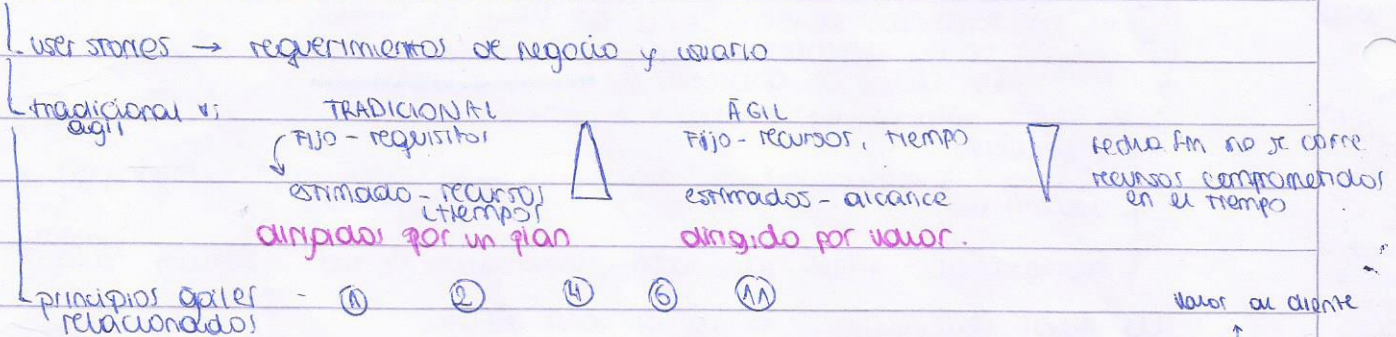
↓
compromiso entre nada de proceso y demasiado proceso
↓
el equilibrio con lo que determina

Algunos frameworks

↳ RUP, ITIL, Crystal, Scrum, XP

de negocio, no software entrega frecuente y temprana.

- Requerimientos en Agile** - lo importante es el valor del negocio que genera el software (medio para un fin)
 - más duro que req. los ^{historia} del usuario deben ser claros y con el lenguaje.
 - Priorizar**: "solo lo suficiente"
 - hay muchas cosas que no usas NUNCA
 - parado: el 80% del valor lo da el 20% de la funcionalidad.
 - priorizamos** lo más importante para el negocio (que se da en las primeras etapas de desarrollo)
 - Product backlog - escala de prioridad de características (user stories) - las más importantes (1% del total del software)
 - cosa priorizada
 - just in time** - describimos req. según haga falta.
 - eliminar desperdicios. (no invertir tiempo en req. que van a cambiar).
 - comunicación eficiente** → cara a cara.

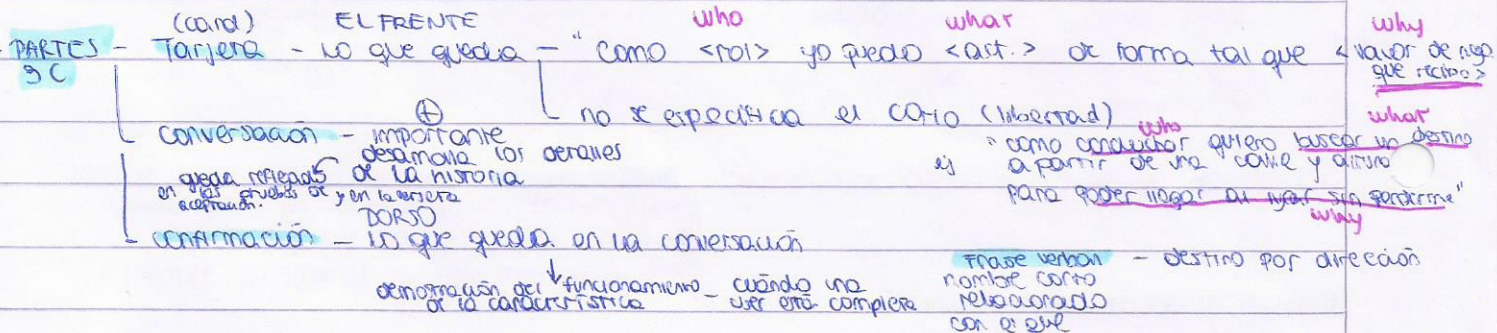


Fred Brooks: NO silver bullets → ESENCIAL

The mythical Man month

USER STORIES - la parte más difícil de hacer software son los requerimientos

- no hay solución perfecta para el problema
- definir partes del producto



son multipropósito - necesidad del usuario - descripción del producto - items de planificación - recordatorio de conversación - código + us = documentación del producto.

product owner prioriza las user stories en el product backlog

- el tamaño depende de cuantas iteraciones son necesarias para completarla.

porciones verticales - con una parte de cada capa desde UI hasta BD

modelado de roles - importante, también determinado con tarjetas.

usuarios representativos (proxies) - cuando el PO no está disponible

- eliminarlo cuando sea posible

critérios de aceptación de user stories - formalización para definir las pruebas de aceptación - user proof

- definen una intención no una solución

los detalles quedan en la conversación y en la documentación y en las pruebas.

2

debe incluir los criterios de aceptación.

Pruebas de aceptación - (pasan) (fallan)
- hay positivas y negativas

La acuerdo, si funciona lo acordado se acepta la user.

Definition of Ready DoR

definición de listo - control de calidad cuando esta lista para ser incluido en el sprint backlog.

aplicado a la user. Apto para ser integrado en una iteración

definido por el equipo

Definition of Done

una buena user stories

determina si la user está terminada. completa y correcta. si lo define si p. para o producción.

INVEST (INVEST)

Definition of Done

Independent - Puede implementarse en cualquier orden

Negotiable - historia definida en "qué" no el "cómo". conversación.

Valuable - debe tener valor para el negocio (no al desarrollo, tecnologías)

Estimable - cuando puede asignarse un número (costos) Tamaño, complejidad y esfuerzo.

Small - consumible en una iteración, comienza y termina en la iteración (sprint)

Testable - demostrar que fueron implementadas

tamaño relativo al equipo y el tiempo.

Algo más

NO ERS

no son especificaciones detalladas de requerimientos

niveles de granularidad / abstracción

maintain goal software - Mike cohn

epic - user story grande, hay descomponerla en user más pequeñas. (dejamos para después, prioridad baja)

theme - colección de user stories relacionadas

User story: descripción de una funcionalidad desde la perspectiva del cliente o usuario

Clase 24/08

Requerimientos ágiles

no establecidos desde un comienzo

FOCUS: valor del negocio. Software como medio de entrega de valor

partimos desde una "visión" de producto → Define la primera versión del producto.

Product backlog: nunca está completo

Necesitamos una colección de historias de comienzo

apunta a recibir cambios hasta en etapas finales.

no podemos tener una visión final del producto al comienzo. No definimos de antemano definición

lista - cola PRIORIZADA → fn. del product owner (que forma parte del negocio)

JUST IN TIME: diferir decisiones

detallar solo lo que vale la pena en el momento

User stories - usada en XP Scrum (adoptada, no nativa).

gestión binaria - está terminada o no: No hay porcentaje de completitud

vs

gestión tradicional - porcentaje de avance: realmente no se puede medir

por esto hacemos user stories, para poder entregar valor rápidamente.

Definition of done - define cuando historia se puede mostrar al product owner.

definido por el equipo de desarrollo.

expectativas y necesidades.

visible para todo el equipo.

TABLEROS visualización del trabajo:
Sprint
To do
Doing
Done

SPRINT BACKLOG

comunicación cara a cara
reuniones
región de equipos auto organizados

el producto tiene calidad cuando cumple con los req.