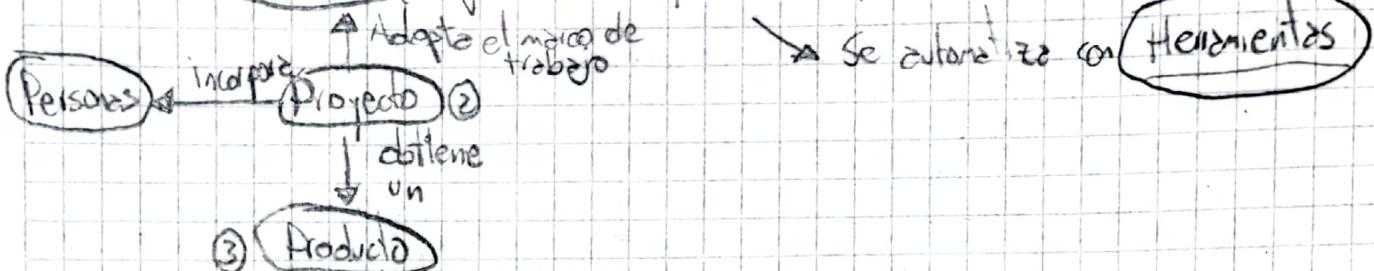


Filminas SCM

Concepto de Software → Software en Contexto

① **(Proceso)** (conjunto de tareas/actividades)



Evolución del Software (la pirámide) → De lo Útil a lo Usable

↳ La conveniencia es limitante xq → mayor funcionalidad que le intento agregar y no es útil, deteriora la integridad y entrego un SW de menor calidad ↳ Requerimientos que usan mayormente son ≈ 30% (Reg. JIT)

SCM → Concepto
↳ Transversal a todo el proyecto pero perdura a lo largo del ciclo de vida del producto

Disciplinas SCM = control de calidad de proceso + control de calidad de producto + Pruebas SW = Manejo / Gestión de la Integridad

Integridad → Medio que me asegura la calidad

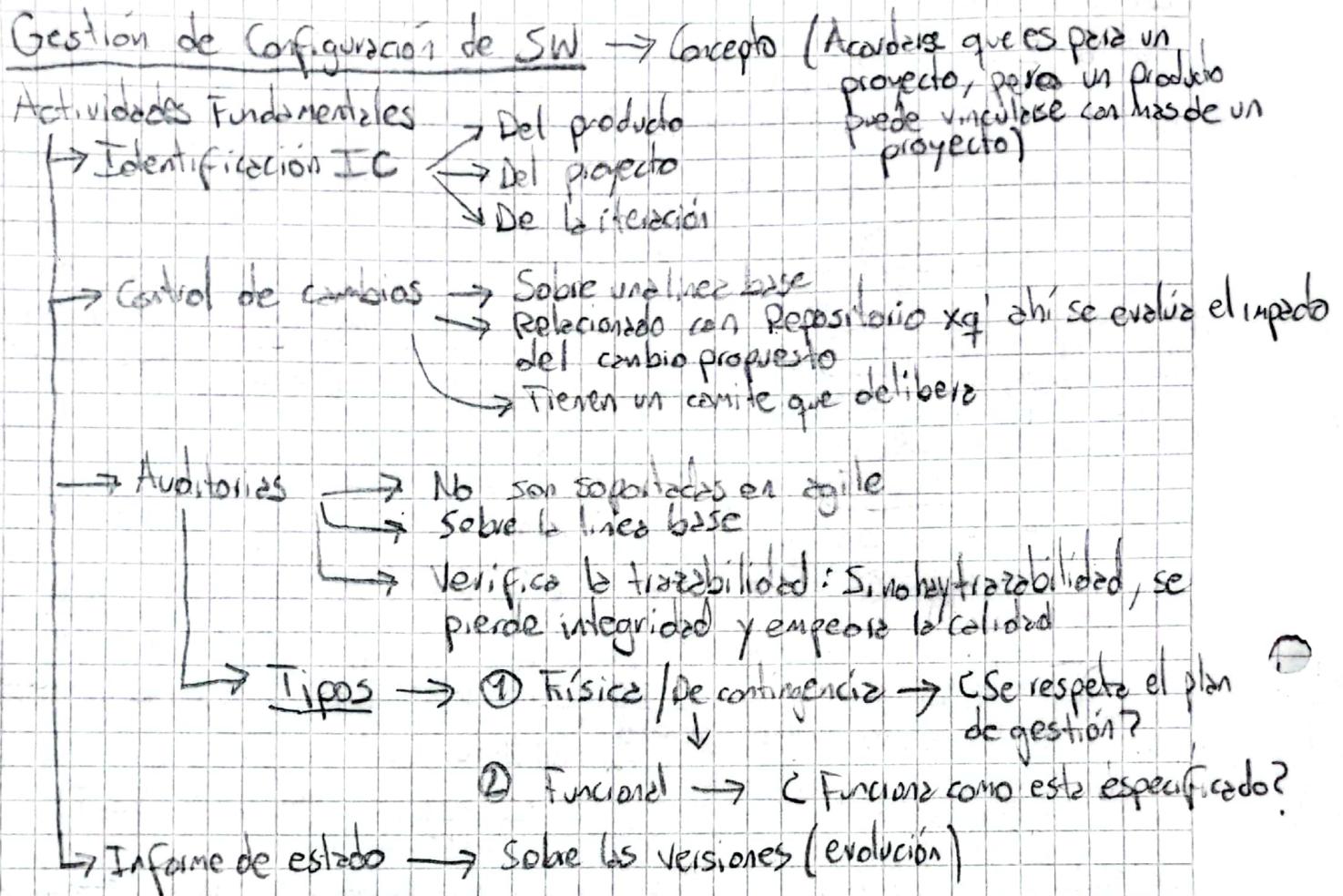
Problemas en el manejo de componentes → Pierdo integridad (en algún concepto)

Ej.

- Perdida de un componente / cambios → pierdo Trazabilidad
- Doble mantenimiento → pierdo recursos

Conceptos Clave

- Item de Config.
- Versión
- Variante → Versión de un IC que evolucionó por separado → BRANCH
- Configuración del SW
- Repositorio → Centralizado
→ Descentralizado
- Línea Base ≠ Versión → Es la Configuración revisada formalmente
 - ↳ De especificación
 - ↳ De productos (a.e. hayan pasado por el control de calidad de SCM)
- acordarse: Calidad → Integridad → Estabilidad
Requiere Proporcional



Plan de Gestión de Configuración

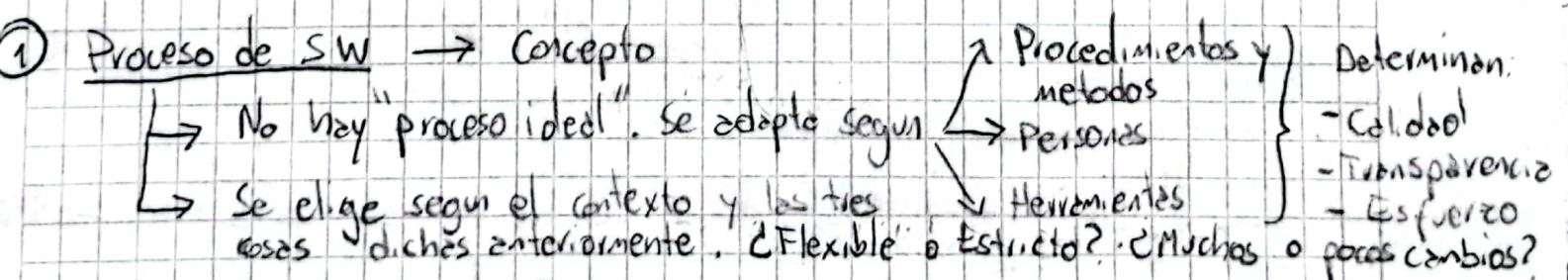
- → planificación es importante para el éxito de un proyecto.
- La gestión de configuración es parte de una disciplina de soporte y también debe planificarse

SCM en Agile (Ver)

Familias Componentes de Proyecto (se puede dividir en)

- ① Proceso
- ② Proyecto
- ③ Producto

de SW



Tipos de procesos

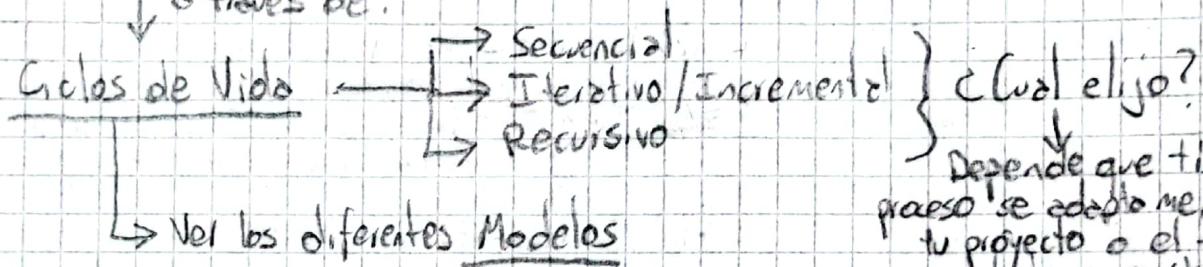
- Empírico → Experiencia → Flexibilidad → Inspección y Adaptación cte.
- No permiten ciclos de vida secuenciales ni actividades inflexibles frente a constantes cambios
- Transparentes a los personas que lo conforman

→ Definido → Estructurado → No tan flexible → Control de riesgos
 → Es + importante otras disci.
 plines como la planificación
 y la administración
 (xq'si acepta cambios
 pero es más difícil)
 → Usan cualquier ciclo de vida

Los MODELOS DE PROCESOS especifican → Fases del proceso (actividades)

se representan en
 un proyecto o producto
 a través de:

→ Orden que se llevan a cabo



② Proyectos de SW → Concepto

→ Características → Orientado a objetivos
 → Limitado en el tiempo (A diferencia del producto, que
 → Esfuerzo / Recursos su utilidad no cesa con el fin
 → Únicos del desarrollo)

→ Administración → Primordial para asegurar la calidad → Éxito del proyecto
 → Actividades → Planificación → A través del Plan de Proyecto
 → Monitoreo y Control

Por más que un proyecto tenga una buena administración, se puede perder la calidad del proyecto si se desbalancean alguno de los tres factores

$$\text{Alcance} = \text{Requerimientos} / \text{Funcionalidad}$$



$$\text{Tiempo} = \text{Fecha calendario a entregar}$$

LA RESTRICCION TRIPLE

$$\text{Costo} = \text{Recursos implicados en el desarrollo}$$

¿Cómo se seleccionan?
 Después lo veremos

Es el Líder de Proyecto quien balancea esto

Equipo de proyecto → Concepto
 → Características

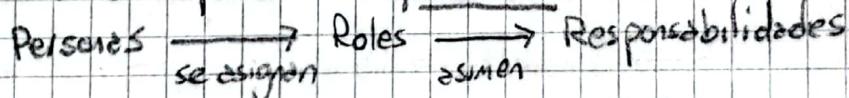
PLAN DE PROYECTO → Antecedentes → Es una 'hoja de ruta' del proyecto
 → contiene planes de soporte.

→ Que IMPLICA

- Definición del objetivo del Proyecto. Responde al ¿Qué? → sólo de contado:
- Alcances → Del producto → Características - ERS
→ Del proyecto → Solo el TRABAJO - Plan de Proyecto
 - Definición del ciclo de Vida (Teniendo en cuenta los criterios nombrados)
 - Estimación → Deriva de los requerimientos / alcances
 - Enfoque agil → Story Points
 - Enfoque tradicional.
 - ① Tamaño - Del producto. ¿Como medida? → Cantidad de Requerimientos
 - Leyes de Uso
 - ② Esfuerzo - Deriva del tamaño (Hrs/personas)
 - Líneas de código (antes)
 - ③ Tiempo - Descomponemos los alcances en tareas y determinamos cuanto tiempo llevará
 - abarca el 80% de los costos
 - ④ Costo - Económicamente, depende del esfuerzo y por ende, del tamaño

- Gestión de Riesgos → Asociadas a una PROBABILIDAD de ocurrencia
 - Ponen en riesgo el objetivo del proyecto
 - Si hay certeza, NO es un riesgo, es un PROBLEMA
 - Se miden en → Probabilidad
 - Impacto (Daño que ocasionaría si sucede)
 - EXPOSICIÓN → Medida cuantitativa del riesgo

→ Asignación de responsabilidades / Recursos



- Calendarización → Deriva de la estimación del tiempo
 - se descompone al minimo detalle las tareas

- Definición de métricas → Medidas numéricas que aportan visibilidad del avance del proyecto, proceso o producto

→ Enfoque Agil → Incrementos x Iteración

→ Enfoque tradicional

- Tamaño del producto
- Esfuerzo
- Tiempo (Calendario)
- Defectos
- Costos

La diferencia es que
Estimación es previo, sobre algo que no se sabe a ciencia cierta. La métrica es una medición durante el desarrollo

* Hay que balancear las expectativas de las métricas según las restricciones para poder asegurar un producto de calidad *

- Monitoreo y Control → Verificar como va el proyecto según las métricas
 - Garantiza mayor probabilidad de éxito

PLANIFICADO VS LO REAL

Filminas Gestión de Productos

③ Productos de SW → ¿Para que los creamos?

↳ Satisfacer clientes

↳ Dinero

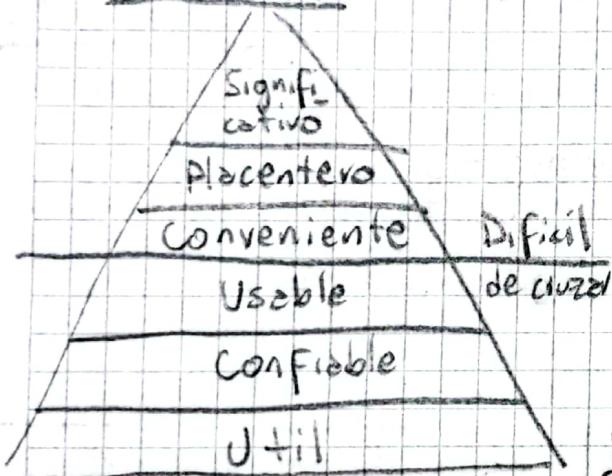
↳ Cambiar el mundo

↳ Tener muchos usuarios logrados

¿Qué características de un producto de SW usamos?

- Un 70% SIEMPRE
- Un 13% FRECUENTEMENTE } Hay que concentrarse en estas caract.
- Un 16% A VECES
- El 64% restante, MUY pocas veces o NUNCA

Evolución de los Productos de SW



Ver Concepto de MVPs, MMFs y MFVs con el dinosario Lean/Agile

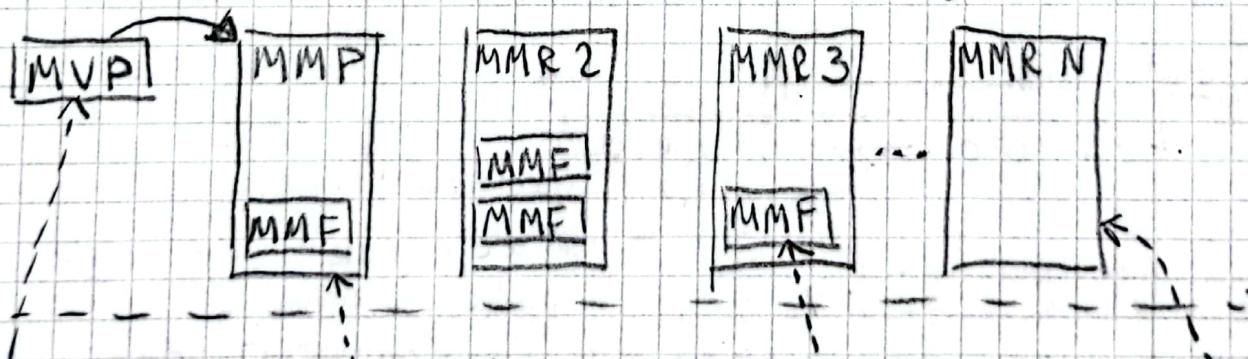
- Siempre se empieza por lo base funcional/útil
(Aquellas características que más se usan)

- El software se hace confiable (en especies de seguridad y resultados)

- El software se hace usable (con respecto a comodidad / satisfacción)

- Es difícil cruzar la linea de lo usable a lo conveniente, porque que sea conveniente para el desempeño de ciertas tareas hace que los usuarios dependan mucho del producto. Además, implicaría describir las características que no se usan muy a menudo, degradando la calidad del software y costando más tiempo, esfuerzo y recursos

Relación entre MVP, MMF, MMR y MMF (Productos mínimos para la gestión de productos)



Mínimo producto visible

- Versión de un nuevo producto con menor esfuerzo

- Utilizado para APRENDER
NOTA: - Mas prototipo que producto

Primer release de un MMR N = MMR1

- Dirigido a los primeros usuarios

Recordar!

* VIABLE → Puedes algo con menor esfuerzo
* COMERCIALIZABLE → Vendible, se puede desplegar con valor

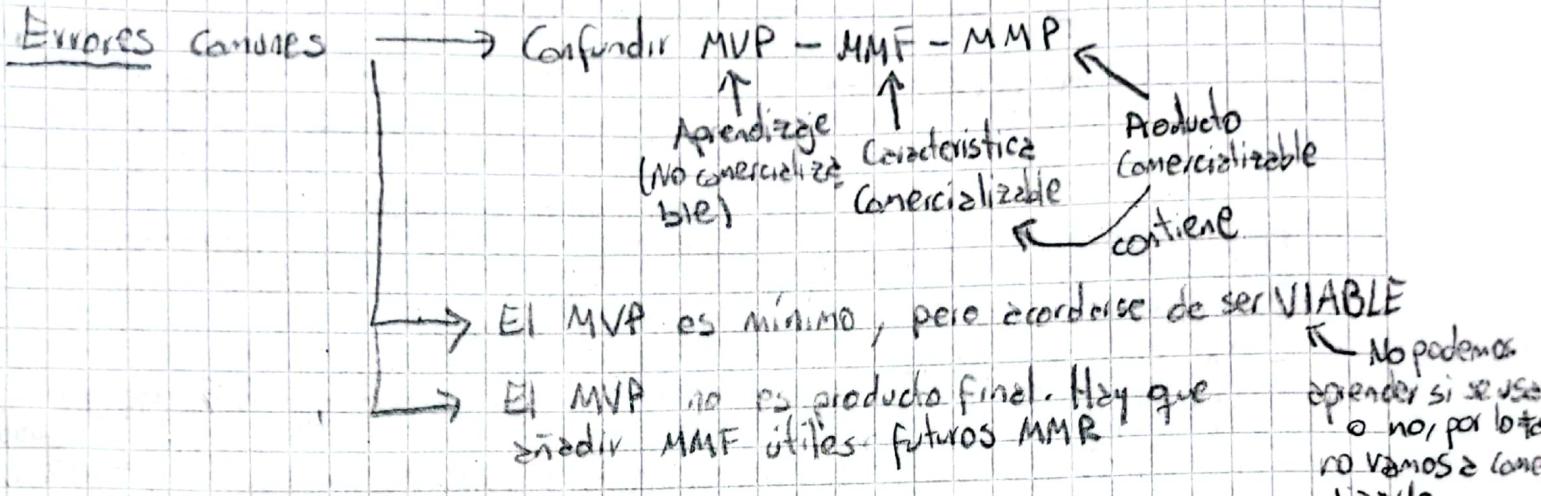
Mínimo característica comercializable

- Pieza más pequeña de funcionalidad que puede ser liberada

Mínimo release comercializable

- Release de un producto de características + pequeño que ofiere valor

MVP > MVF → La característica mínima viable es un mini MVP (MVF)



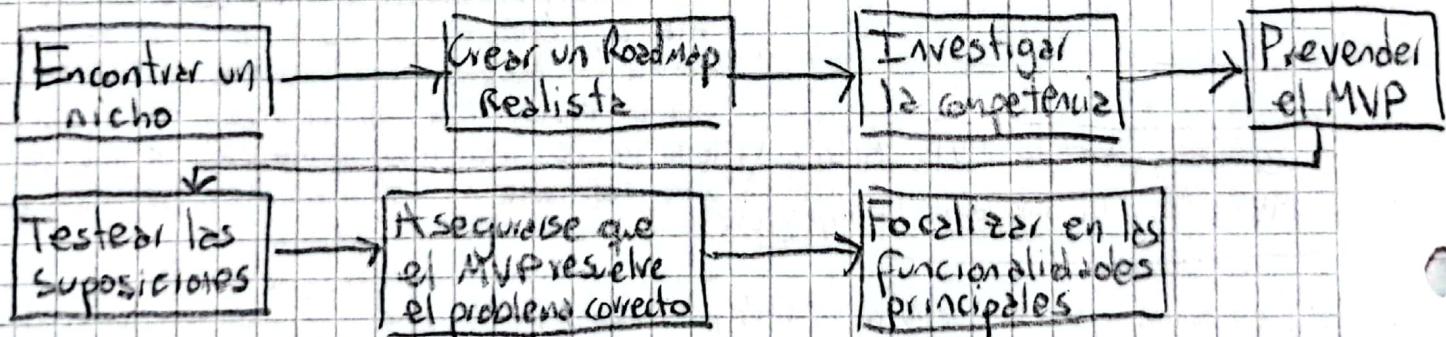
Un Startup mide su productividad en términos de averiguar la cosa correcta a construir día a día. Si es beneficioso para el cliente, propone VALOR

El éxito no es entregar un producto, sino entregar algo que el cliente usará

BUILD - EXPERIMENT - LEARN feedback loop

- Permite descubrir las necesidades del cliente y ordenarlas
- Ver el dibujito Lean UX página 85

Preparar un MVP



Características de un MVP

- Diseño
 - Adecuado
 - UX que deleite
 - Satisfacer aspecto visual / interacción
- Usabilidad
 - Suficiente para que la gente lo quiera comprar
 - Util para su público objetivo
- Confidibilidad
 - Los early adopter confíen en su solución
 - Incluso cuando tiene poco tiempo en el mercado
- Funcionalidad
 - Necesaria para solucionar un problema específico
 - Satisfacer las demandas y permite evaluar futuras implementaciones

Filosofías Requerimientos Agiles

- User Stories

Filosofía de desarrollo Agil desarrollada en MANIFIESTO AGIL (2001)

→ No es una metodología / proceso

→ Es un pensamiento / ideosincrasia

Agilismo

uso

Procesos Empíricos

3 pilares

→ TRANSPARENCIA

→ INSPECCION

→ ADAPTACION

Valores

- ① Individuos en interacciones, por sobre procesos y herramientas
- ② Software Funcionando por sobre documentación excesiva (Esto NO quiere decir que no hay que documentar)
- ③ Colaboración con cliente por sobre negociación contractual
- ④ Responder a cambios sobre seguir un plan

Los valores agiles son todo lo contrario a los valores de los procesos tradicionales.

- ① El contexto / sensación del cliente y equipo VS lo que dice un proceso
- ② Software funcionando = valor VS Documentación sin uso / valor inmediato
- ③ El cliente es parte del proceso VS El cliente recibe el producto negociado al final
- ④ La importancia viene de la adaptación VS La planificación previa

Por otro lado TRANSPARENCIA permite → INSPECCION influye sin → ADAPTACION

→ Principios: Ver los 12 principios y como se relacionan con los valores (pagina 30)

→ Frameworks Agiles → Aplicable en el contexto de lo complejo

FDD ATDD Scrum XP

Cerca de la certeza / Lejos del azarido
Lejos de la certeza / Cercas del azarido

Ser Agil ≠ Ser veloz

Ser Agil ≠ Ser indisciplinado

La idea no es construir de a pedacitos, sino algo funcional pequeño e ir mejorandolo y adaptandolo segun necesidades del cliente

Desarrollo en ambientes Ágiles

Desarrollo + Gestión = Ingeniería de requerimientos

④ Requerimientos Ágiles → Fundamento en los valores de Agile

→ Pilares

- ① Usar el valor para construir lo correcto = útil para el cliente y negocio
- ② Determinar que es "solo lo suficiente" = se empieza pequeño y se genera retroalimentación
- ③ Usar historias y modelos para mostrar que construir = trabajamos junto al cliente.

Es una etapa de mucha comunicación, para encontrar una visión del producto en conjunto

→ Big Requirements Up-front - BRUF

→ Es la rueda de los requerimientos. Solo se usa el 70% siempre. Priorizamos esas funcionalidades

↓
rebasan con
"BRUF"

Gestión de Requerimientos en ambientes Ágiles

→ a través del

PRODUCT BACKLOG

→ A diferencia del enfoque tradicional, donde el equipo de desarrollo prioriza los requerimientos, en ágil lo hace el cliente, según el valor de negocio

es

→ Lista priorizada de requerimientos pueden → Cambiar

→ Eliminarse

→ Agregarse

Los requerimientos "persisten" en el producto, en cambio en el enfoque tradicional en la ERS. Si se cambian, se cambia el documento.

⑤ (sigue) Requerimientos Ágiles

→ Fuente → Comunicación cara a cara VS ERS (enfoque tradicional)

→ Tipos = Dominio del Problema

Requerimientos de Negocio

Requerimientos de Usuario

Dominio de la Solución

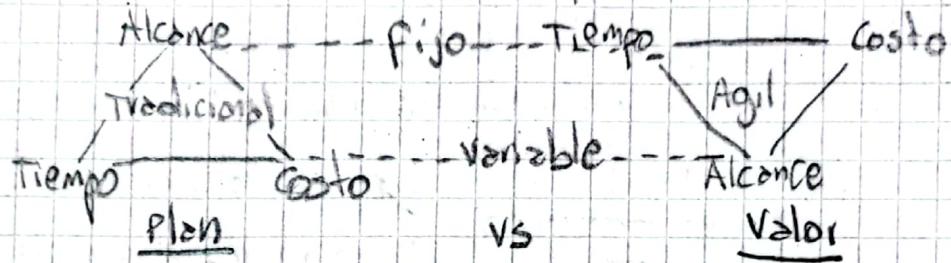
Requerimientos de Software

A nivel de la gestión ágil y lo que le interesa al cliente son los de negocio y usuario

→ Requerimientos JIT (Just in Time)

- El product Backlog nunca esté al 100% xq' se trabaja con los requerimientos cuando se necesita
- El enfoque tradicional especifica todo el principio, por lo q' un error tiene alto impacto

→ La triple restricción vs Enfoque tradicional



→ Asumir que

- Van a haber cambios constantemente
- Stakeholders (los involucrados): no son todos los que están
- "El usuario dice lo que quiere cuando lo recibió"
- No hay técnicas ni herramientas para todos los casos
- Lo importante es entregar un resultado, una solución de "Valor"
- Principios Agiles relacionados = ①, ②, ④, ⑥ y ⑪ (Ver)

User Stories → "Lo mas importante es decidir que constituir"

- Técnica para trabajar requerimientos agiles.
- Lo importante no es lo que se escribe, sino lo que uno habla por eso es tan importante (diría que lo +) la comunicación
- Mientras la información sube y la incertidumbre baja, a lo largo del proyecto, se toman las decisiones.
- Es ≠ a los Casos de uso (Eso son del dominio de la solución y nosotros trabajamos en el problema)
- Los partes / Los 3 "Cs" de una US
 - Card = Parte visible de la US. Contiene el resumen de la conversación.
 - Conversation = Parte + importante, como resultado aparece la card
 - Confirmation = Resultado de las pruebas
- El PO prioriza en el PB en cualquier momento (Flexibilidad)
 - Mientras + Alto + importantes / Prioridades

- Posiciones Verticales → tenemos que Desarrollar → GUI, Business Logic, Database } Impacta en la arquitectura
- Modelado de roles → Tarjetas de Rol de Usuario
 mediante
 → Por persona o por Personajes Extremos
 → evita → Usuarios Representantes (Ej: Cliente, Vendedor, etc)
- Criterios de aceptación → ¿Como se debe comportar el SW para que el PO lo acepte?
 → Definen límites
 → Ayuda a los testers a definir las pruebas
 → Ayudan a los desarrolladores a saber cuando poner de agregar funcionalidades
 → Son buenas si:
 → Definen una intención
 → Son independientes de la implementación
 → De alto nivel
- Pruebas de aceptación → Expresan detalles de la conversación
 → ¿Qué hay que probar? → No es posible TODAS
- Definición de listo - D.O.R → ¿Esta en condiciones la US para entrar en la Iteración de desarrollo?
 ↓
 La US debe satisfacer el INVEST
 → Independiente
 → Negociable
 → Valuable
 → Estimable
 → Small
 → Testable } Detalles en pag 49
- Definición de hecho - D.O.D → ¿Esta implementada completa y correctamente, para ser mostrada al PO? → Aporta Valor → Desarrollar
- Spikes → Tipo especial de US
 → Quito el riesgo / Incertidumbre → Si existe, NO es estimable
 → Dos tipos → Técnicas → ¿Como implemento algo?
 → Funcionales → ¿Como funcionará o lo usará el usuario?

- ↳ Lineamientos
 - Estimables, demostrables y aceptables
 - La excepción, no la regla (es la última opción)
 - Implementarla en una iteración separada

Filmos Estimaciones Ágiles

Estimación Ágil → Presunciones numéricas asociadas a la probabilidad de que ocurra cierto evento

- Estima quien hace el trabajo → + comunicación (el equipo) (valor agil, interacción)
- No son compromisos → Estimo → Planifico

→ ¿Qué estimamos? → User Stories

→ ¿Cómo estimamos? → Story Points → Unidad de medida RELATIVA

↓
Se compara con otra cosa (otro US)

Tiene en cuenta

- Incertidumbre → definen el TAMAÑO
- Complejidad → TAMAÑO
- Esfuerzo → "PESO" de US

→ Tamaño /S Esfuerzo

↓
Conlleva HS lineales (Ideales)

↓
Depende de quien realiza el trabajo Fibonacci

↓
¿Qué tan compleja es una US?

→ No depende de quien lo haga
↓
Se recomienda estimar en Horas

→ Velocidad → Se MIDE, NO se estima (Métrico)

↓
Progreso del equipo → Story Points x Iteración

→ Poker Planning → Técnica de estimación (ver detalles pag 58)

→ Reestimaciones → Acordarse Agile es EMPIRICO

→ Adoptarse y cambiar si modifica la opinión del tamaño de una US

Filmos Estimaciones (en Enfoques Tradicionales)

Estimación Tradicional → Aproximación sobre una medida (Métricas)

↓
¿Qué y como? → Ver en Plan de Proy.

- Al igual que en ambientes ágiles NO son compromisos
- No siempre estima el equipo → Líder de proyecto → Errores
solo conduce
- ¿Para qué estimamos? → Predecir Completitud } Planificación y
Administración riesgos } Control
- ↓
¿Cuanto costará, cuanto tiempo, cuanto trabajo? → Previo al desarrollo hay poca precisión
Estimaciones difieren 2-4 veces ↘
↳ Se reestiman

- Problemas en el enfoque tradicional → Lo base son las estimaciones
 - Estimar pronto → se planifica previo al desarrollo
 - Inflexibles / Duros con los cambios → Poca reestimación
 - Degrado ↗ ↙ Incumplimiento ↙ ↗ Pretensiones poco realistas
calidad en el tiempo / costo
- Técnicas / Métodos (pag 52)
 - ↳ Basados en la experiencia
 - Datos Históricos
 - Juicio Experto
 - Puro
 - Delphi
 - Analogías