Procesos empíricos

PPP –> procesos, proyecto, producto.

Ahora se incorpora otra P --> Personas.

Somos una actividad humano-intensiva y las personas tienen el poder de influir en el éxito o fracaso de un proyecto.

Procesos definidos → PUD, RUP. Intentan ser una expresión de completitud, expresan todo lo que necesitamos, y en cada proyecto se toma lo que necesita. Está definido de ante mano y por gente diferente a la que realizara el proyecto.

Procesos empíricos → utilizado por la filosofía ágil y lean. Basados en la experiencia. ¿Pero cómo la consigo? Se basan en tres pilares

- Inspección
- Adaptación
- Transparencia → fundamental para que funcionen los otros dos. Debe haber transparencia entre las personas y la información que se comparte, blanquear situaciones. Se basa en la comunicación. El objetivo, estado, situación deben ser visibles siempre para todo el mundo. El manejo de las personas y sus relaciones son lo más difícil de manejar. La mejor comunicación es la cara a cara.

Los ciclos de desarrollo tienen que ser cortos. Todos los procesos empíricos se basan en ciclos de vida iterativos sí o sí. Esto es porque necesito experiencia, que la gano cuando tengo ciclos rápidos y cortos con mucha retroalimentación.

En la retroalimentación de los tres pilares aparece la adquisición de experiencia. La experiencia se gana en un contexto y con un mismo equipo

Los frameworks agiles y lean te dan solo algunas pautas para ciertas cosas. No procesos ni metodologías.

Acá quiero decidir que procesos voy a usar. Lo menos que se necesite para que pueda funcionar.

PRINCIPIOS AGILES

Requerimientos emergentes → los que aparecen mientras el producto se está haciendo, creciendo y evolucionando. El 50% de los R surgen a medida que se ven avances y se van ocurriendo cosas. El otro 50% son los conocidos, que nos dice el cliente, y los desconocidos, que son desconocidos porque nosotros no preguntamos, es decir por errores nuestros.

VALORES AGILES

Valoramos todo, lo que está arriba se valora más, pero no quiere decir que el resto no se valore.

Filosofía ágil valora más las personas y las relaciones y no tanto el software y las herramientas

Como enfoque debo valorar lo que valora el cliente

1. Individuos e interacción sobre los procesos y las herramientas
2. Software funcionando sobre la documentación extensiva

<u>Desarrollo Ágil de Software (Agile)</u>

Agilismo es mucho más estricto respecto de sus formas de ejecutarse

Agilismo no es la legalización del lio.

Algunos frameworks ágiles son

- ATDD
- FDD
- Crystal
- XP
- Scrum

Lo importante no es el software que construimos sino el valor del producto que le entregamos al cliente. El software lo vemos como una herramienta para llegar al fin.

<u>Requerimientos en Agile</u>

(los cambios son bienvenidos aun en las etapas finales de un proyecto)

Historias de Usuario

- La descripción de la historia debe ser corta, y debe apuntar a describir una necesidad del usuario, que en la medida de lo posible lo escriba el, en su lenguaje.

Priorización

- En general nos cuesta mucho entender y determinar qué es lo bueno y suficiente para entregarle al cliente.
- Hay un altísimo porcentaje de funcionalidad que no se usa realmente, ver como ejemplo el Excel, power point, etc.

- Debemos enfocarnos en esto a la hora de decidir cuáles son nuestras prioridades en cada reléase (20% de las funcionalidades)

Just in time

- Viene a pelear contra la definición completa de antemano de todos los requerimientos. Establece que los vamos a encontrar, analizar, describir a medida que vayan apareciendo. Analizo solo cuando necesito, y de esa manera eliminamos desperdicios, lo que tiene que ver con no invertir tiempo en escribir requerimientos que despues van a cambiar o desaparecer.

Triple Restricción o Triangulo de hierro

Dimensiones en las que se mueve un proyecto en función de las decisiones que se tienen que tomar

- Alcance
- Costo
- Tiempo

Se condicionan y dependen entre si

Cada iteración → duran lo que el equipo decida y no se puede cambiar. El equipo debe permanecer fijo, y entre ellos acuerdan que se va a entregar para esos requerimientos y para ese tiempo

USER STORY

Es un producto

Partes de una User Story (las tres C)

- Card
- Conversation → lo más importante
- Confirmation

Las 3 W (sintaxis)

- Who I want
- What I want → lo más importante, implica el valor del negocio
- Why

En ningún momento se especifica el cómo. En lenguaje común del negocio

Se recomienda acompañar esta Card con criterios de aceptación.

Frase verbal: nombre corto. Tiene que ver con el que. Permite un acceso más rápido a las user stories. NO las reemplaza.

El Product Owner debe priorizar las historias en el Product Backlog

Criterios de Aceptación

Nada es obvio, lo que es obvio para mi puede no serlo para los demás.

- La altura de la calle es un numero entero
- La búsqueda no se puede demorar más de 30 segundos

Detalles

Los detalles deben ir en las pruebas

- INVEST MODEL

I- Independent: calenderizables e implementables en cualquier orden

N- Negotiable: el “qué” no el “como”

V- Valuable: debe tener valor para el cliente

E- Estimable: hay tantas cosas que no tiene respuesta que no las puedo estimar. Es el proceso que nos pasa cuando nos damos cuenta de que es más lo que no sabemos de la user que lo que si sabemos. La user tiene que ser estimable porque necesito ese valor para saber si me puedo comprometer a desarrollarla o no- Es estimable cuando puedo asignarle un valor numérico.

S- Small: consumible en una iteración. Iteración que scrum llama sprint. Hay técnicas para particionar user stories cuando son muy grandes.

T- Testeable: que se puede demostrar que esa user se implementó.

Esto es la base y lo mínimo, pero se le pueden sumar cosas, por ejemplo, un prototipo, especificar reglas de negocio si no las hay.

Pregunta obligatoria del articulo --> no silver bullet (lo pregunta en el parcial)

¿Que son los requerimientos agiles?

Se debe tener en cuenta el valor del negocio – nosotros hacemos software como un medio para entregar un valor de negocio al cliente.

Parto de algo que se llama VISION del producto, no del producto completo. La visión debe responder al valor del producto y me da la definición de la primera versión del producto que voy a tener.

La visión del producto me va a permitir generar el PRODUCT BACKLOG. Lo mínimo para un product backlog es la cantidad suficiente de historias como para ejecutar la primera iteración, que nos va a permitir tener retroalimentación. El product backlog es una pila o cola priorizada. Del problema de la priorización se encarga el Product Owner.

Just in time → diferir decisiones hasta el último momento responsable. Viene del lean. Alineado con la idea de no especificar de antemano, cosas que quizás despues no necesitamos (primer principio lean)

Principio de comunicación.

Los mejores requerimientos emergen de los equipos autoorganizados.

DOR (definition of ready) se construye de mínima con el INVEST MODEL. La historia está lista para incorporarse desde el product backlog al sprint backlog

SPRINT BACKLOG → tres columnas: to do, doing, done.

DOD (definition of done): significa que esa historia está en condiciones de ser mostrada al product Owner.

Una épica es una user story muy grande, que tenemos que descomponer en user stories más pequeñas.

SPIKE es una user story con un nivel de incertidumbre que no puedo estimar. No cumple con la E del INVEST. Dos tipos

- Relacionadas con la tecnología – COMO
- Funcional relacionada con el PO, negocio. Indefiniciones.

Requerimientos ágiles

Estimaciones agiles tiene sus características y sustentos teóricos que apuntan a porque funcionan como funcionen

Características

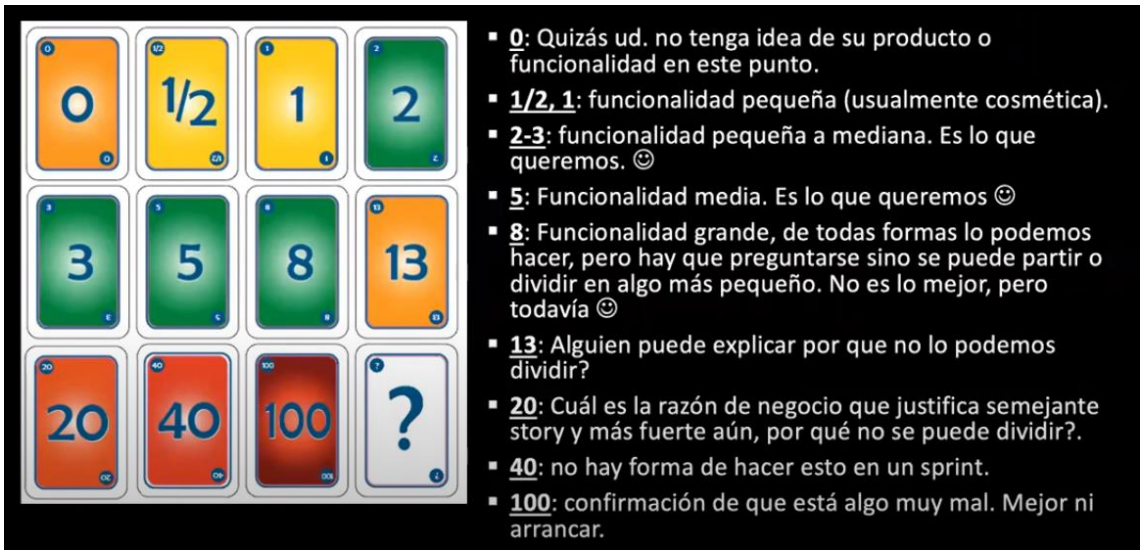
- Relativas: tienen que ver con que los seres humanos somos mejores comparando, es decir que sabemos responder mejor cuando hacemos comparaciones. Hacemos estimaciones por comparación.
- Foco en la certeza no en la precisión: porque la precisión es cara, debo invertir más recursos y tiempo. Los enfoques tradicionales hacen mucho énfasis en la precisión, y luego puede que no cumplan.
- Diferir las decisiones: no hagamos un esfuerzo en estimar todo el producto, se va estimando conforme vaya habiendo partes. Se estima en todo momento, hay momentos donde se estima más general, con granularidad más gruesa, y otros donde se lo hace más especifico.
- Estima el que hace el trabajo: en contraposición a lo tradicional que estima el líder del proyecto, cuando él no tiene nada que ver con la gente que va a hacer el trabajo. Acá estima un equipo empoderado que son quienes efectivamente van a hacer el trabajo

Unidad de estimación en las stories

La forma de estimar una user story se llama → Story Point: es la unidad de estimación de las US. Representa el tamaño de la US.

Cuando estimo necesito dar un valor cuantificado.

Serie de Fibonacci: si la elegimos usar para las estimaciones, usar: 0,1, 1, 2, 3, 5, 8, 13. Tiene un crecimiento exponencial al igual que el software, por eso se recomienda. Se puede usar cualquier serie con un crecimiento exponencial, pero se recomienda esta.



Se estima en forma colectiva, en equipo.

Encontrar una **user story canónica**: es el elemento que nosotros vamos a utilizar para comparar. Se elige una historia que será la canónica, y contra esa se comparan todas las demás. Tiene que ser un numero bajo, 1, 2 o 3. No debe superar los 3 puntos.

La mayoría de los equipos solo estima el tiempo de programación -> MAL, esa no es la DOD, definition of done.

3 dimensiones que puede tener una US al momento de asignar una story point.

- Complejidad: cuan dificultosa es la característica por implementar. Baja, media o alta.
- Esfuerzo: se refiere a TRABAJO, no tiempo. Son las horas ideales de trabajo. Sacando descansos, breaks, etc. Cuantas horas reales necesito para construir algo.
- Duda/Incertidumbre: que nivel de desinformación tengo respecto a esta historia. Esta puede ser técnica o de negocio. Es lo mas importante de definir, porque si no se la incertidumbre, no puedo estimar. Y mientras mas incertidumbre tenemos, mas compleja es y más grande será el numero de SP

Story point es un número que homogeniza las tres dimensiones. Puede que una US tenga mucho esfuerzo y poca duda y hay que hacer un balance.

US de consultar se puede dividir en 2

- Consulta simple: sin gastos registrados
- Consulta compleja: tiene filtros, puede fallar.

Cuando hablamos de fallas, no hablamos de que ocurra un error, sino que se tome un camino que no es el ideal. Por ejemplo, si no se ingresa nada en un campo obligatorio, sale un cartel de advertencia, esto no es una falla, sino un camino no ideal.

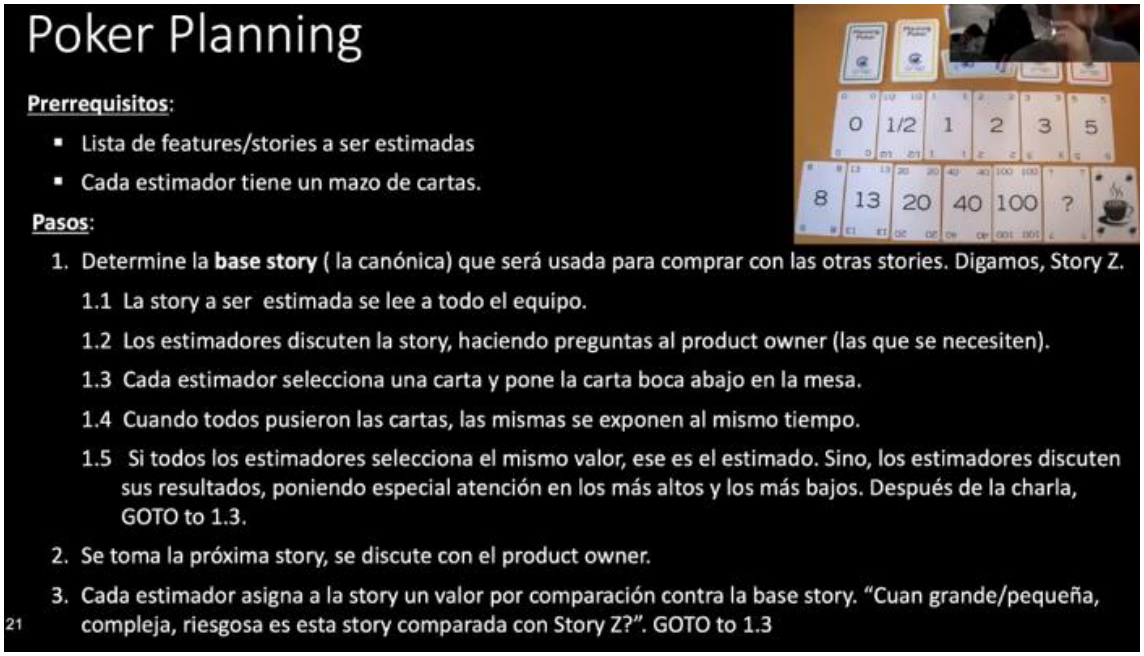
Necesitamos saber el tamaño de lo que vamos a estimar, si es mucho, poco, grande o complejo, etc.

Velocidad

Es una medida del progreso de un equipo. Se calcula sumando el número de Story Points asignados a cada user story que el equipo completa durante la iteración.

- Se cuentan los Story Points de las US que es estan completas, no parcialmente completas.
- La velocidad corrige los errores de estimación.

Póker Planning



Cuando hay mucha duda puede que no sea factible estimar.

MVP → minimal viable product. Característica fundamental del MVP: HIPOTESIS. El MVP se crea para validar una hipótesis. La hipótesis fundamental es que vamos a invertir en un producto que la gente va a querer, que va a usar. Debemos preguntarnos si el producto realmente satisface una necesidad, si vamos a conseguir clientes, si vamos a conseguir la cantidad de descargas que queremos, etc.

Si la hipótesis sale mal validada tengo la posibilidad de hacer modificaciones, es para eso.

La idea es que el producto sea único y tenga valor para el mercado destino.

El proyect manager PM se encarga de las actividades necesarias para gestionar el proyecto. Administra los recursos y gestiona las personas, para poder cumplir con los compromisos que asumió con respecto al proyecto.

Product manager PM: este rol dentro de los equipos agiles se llama product Owner. A veces coinciden, es decir que el PM es el mismo que el PO.

Cuando aparece la necesidad de la tercerización → aparecen las consultoras.

Al decidir qué debe tener el producto desde la primera versión, nace el concepto de MVP

MVF → minimal viable feature

MMF → minimal marketable feature. Características mínimas que debe tener un producto para que yo pueda salir a venderlo, salir a producción, y que la gente lo pueda usar. Muchas veces se confunde con el MVP, pero si no lo estoy usando para validar una hipótesis, ya puedo asegurarme qué eso no es un MVP

A veces el MVP es un MVF.

Un MVP puede ser un video, una presentación de power point. No necesariamente es un producto.

Hay ambigüedades. No existe el siempre y el nunca

Personas	Procesos	Definidos – Empíricos (ágil - lean)
	Proyectos	
	Producto	

Proceso definido: establecer de ante mano todas las cosas que voy a necesitar hacer para poder cumplir con el producto: el orden en que hay que hacerlas, los artefactos que genera cada tarea, que rol hace que tarea. Esta decisión viene definida “por otra persona” y hacer ajustes o cambios al proceso es más costoso

La motivación de quien quiere tener un proceso definido

El proceso ayuda a la gente nueva que entra a la organización, en la generación de cultura organizacional, nos ayudan a relajarnos cuando trabajamos

Si yo ya tengo establecido que tengo que hacer paso a paso, y ya se que me lleva tanto tiempo, y en tales condiciones, puedo saber e inferir

Empirismo: la única experiencia que me sirve es la experiencia de este equipo en este proyecto y en este momento. Ni siquiera puedo traer o me va a servir experiencia del mismo equipo en otro proyecto.

El ciclo de vida del producto dura más que el ciclo de vida del proyecto – tenemos muchos proyectos en un producto

Tipos de ciclo de vida

- Secuencial: cascada

- Iterativo: iterativo e incremental (los procesos empíricos solo funcionan con este)
- Recursivo: espiral

El ciclo de vida es más abstracto que el proceso, te dice en qué orden se ejecutan las tareas, cuanto se ejecuta de cada tarea, en que momento se lleva adelante cada una de las tareas definidas en el proceso. Ayuda a que el proceso que es algo más lineal, se pueda efectivamente ejecutar.