Fichas ágiles

#KnowledgePills

autentia

Índice

Scrum

Herramientas visuales

XP

Coaching

Inception Deck

Artefactos ágiles

Tecnología

Escalado agile

Lean

Kanban

Métricas ágiles

Sesgos cognitivos

Value Stream Mapping - VSM

Agile testing

EBM - Evidence-Based Management

Retrospectivas

Estimaciones ágiles

autentia THINKING WITHYOU

Scrum Master

Su objetivo es actuar como **facilitador, un eliminador de impedimentos**, un **líder que sirve tanto a la Organización, como al Equipo Scrum**. Asegura que el proceso Scrum se usa como es debido.

AYUDA A LA ORGANIZACIÓN

- Liderando y guiando la **adopción de la Agilidad**
- Apoyando a **entender y adoptar Scrum** conforme su teoría, prácticas y reglas.
- **Motivando cambios** que incrementen la productividad del Equipo Scrum.
- Facilitando a las personas externas el **cómo interactuar con el Equipo** Scrum.

AYUDA AL EQUIPO A

- Ser **autoorganizado**.
- Y **multifuncional**.
- Crear **productos de alto valor**.
- **Eliminar impedimentos**.
- Facilitar las **dinámicas Scrum**.

AYUDA AL PRODUCT OWNER A

- Encontrar técnicas para gestionar el **Product Backlog** maximizando el valor.
- Entender cómo **crear elementos** de Product Backlog **claros y concisos**.
- **Entender la planificación** del producto en un entorno empírico.
- Asegurar que el **valor aportado** es el **máximo posible**.
- **Entender y practicar el agilismo**.
- Facilitar las **dinámicas Scrum** que sea necesario.

¿QUÉ NO ES?

- Un policía del Scrum Team.
- Un escriba o secretario.
- El que trae el café al equipo.
- El que preside la reunión.
- Un administrador.
- El líder del equipo.
- El jefe jerárquico del resto.

HABILIDADES NECESARIAS

- Habilidades estratégicas como **visión de la organización**, capacidad de **influencia** y habilidades de **negociación** y **gestión de conflictos**.
- Habilidades de **comunicación** como escucha activa, asertividad, dar y recibir **feedback**.
- Habilidades para la **construcción** y evolución de **equipos de trabajo**.

EVENTOS

EN LAS QUE PARTICIPA ACTIVAMENTE

- SPRINT
- SPRINT PLANNING
- DAILY
- SPRINT REVIEW
- SPRINT RETROSPECTIVE

PREGUNTAS QUE DEBO HACERME

1. ¿Soy accesible para todos los miembros del equipo?
2. ¿He revisado qué bloqueos son recurrentes?
3. ¿El equipo se coordina y conversan entre ellos o están reportando a alguien? ¿Estoy consiguiendo que el equipo se auto-organice?
4. ¿Estoy sugiriendo utilizar todos los mecanismos para que el equipo evolucione a nivel técnico y a nivel de habilidades?
5. ¿Estoy en contacto y coordinándome con otros Scrum Masters y con la práctica ágil de otros equipos de la organización?
6. ¿Estoy contribuyendo con mi práctica Agile al cambio de la organización?
7. ¿Me mantengo actualizado en cuanto a nuevos modelos, herramientas Agile?

STOPPERS

- **Carencia de influencia** y liderazgo hacia otras posiciones jerárquicas.
- Bloqueos con otros equipos que no sean ágiles.
- Que los miembros del equipo **NO entiendan su rol** y sus relación con los demás.

autentia THINKING WITHYOU

Product Owner

Es el responsable de **mantener la visión del producto** que se va a construir **maximizando la cantidad de valor entregado** al finalizar cada iteración. Para mantener esa visión el dueño de producto tiene diferentes interlocutores a los que llamamos *stakeholders* o interesados en el proyecto.

¿CUÁLES SON SUS FUNCIONES?

- Obtener la información necesaria de los stakeholders para garantizar que el **producto desarrollado es el solicitado.**
- Transmite a los **stakeholders** las motivaciones de la metodología y **promueve su implicación.**
- Gestionar el backlog del producto, **definiendo los ítem del Backlog** a nivel de Negocio.
- Es responsable de transmitir de una forma efectiva la visión de producto a todos los implicados y **priorizar el trabajo**, de acuerdo al máximo valor aportado.
- Debe garantizar que el equipo tiene acceso al Backlog, aportando transparencia y claridad sobre cuál es la próxima funcionalidad a desarrollar.
- Debe velar porque todo el **equipo comprenda los ítems del Backlog.**
- Ha de **validar la funcionalidad** desarrollada de acuerdo a los criterios de aceptación que ha definido.
- Debe **conocer los impedimentos** que afecten al equipo y estar informado del desarrollo del Sprint.
- Es el responsable de **controlar** toda la **inversión** relacionada con el proyecto.

¿QUÉ NO ES?

- Un Analista Funcional.
- Un Jefe de Proyecto.
- El jefe del resto.
- Un dictador.

Product Owner

autentia THINKING WITHYOU

HABILIDADES NECESARIAS

- **Identificación de riesgos**, **priorización**, levantamiento de métricas.
- ***Story Telling*** (asociado a Historias de usuario o no).
- Habilidades estratégicas como **visión** de la organización, **conocimiento del producto**, de las tendencias de la **industria/mercado y tecnología**.
- Habilidades de **negociación** y **toma de decisiones**.
- Habilidades de **comunicación** para transmitir ideas y alternativas al cliente, al equipo y a todos los implicados.

EVENTOS

EN LAS QUE PARTICIPA ACTIVAMENTE

- SPRINT PLANNING
- SPRINT REVIEW
- SPRINT RETROSPECTIVE

PREGUNTAS QUE DEBO HACERME

1. ¿Conozco el valor que aporta cada funcionalidad del producto para la organización?
2. ¿Están las necesidades definidas de una forma clara?
3. ¿Están bien definidos los criterios de aceptación?
4. ¿He revisado mi priorización teniendo en cuenta además el feedback del equipo?
5. ¿El backlog es lo suficientemente claro y transparente para el equipo?
6. ¿Estoy transmitiendo los objetivos para cada sprint?
7. ¿Conozco todos los impedimentos que pueden afectar al desarrollo del sprint?
8. ¿Una vez he cerrado una iteración dispongo de todo el feedback y las métricas para priorizar la siguiente?

STOPPERS

- Estar en una **posición** que me **condicione** la **priorización** del producto
- No conseguir que los stakeholders **diferencien entre importante y urgente**
- Que los stakeholders no se involucren de forma que pueda comprender claramente los **beneficios para la organización**
- No tener **capacidad de decisión** sobre varias peticiones que coincidan en importancia y en el tiempo

Development Team

El *development team* son los profesionales encargados de **construir el producto o servicio**, la pieza de valor. Los Equipos de Desarrollo son multifuncionales, contando como equipo con todas las habilidades necesarias para crear un Incremento de producto. Estos equipos deben ser **pequeños** (de entre 3 y 9 personas), **multidisciplinares** —que entre todos los miembros puedan dar servicio al proyecto— y **autoorganizados** —que ellos mismos decidan la mejor manera de desarrollar su trabajo.—

¿CUÁLES SON SON SUS FUNCIONES?

- Es **autoorganizado**. Nadie debe especificar cómo convertir el backlog en incrementos de funcionalidad.
- Es **multifuncional**. Cuenta con todas las habilidades necesarias para crear un incremento del producto, además, **no debe haber especialistas**.
- **El trabajo de cada uno de ellos aporta de forma directa a la entrega de valor.**
- Aunque puede haber especialistas, **el equipo es responsable como un ente completo**.
- Es deseable que se comparta el conocimiento para **evitar centralizarlo** en una única persona.
- Es el **responsable** de la **construcción** del producto.
- **Presenta el resultado** de la iteración al Product Owner.

¿QUÉ NO ES?

- Un grupo de esclavos o robots
- Calculadores de la fecha exacta de entrega
- Un grupo sin objetivos comunes elegido al azar

Development Team

autentia THINKING WITHYOU

HABILIDADES NECESARIAS

- Habilidades para **trabajar de forma coordinada, autónoma y responsable**.
- Habilidades **especializadas** según el rol en el equipo.
- Habilidades de **comunicación y feedback**.

EVENTOS

EN LAS QUE PARTICIPA ACTIVAMENTE

- SPRINT PLANNING
- DAILY
- SPRINT REVIEW
- SPRINT RETROSPECTIVE

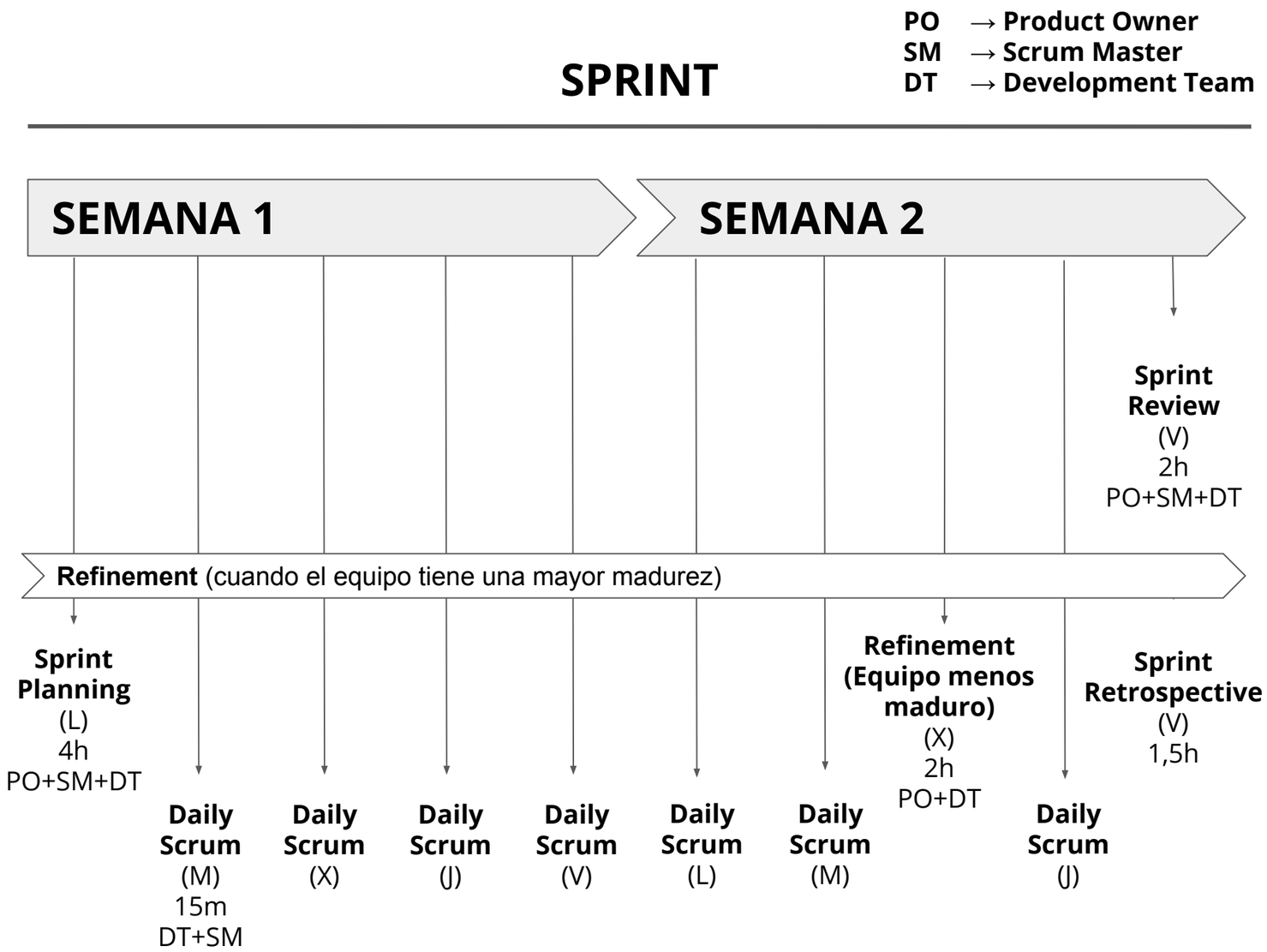
PREGUNTAS QUE DEBO HACERME

1. ¿Cómo estamos trabajando como equipo? ¿Nos estamos coordinando bien?
2. ¿Tenemos acuerdos de equipo, son explícitos y los cumplimos?
3. Del 1 al 10, ¿cuán de autónomos somos para llevar el producto end-to-end?
4. ¿Cuán de participativos somos en cada una de los eventos? ¿Qué necesitamos para serlo más?

STOPPERS

- Que el equipo no cuente con recursos para la entrega de valor.
- Que haya un exceso de dependencias externas.
- Que el Product Owner no pueda resolver las dudas del equipo.

Modelo de Sprint de 2 semanas



Time-box: hablamos de duraciones máximas de los eventos, todo lo que sea menos, si es suficiente, bienvenido.

Errores comunes en los eventos

SPRINT REVIEW

- Querer usarlas para justificar un avance parcial.
- Se eternizan.
- No preparadas por el equipo.
- Que cada uno presente su parte.
- No pedir feedback.

SPRINT PLANNING

- Definición de las necesidades no READY.
- Product Owner no disponible.
- Criterio de DONE no completo.

DAILY SCRUM

- Baja energía / perseguir gente.
- Que se use como una herramienta de supervisión o reporte.
- Hablar de tareas que no estén reflejadas en el panel.
- Entrar demasiado al detalle de los temas.
- Crear conversaciones paralelas.

SPRINT RETROSPECTIVE

- Monotonía.
- No guiadas cada vez por una persona.
- No fijar acciones concretas y responsables para mejorar.
- Que solo se traten temas técnicos o que solo se traten las relaciones entre el equipo.
- No revisar los aspectos identificados en la retrospectiva anterior.

NO cumplir los Time-box establecidos para cada uno de los eventos.

Sprint Planning

Es el evento en el que se planifica el trabajo a realizar durante el siguiente sprint. Deben responderse las dos siguientes preguntas: ¿**qué puede entregarse** en el Incremento resultante del Sprint que comienza? ¿**Cómo se conseguirá** hacer el trabajo necesario para entregar el Incremento?

OBJETIVOS

- Fijar un **objetivo para el sprint** (Sprint Goal)
- **Seleccionar** y **estimar** (esfuerzo) los elementos sobre los que va a trabajar el equipo de desarrollo (Sprint Backlog).

¿CUÁNTO DURA (Time-box)?

- Máx. **8 h** en un sprint de 1 mes
- Máx. **4 h** en un sprint de 2 semanas

MOTIVACIONES

- Repasar y **comprender los ítems del Product Backlog** a desarrollar en esta iteración, y que pasarán a formar parte del **Sprint Backlog**.
- Asegurarse de que lo priorizado cumple con la definición de **Ready**.

¿QUIÉN DEBE PARTICIPAR?

- El **Product Owner**
- El **Equipo de Desarrollo**
- El **Scrum Master**

Daily Scrum

Es la ceremonia/reunión de **planificación y sincronización diaria** entre los miembros del Equipo de Desarrollo, en la que además subyace el plan de trabajo para las **próximas 24 horas**. Para ello se responden las tres preguntas desde el punto de vista del objetivo a lograr en el Sprint: **¿qué hice ayer? ¿Qué haré hoy? ¿Tengo o veo algún bloqueo o impedimento?**

OBJETIVOS

- **Comunicar y hacer explícitos los impedimentos** que existan o que vayan apareciendo.
- Hacer **partícipe a todo el equipo** de la situación actual del sprint y el avance planificado para las siguientes 24 horas. (Un panel físico refuerza este punto).

¿CUÁNTO DURA (Time-box)?

- Máx. **15 min** todos de pie.

MOTIVACIONES

- Mejorar la **sincronización** y el flujo de información entre los miembros del equipo.
- **Fomentar la comunicación** de manera activa.
- Adquirir un compromiso y **disciplina de equipo**.

¿QUIÉN DEBE PARTICIPAR?

- El **Equipo de Desarrollo**
- Recomendable que asistan (no participen) periódicamente el **Product Owner y el Scrum Master**

Sprint Review

El Equipo de Desarrollo explica **qué funcionalidades no se han terminado** y muestra al Product Owner las **funcionalidades que se han terminado y que componen el incremento**. El Product Owner aporta la **visión del mercado y del uso del producto**. El Equipo Scrum determina si se ha alcanzado el **objetivo del Sprint**.

OBJETIVOS

- **Inspeccionar** el Incremento y las mejoras construidas en esta iteración.
- Recibir **feedback** que permita al equipo orientarse hacia el producto final requerido por el usuario.

¿CUÁNTO DURA (Time-box)?

- Máx. **4h** en un sprint de 1 mes
- Máx. **2h** en un sprint de 2 semanas

MOTIVACIONES

- Comprobar que la funcionalidad solicitada se ha **desarrollado según los criterios de aceptación** propuestos.
- **Adaptar el Product Backlog** si fuese necesario.

¿QUIÉN DEBE PARTICIPAR?

- El **Product Owner**
- El **Equipo de Desarrollo**, y en caso necesario, los **stakeholders** pueden ser invitados a participar.
- El **Scrum Master** puede asistir.

Refinement (no oficial de Scrum)

El Refinamiento se consideraba previamente una **ceremonia** en la que se depuran los *Product Backlog Items* para que sean completamente claros y cuenten con toda la información necesaria para su desarrollo. Lo ideal es que este refinamiento **se produzca de forma continua** durante la iteración, sin recurrir a fijar una ceremonia específica. Originalmente se denominaba como *Grooming*, pero debido a las connotaciones negativas de este término se ha sustituido en recientes revisiones por *Refinement*.

OBJETIVOS

- Obtener un sprint backlog **priorizado**.
- Alcanzar el estado de **READY** para cada PBI.

¿CUÁNTO DURA (Time-box)?

- Máx. **10%** del tiempo del equipo en una iteración.

MOTIVACIONES

- Detectar los posibles **impedimentos o dependencias** que puedan existir, y madurar los ítems del backlog para que no se produzcan paradas o bloqueos durante el desarrollo.

¿QUIÉN DEBE PARTICIPAR?

- El **Product Owner**
- El **Equipo de Desarrollo**.

Sprint Retrospective

La *Sprint Retrospective* tiene lugar después de la *Sprint Review* y antes de la siguiente *Sprint Planning*. Es una oportunidad muy valiosa para el **Equipo Scrum** de **inspeccionarse a sí mismo** y crear un **plan de mejora** para el siguiente Sprint, es decir está **enfocada en el equipo y su proceso**, y no en el producto como otros eventos.

OBJETIVOS

- Identificar las **fortalezas** y las **áreas de mejora** tanto en los aspectos técnicos como en las relaciones del propio equipo.
- Obtener **acciones concretas** y responsables de su seguimiento.

¿CUÁNTO DURA (Time-box)?

- Máx. **3 h** en un sprint de 1 mes
- Máx. **1,5 h** en un sprint de 2 semanas

MOTIVACIONES

- Realizar un ciclo de **inspección y adaptación** sobre el propio equipo y el proceso.
- Fomentar la **mejora continua** como parte del proceso natural de un equipo ágil.

¿QUIÉN DEBE PARTICIPAR?

- El **Product Owner**
- El **Equipo de Desarrollo**.
- El **Scrum Master**.

Sprint Retrospective

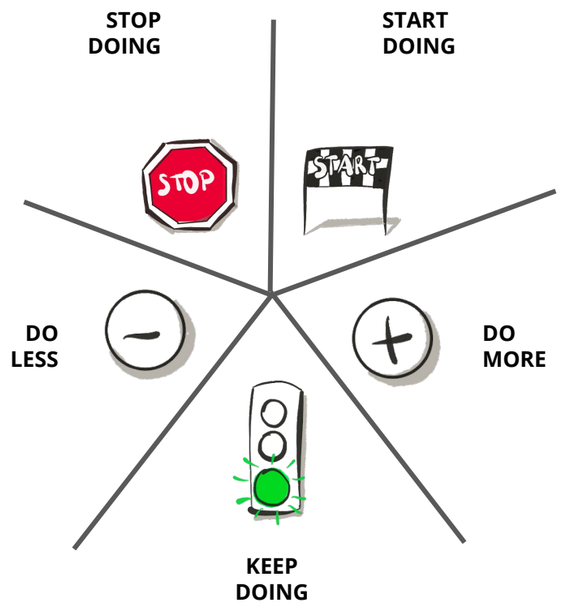
autentia THINKING WITHYOU

Hay múltiples modelos para llevar a cabo una retrospectiva, y es importante emplear diferentes para que no se convierta en algo rutinario y monótono para el equipo. Podemos usar fuentes de referencia para realizarlas como **Retromat** o **Fun Retrospective**.

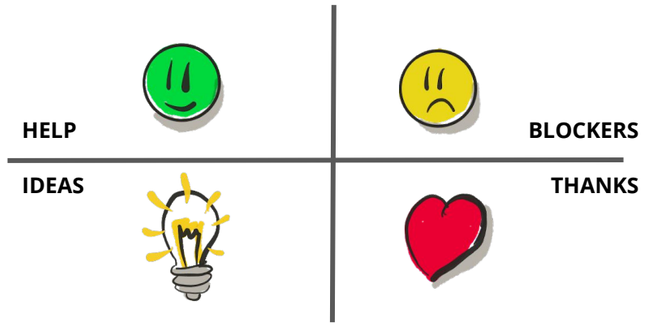
FASES

1. **Preparar el escenario:** recordar el objetivo de la reunión, y el tiempo máximo que puede durar.
2. **Recabar datos:** analizar el estado de ánimo del equipo, presentar hechos relevantes que han ocurrido en la iteración, identificar items terminados, no terminados, bloqueos, decisiones relevantes tomadas, etc.
3. **Generar entendimiento:** analizar los hechos y responder a las preguntas ¿qué ha pasado? ¿Por qué ha pasado? ¿Cómo lo podemos mejorar?
4. **Decidir qué hacer:** de todo lo identificado, elegir aquello sobre lo que queremos trabajar para incorporarlo en la siguiente iteración.
5. **Cerrar la retrospectiva:** marcar responsables para seguir cada una de las acciones que hemos elegido. Motivar al equipo agradeciendo el trabajo, la participación y la transparencia.

MODELO ESTRELLA DE MAR



MODELO CLÁSICO



Chore

Los **Chores** son **items** que son **necesarios** pero que **no aportan** un **valor de negocio directo** u obvio al cliente.

DEFINICIÓN

Un Chore es **algo que se necesita hacer antes o durante el desarrollo de un item** de funcionalidad que puede afectar a una o a varios product backlog items al mismo tiempo, **trabajos esenciales** para establecer entornos o de soporte para las funcionalidades, no aportan funcionalidad directamente, pero la construcción de esta **depende de ellos**. Algunos ejemplos podrían ser: “preparar un entorno de desarrollo” o investigar “por qué un determinado conjunto de pruebas tarda demasiado”.

Los Chores son un **coste para el negocio**. Utilizan recursos sin contribuir a la velocidad del equipo pero son necesarios. Los Chores tampoco son trabajos básicos, en general se necesita una capacidad y especialización similar a la necesaria para desarrollar cualquier otra funcionalidad.

Dicho esto, la importancia y necesidad es obvia pero:

- Los Chores pueden representar **deuda técnica** o **puntos de dependencia** con otros equipos. Pueden afectar a muchas funcionalidades y ser una fuente de una i**mportante carga de trabajo**.

- Los Chores **emergen a lo largo del tiempo** según la necesidad y ese es el momento de acometerlos **no son actividades que se puedan o deban anticipar**.

Spike

Spike, es un término que respondería a la **pregunta**: ¿Qué es lo **más simple** que podemos **programar** para **asegurarnos** que vamos por el **buen camino**?.

DEFINICIÓN

Es decir, un Spike sería el programa más sencillo que nos permite explorar diferentes opciones o soluciones. Es una tarea orientada a contestar una pregunta o conseguir una información más que a producir un producto entregable.

A veces un *product backlog item* no se puede estimar correctamente hasta que el equipo de desarrollo haga algún trabajo de investigación para resolver o contestar una pregunta técnica o un problema de diseño. La solución es crear un “spike”, un trabajo o actividad, cuyo objetivo es conseguir esa respuesta o solución.

AVANZADO

En SAFe, por ejemplo, los Spikes se definen como un *item* facilitador de un tipo de exploración. Sus orígenes están en XP y representan actividades del tipo de diseño, investigación, exploración y prototipado. El objetivo no es otro que ganar el conocimiento necesario para reducir el riesgo de un planteamiento técnico, entender mejor un requerimiento o mejorar la fiabilidad de la estimación de un funcionalidad.

Como cualquier otro *item*, los Spikes deben ser estimados y demostrados al final de su Sprint.

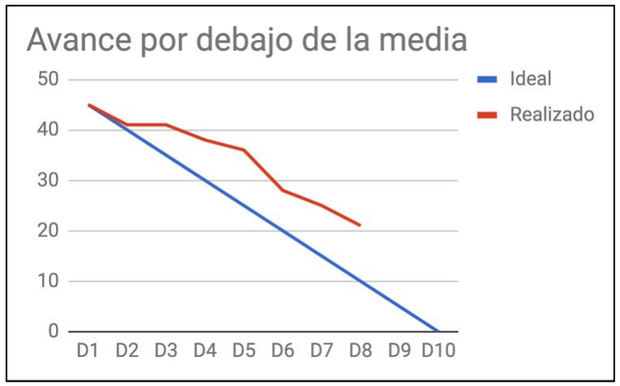
Burn down chart

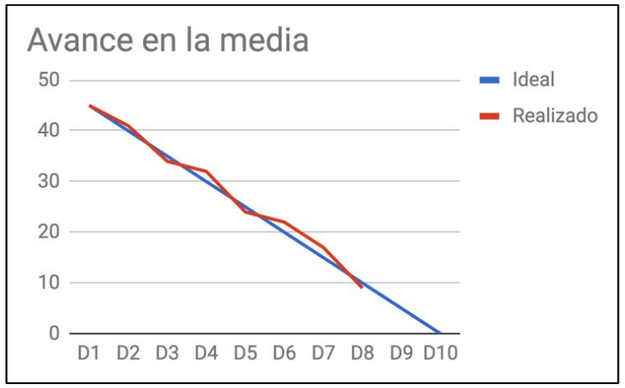
El objetivo de este gráfico es mostrar el **trabajo** (esfuerzo) **que queda por realizar** en un periodo de tiempo concreto (iteración). En los equipos de desarrollo ágiles, el gráfico Burn Down se suele **aplicar por sprint** y ayuda al equipo a la **visualización** del **esfuerzo realizado** y el que queda por realizar hasta la finalización del sprint.

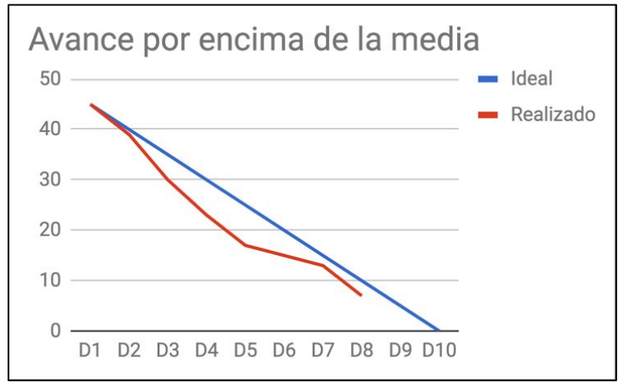
UTILIZACIÓN

- En el eje **X se representa el tiempo**, bien en semanas, en días, etc, dependiendo de la duración de la iteración.
- En el eje **Y se representa el esfuerzo** a realizar, la mejor unidad para hacerlo son los puntos de esfuerzo.
- Si el gráfico representa un sprint, entonces la Y debería ser los **puntos totales de esfuerzo** para este Sprint, y la X la **duración del Sprint**.
- Se han tomado como referencia para el ejemplo un Sprint de 2 semanas (10 días) y una estimación de 45 puntos de esfuerzo a realizar por el equipo.

- Uniendo ambos puntos, obtendremos la pendiente que representa el **esfuerzo medio a realizar por día**. Es decir, el avance teórico del consumo de *items* y tareas a lo largo del sprint (ideal).
- El gráfico se usa colocando una marca **(un punto) en la coordenada** que corresponde al **día** dentro **del sprint** y a los **puntos** de esfuerzo **que quedan** al restar, los realizados hasta el momento, menos el total de puntos del sprint. Tras esta acción, se debe unir esta coordenada con la coordenada representada anteriormente con una línea.
- Si estamos **por encima de la pendiente**, el sprint va avanzando **por debajo de la media**, si estamos **por debajo de la pendiente**, el avance es **por encima de la media**.







Velocity

La velocidad **no es un objetivo**, **es una consecuencia**, se puede denominar como la cantidad de puntos de esfuerzo que el equipo ha sido capaz de finalizar en un sprint.

¿PARA QUÉ NOS SIRVE?

- La Velocidad **no es un valor que permita comparar la productividad entre equipos**, ya que depende de numerosos factores, como las **tecnologías** que se emplee (un equipo puede trabajar con Java y otro con Python), la complejidad del **producto** en el que trabajen (uno puede ser un producto de terceros que adaptamos, otro un desarrollo desde cero in-house), la complejidad del **negocio** (no es lo mismo calcular una órbita de reentrada, que la franja de entrega de un paquete), las **dependencias** (hay desarrollos fuertemente dependientes de terceros que pueden impactar)...
  Básicamente estaríamos comparando "peras" con "manzanas".

- Dentro de un mismo equipo sí que nos puede ayudar para **analizar la tendencia** que tiene y aunque sea de forma estimativa, comparar la velocidad, con puntos de esfuerzo del backlog para hacer una proyección (con todas las consideraciones que una proyección puede tener) en un mismo equipo.
- Debe ser uno de nuestros objetivos **modificar el criterio de medición**, de productividad a **valor entregado**.

Definition Of Ready (No forma parte de Scrum)

Aunque no forma parte de la Guía Scrum muchos PO y equipos de desarrollo están de acuerdo en utilizar el término **Definition of Ready (DoR)**. El Definition of Ready es una *lista de comprobación que captura el entendimiento compartido del Product Owner y el Equipo de Desarrollo en forma de criterios que cada ítem del Product Backlog debe reunir para que se pueda empezar a trabajar en él.*

UTILIZACIÓN

- **Antes de que el trabajo pueda planificarse en un Sprint, los elementos más relevantes del Product Backlog deben estar terminados**, entendiéndose por ello, estar lo suficientemente bien descritos y comprendidos.
- **El Equipo de desarrollo debe comprender su alcance** lo suficiente como para poder planificarlo en un Sprint y estimar su implementación para que se pueda alcanzar el Objetivo del Sprint.
- Es aquí donde entra en juego el **refinamiento**, sesión a la que no deberían dedicar más del 10% de su tiempo y en **donde se agregarán detalles a los ítems, se modificarán estos**, se añadirán nuevos ítems pero sobretodo se desglosarán los más grandes en ítems más pequeños, detallados y estimados.
- En esta sesión **los ítems podrán ser refinados sucesivas veces** antes de dejar de forma adecuada el Product Backlog.
- Disponer de un DoR ayuda a **reducir los riesgos del Sprint Planning** ante tareas que el equipo no entiende o no se pueden abordar por motivos técnicos.

EJEMPLOS

- Cada DoR muestra las expectativas en conjunto del PO y el equipo de desarrollo en diferentes momentos. **Actualizaron el DoR según lo vieron necesario.**

Team Avengers, 2018

- Sistemas debe montar el entorno de desarrollo.
- Se definen los Criterios de Aceptación más relevantes en la sesión de planificación.
- La Historias de Usuario deben estar registrada en JIRA.
- Las Historias de Usuario deben estar estimadas en puntos de Historia.

Team Avengers, 2020

- La Historia de Usuario está refinada y estimada.
- Cada miembro del equipo entiende la Historia de Usuario y el valor que aporta a negocio.
- Está documentada en Confluence.
- Cada Historia de Usuario debe estar acompañada por los bocetos e instrucciones del dpto. de UX/UI.

Definition of Done (DoD)

En un equipo Scrum, tanto el Equipo de Desarrollo como el Product Owner reflejan su entendimiento compartido de cuándo algo está terminado (Done) en el **Definition of Done**.

UTILIZACIÓN

- **Según la Guía Scrum:** cuando un elemento de la Pila del Producto (Product Backlog) o un Incremento se describe como “Terminado”, todo el mundo debe entender lo que significa “Terminado” (“Done”). **Aunque esto puede variar significativamente para cada Equipo Scrum**, los miembros del equipo deben tener un entendimiento compartido de lo que significa que el trabajo esté completado para asegurar la transparencia. Esta es la definición de “Terminado” (“Done”) para el Equipo Scrum, y **se utiliza para evaluar cuándo se ha completado el trabajo sobre el Incremento de producto**.
- Esta misma definición **guía al Equipo de Desarrollo** en saber cuántos elementos del Product Backlog puede seleccionar durante la Planificación del Sprint (Sprint Planning).
- **Si la definición de “Done” para un incremento es parte de las convenciones,** estándares o guías de la organización de desarrollo, **al menos todos los Equipos Scrum deben seguirla.**

EJEMPLOS

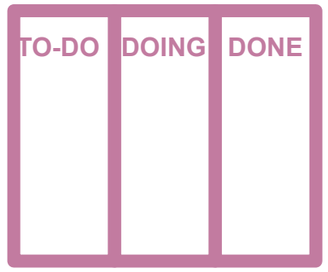
- Cada DoD muestra las expectativas en conjunto del PO y el Equipo de Desarrollo en diferentes momentos. **Actualizaron el DoD según lo vieron necesario.**

Team Rocket, 2018

- Se cumplen todos los criterios de aceptación.
- La calidad del código es superior al 85% en SonarQube.
- Pruebas Automatizadas realizadas en Selenium.
- Pruebas Manuales realizadas.
- Se contempla un registro de todos los cambios.
- Existe una demo probada en el entorno de pruebas.

Team Rocket, 2020

- La documentación está actualizada.
- Existe un ejemplo escrito en Angular.
- Se ha hecho la publicación en Twitter de la actualización.
- Resueltos todos los Issues de Github.
- El código ha sido versionado y mergeado.



Kanban

Es una **técnica** que se creó en **Toyota** para optimizar la producción de automóviles. Se utiliza para gestionar cómo se van completando las tareas y avanzando el **trabajo** de una manera **visual** a través del movimiento de **tarjetas**.

QUÉ ES

- Una herramienta.
- Un método para mejorar la entrega de valor.
- Una metodología para la gestión del cambio.

PRINCIPIOS DE GESTIÓN DEL CAMBIO

- Empieza donde estés.
- Establece cambios evolutivos.
- Promueve el liderazgo en todos los niveles.

LAS 6 PRÁCTICAS

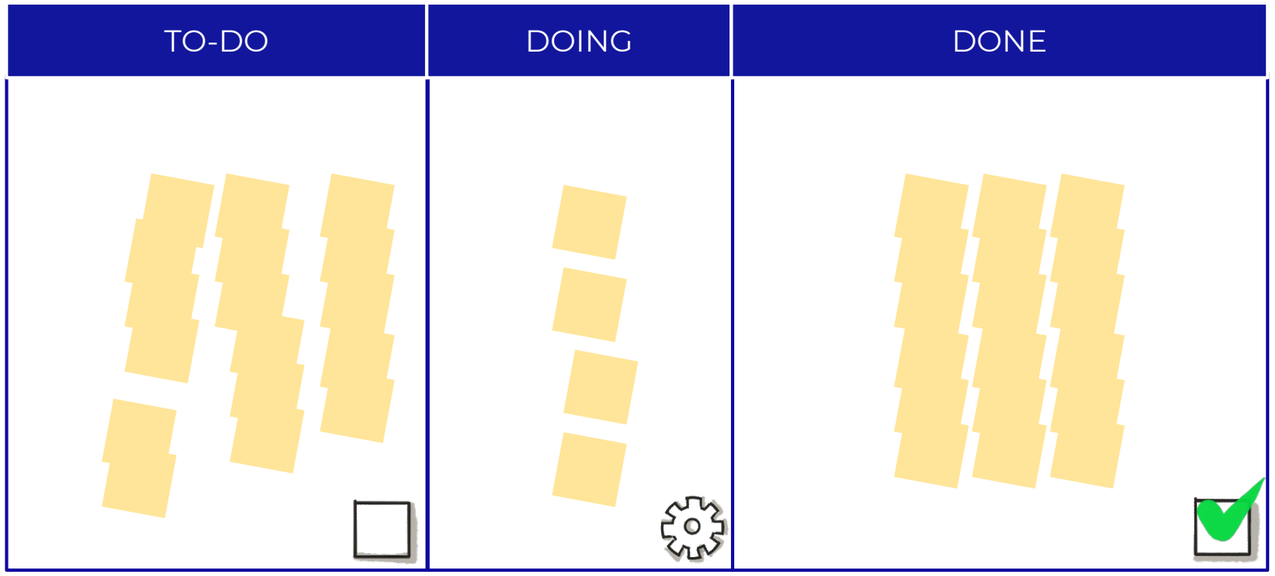
1. Visualiza el trabajo.
2. Limita tu WIP (work in progress).
3. Gestiona el flujo.
4. Haz explícitas las políticas.
5. Práctica la mejora continua.
6. Implementa ciclos de *feedback*.

QUÉ NO ES

- Un procedimiento.
- Un proceso.

Kanban

autentia THINKING WITHYOU



TEMAS/ HISTORIAS	TAREAS PENDIENTES	EN PROCESO				
		IMPEDIDAS	N DÍAS	MAÑANA	HOY	TERMINADAS
		IMPEDIDAS				

Business Model Canvas

Es una herramienta diseñada por Alex Osterwalder destinada a comprender y a **trabajar modelos de negocio** desde la **reflexión estratégica** de cada uno de sus elementos: relaciones, costes, propuesta de valor, impacto, etc. Permite tener una **foto visual y única** de todos los aspectos importantes relacionados con un modelo **de negocio, proyecto o producto**.

COMPONENTES DE UN BUSINESS MODEL CANVAS

- **Alianzas**
  - ¿Qué alianzas serán necesarias para ejecutar el modelo de negocio?
- **Actividades**
  - ¿Qué actividades te permitirán entregar a tu cliente la propuesta de valor?
- **Recursos**
  - ¿Qué recursos necesitas para desarrollar tu actividad?
- **Propuesta de valor**
  - ¿Qué problema resuelves en tus clientes? ¿Qué te diferencia del resto?
- **Relación**
  - ¿Cómo te relaciones con tus clientes?
- **Canales**
  - ¿Cómo entregarás tu propuesta de valor a cada segmento de cliente?
- **Segmentos de clientes**
  - ¿Cuáles son tus segmentos de clientes más importantes? ¿Cómo se relacionan?
- **Estructura de costes**
  - ¿Cuáles son tus costes? ¿Cómo podrías optimizar los costes fijos y variables para conseguir un modelo de negocio escalable?
- **Flujo de Ingresos**
  - ¿Cuál es tu estrategia de ingresos?

Business Model Canvas

autentia THINKING WITHYOU

ALIANZAS / SOCIOS CLAVE

ACTIVIDADES CLAVE

RECURSOS CLAVE

PROPUESTA DE VALOR / VENTAJAS DIFERENCIALES

RELACIÓN CON LOS CLIENTES

CANALES

SEGMENTOS DE CLIENTES

COSTES

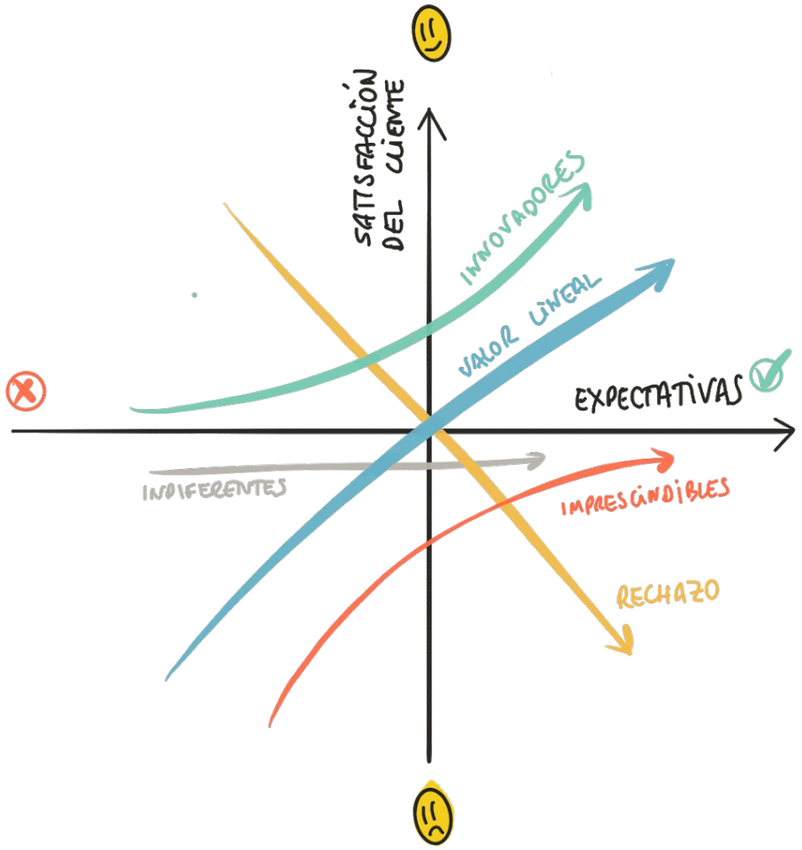
INGRESOS

Modelo Kano

Es una de las técnica de priorización de requisitos propuesta por Noriaki Kano, y que consiste en seleccionar aquellos requisitos o funcionalidades que aporten un mayor valor al usuario, frente a otros criterios que son accesorios. Los criterios de mayor valor son considerados obligatorios, es decir, el producto o servicio no se puede llevar a cabo si incumple alguno de estos requisitos. Esta técnica nos ayudará a enfocar el desarrollo de forma más eficiente.

CRITERIOS MODELO KANO

- **Innovadores:** atributos que los clientes no se esperan y resultan en una gran satisfacción si se consiguen. Por ejemplo, cuando reservamos en un hotel, que nos llame una persona para preguntarnos por nuestras preferencias para el desayuno.
- **Valor lineal:** atributos que, bien conseguidos, aumentan la satisfacción de nuestro cliente. Por ejemplo, que el proceso de reserva en la web sea sencillo y nos envíen un e-mail de confirmación agradeciendo la confianza.
- **Imprescindibles:** necesidades básicas. Atributos que espera nuestro cliente y que conducen a una enorme insatisfacción si no se consiguen. Por ejemplo, que en el proceso de reserva no podamos escoger el día o el tipo de habitación que queremos.
- **Indiferentes:** el cliente no está interesado en ellos. Un exceso de los mismos puede conducir a una insatisfacción. Por ejemplo, que el e-mail de la reserva incluya una guía turística del lugar.
- **Rechazo:** aquellos elementos que se incorporan a la solución pero que no son demandados por el usuario y le suponen una frustración. Por ejemplo, que al finalizar la reserva la web nos lleve automáticamente a una página para alquilar coches y reservar restaurantes y nos pida registrarnos.

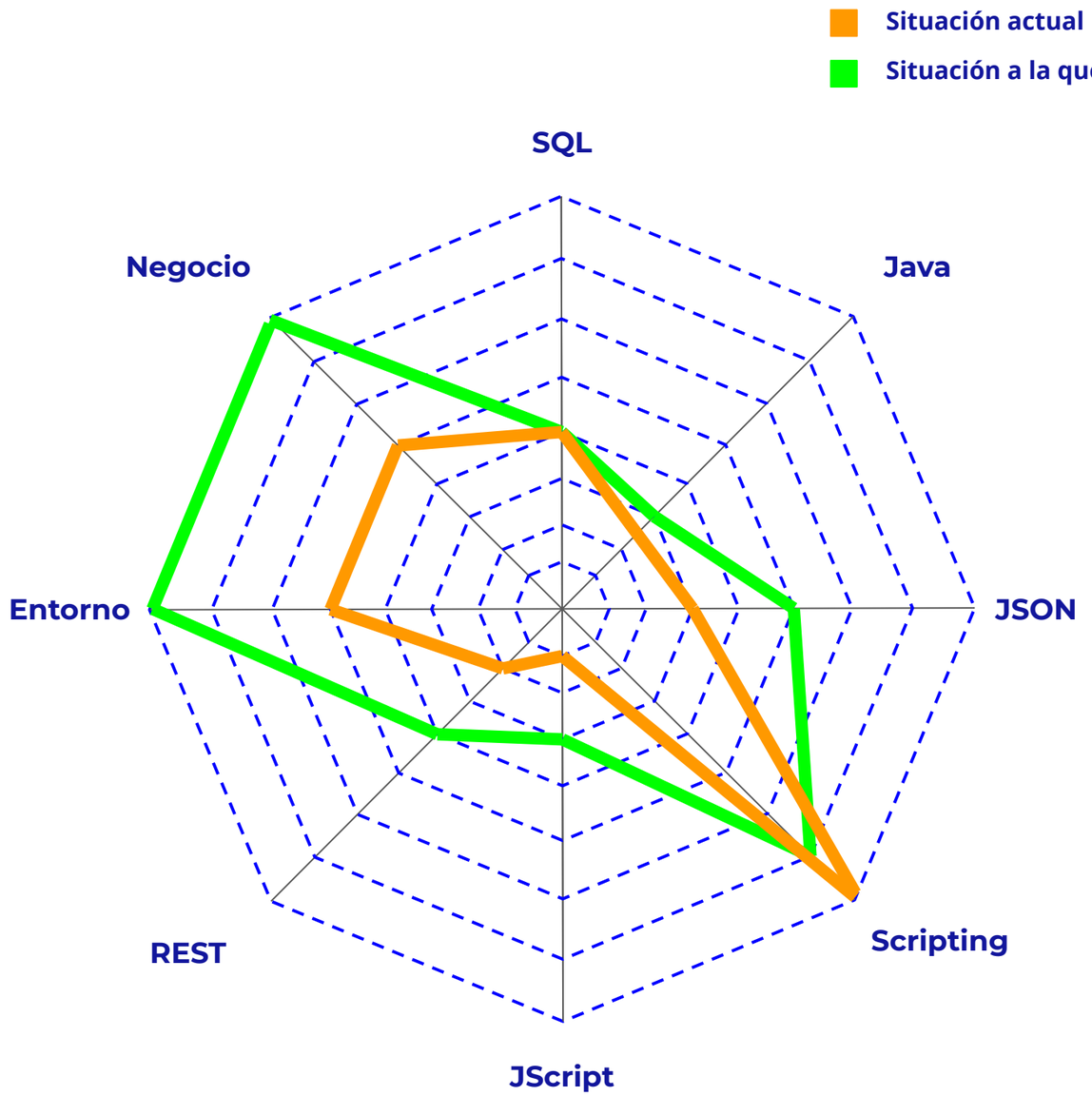


EJEMPLO: ¿Qué esperamos cuando reservamos un hotel?

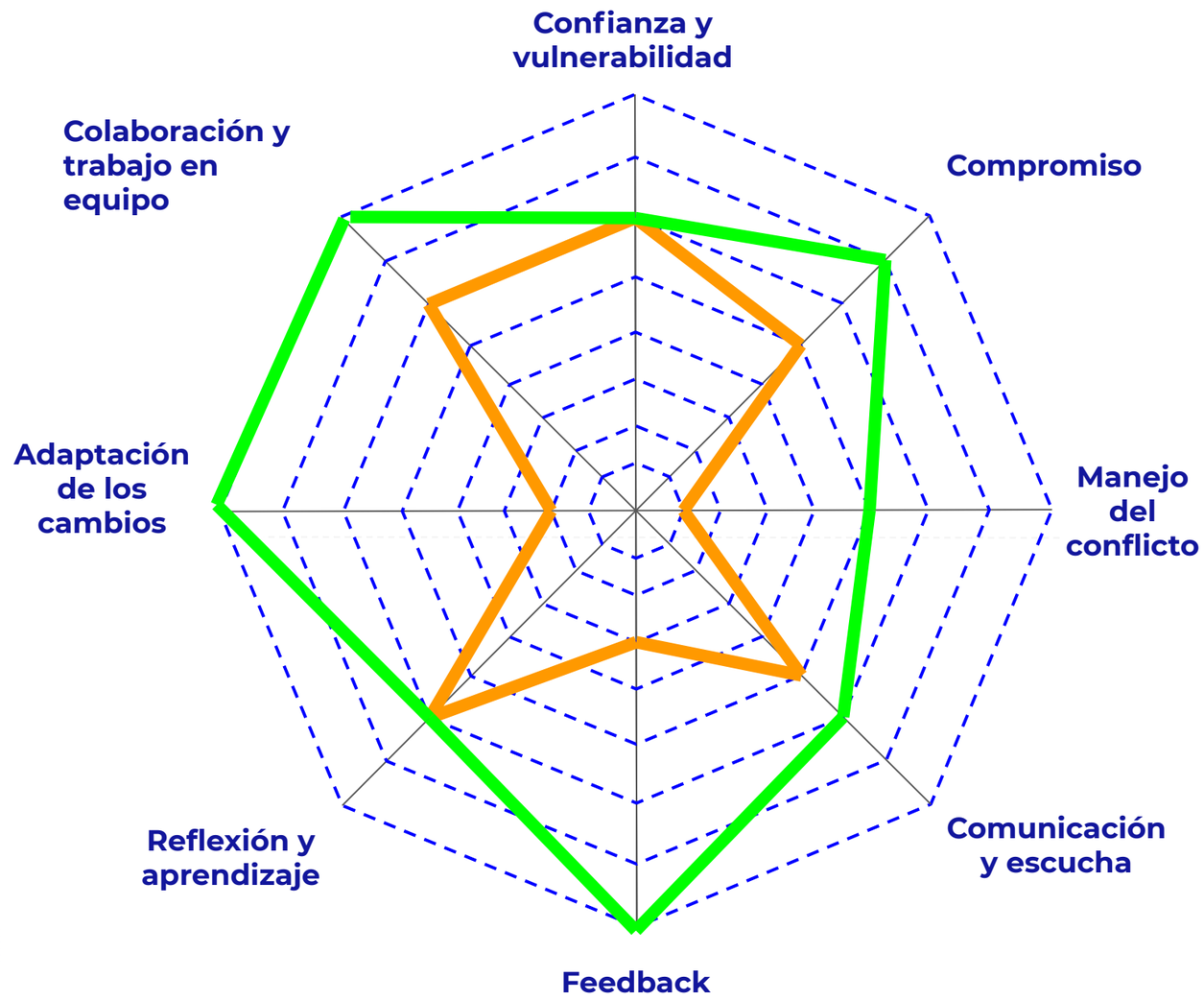
Diagrama de araña

Este tipo de diagrama permite de una forma muy sencilla elaborar una representación visual de las **habilidades como equipo**. Gracias a esta representación podemos identificar las áreas **más críticas que debemos potenciar**.

Dependiendo de la desviación del **estado actual** respecto a la necesidad, podemos identificar diferentes prácticas que aplicar **para alcanzar ese nivel, como Pair Programming, Talleres, Formación externa...**



Ejemplo de criterios técnicos para la evaluación de Equipo



Ejemplo de habilidades para la evaluación de Equipo

Impact Mapping

Es una herramienta de planificación y estrategia que nos permite concentrar los esfuerzos en lo realmente importante, haciéndonos las preguntas correctas para identificar qué es necesario conseguir y cómo hacerlo. Se utiliza en la fase de inception o lanzamiento de un proyecto para enfocar objetivos.

CONTENIDO Y PREGUNTAS DEL IMPACT MAPPING

¿Por qué?

- Por qué estamos haciendo esto, cúal es el objetivo de negocio del proyecto.

¿Con quién?

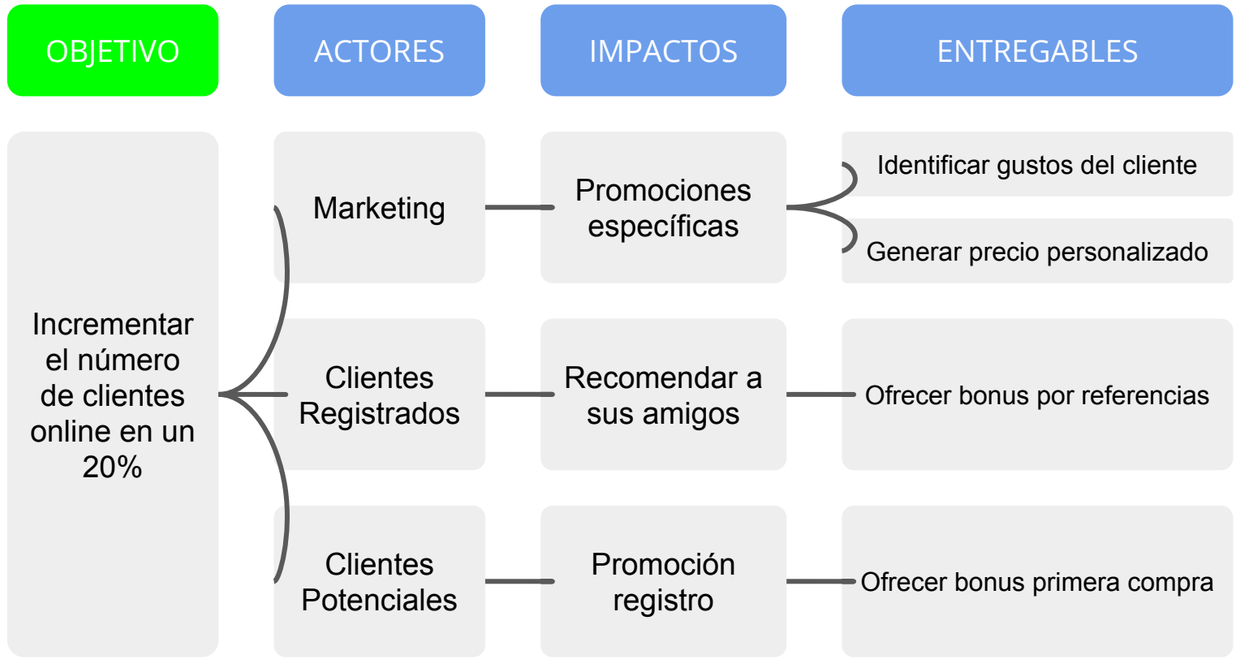
- Qué personas pueden lograr el objetivo.
- Qué personas se verán beneficiadas por cumplirlo.
- Qué personas podrían obstaculizar.

¿Cómo?

- ¿Cómo ayudarán los actores identificados a llegar al objetivo?
- ¿Cómo el objetivo una vez cumplido, impactará en su día a día?

¿Qué?

- ¿Qué podemos hacer como organización o equipo para apoyar los cambios necesarios para lograr la meta deseada?



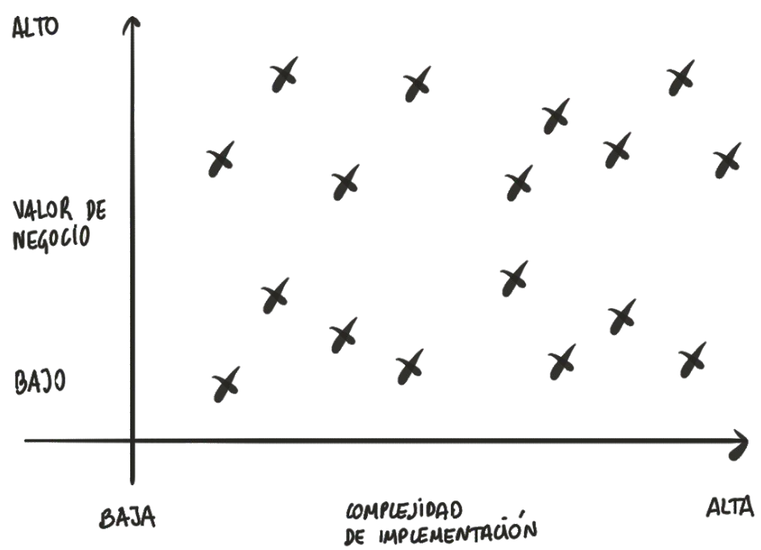
Valor vs. Complejidad

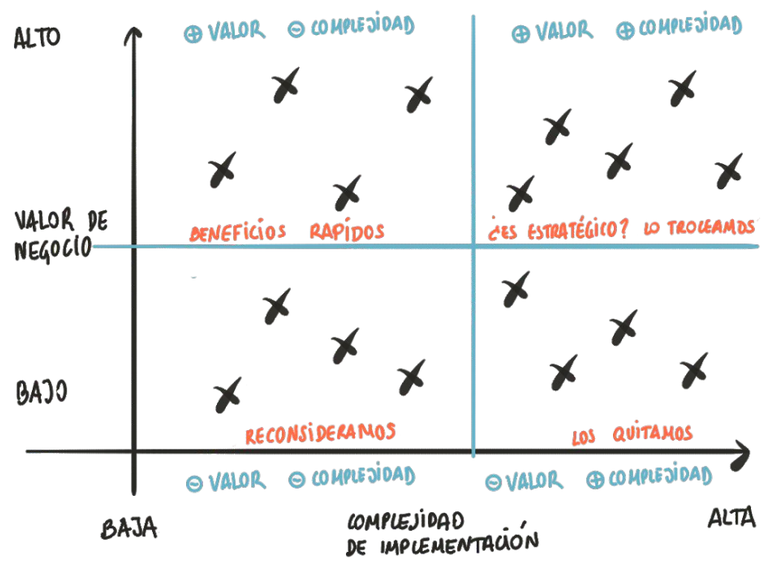
Decidir qué es lo más importante en lo que vamos a trabajar como equipo puede ser una ardua tarea. Siempre hay más peticiones e ideas de las que se pueden acometer, distintas prioridades, y muy diferentes aportes que compiten por llevarse a cabo. Esta sencilla herramienta puede resultar muy útil para que surjan las conversaciones necesarias para la priorización.

UTILIZACIÓN

- La primera premisa a tener en cuenta es que **cada aporte conlleva un valor** para el negocio de forma inherente, **y un nivel intrínseco de complejidad** al llevarlo a cabo. Con esta herramienta la prioridad de cada elemento viene dada por el valor compuesto de ambas variables, siendo recomendado realizar el ejercicio sobre un panel visual, que permita hacer emerger las conversaciones sobre **qué elementos tienen más valor o más complejidad y por qué**.

- Dividiendo el gráfico en un **cuadrante de 2x2** obtenemos un gráfico con cuadrantes de relevancia que **apoyan la toma de decisión**, recomendando atacar primero los elementos del primer cuadrante, pensar si podemos dividir los elementos complejos del segundo, replantear la necesidad de los elementos del tercer cuadrante, y descartando los elementos del cuarto.





Cost of delay / Coste del retraso

Es un modelo que ayuda a examinar el coste para el negocio si una característica en particular se retrasa, sirviendo como guía para ordenar el trabajo. El análisis del coste del retraso (incluso realizado de forma cualitativa) nos ayuda a centrar la atención en el valor de negocio de las funcionalidades, y en cómo su valor varía con el tiempo, por lo que comprenderlo resulta esencial si quieres maximizar el flujo de valor a tus clientes.

UTILIZACIÓN

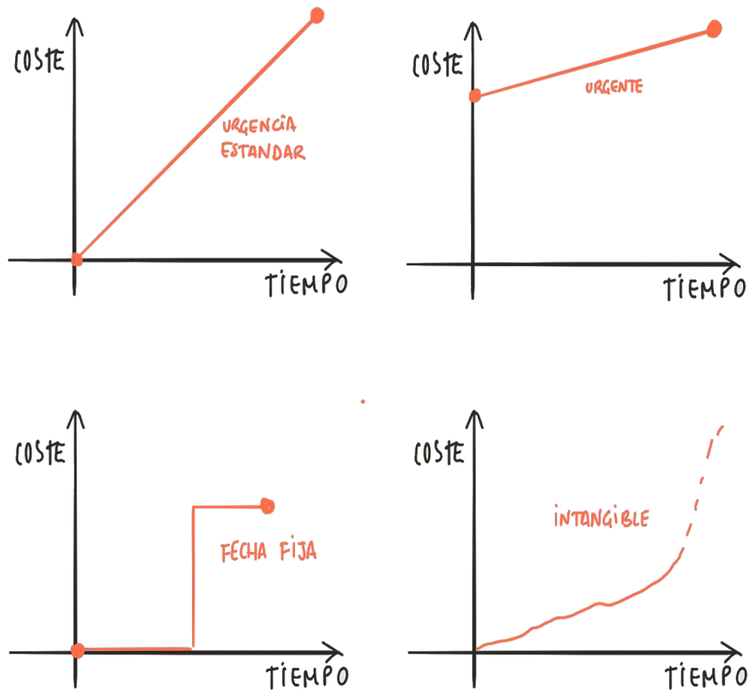
Los tres principales elementos que contribuyen al *cost of delay* son:

- **Valor para los usuarios o el negocio** - ¿Preferimos esto sobre lo otro? ¿Cuál es el impacto en el beneficio de nuestro negocio? ¿Hay algo potencialmente adverso o sanción en caso de retraso?
- **Criticidad del tiempo** - ¿Cómo pierde el valor para el usuario o el negocio a lo largo del tiempo? ¿Existe alguna fecha límite o vencimiento? ¿Esperarán a que terminemos o buscarán otra solución? ¿Hay hitos en el camino crítico afectados por esto?
- **Valor derivado de la reducción del riesgo o de la generación de oportunidades** - ¿Qué más le aporta esto a nuestro negocio? ¿Reduce el riesgo de esta o de futuras entregas? ¿Hay valor en la información que vamos a recibir? ¿Abrirá esta característica nuevas oportunidades de negocio?

Más aún, estando en un flujo continuo y debiendo tener un backlog suficientemente grande de donde elegir, no nos deberíamos preocupar de los números absolutos. Simplemente podemos **comparar de manera relativa** los elementos del backlog entre sí, utilizando la serie Fibonacci modificada que se utiliza en el planning poker.

FÓRMULA CoD

Cost of Delay = user/business value + time criticality + risk reduction and/or opportunity enablement



Weighted Shortest Job First (WSJF)

Weighted Shortest Job First es un modelo de priorización de tareas pensado para producir rápidamente el máximo beneficio, por lo que da más valor a los elementos más cortos y que más aporten. El valor de WSJF se estima como el Cost of Delay dividido por el tamaño de la tarea.

CARACTERÍSTICAS

WSJF se utiliza para priorizar el backlog calculando el Coste del retraso relativo y el tamaño del trabajo, acorde a la siguiente fórmula:

WSJF = Coste del retraso / Tamaño del trabajo

El framework de escalado SAFe propone una simplificación del cálculo del Coste del retraso basado en la percepción cualitativa, de este modo el cálculo simplificado del Coste del retraso calcula como:

Coste del retraso =
Valor para usuario/negocio + Criticidad en el tiempo + Capacidad de crear oportunidad/reducir riesgo

Haciendo uso de estimaciones relativas podemos optar por dar valor a cada uno de los términos de la fórmula utilizando los siguientes valores de la escala de Fibonacci:

1 2 3 5 8 13 21

De esta forma simplificamos el cálculo del valor del WSJF, enfocando las conversaciones necesarias para la priorización del backlog en los ejes propuestos, permitiendo tomar decisiones informadas sobre por qué se realiza determinado trabajo antes que otro, y facilitando también la comunicación de prioridades a nuestros stakeholders.

$$\text{WSJF} = \frac{\text{Valor para usuario/negocio + Criticidad en el tiempo + crear oportunidad/reducir riesgo}}{\text{Tamaño del trabajo}}$$

MoSCoW

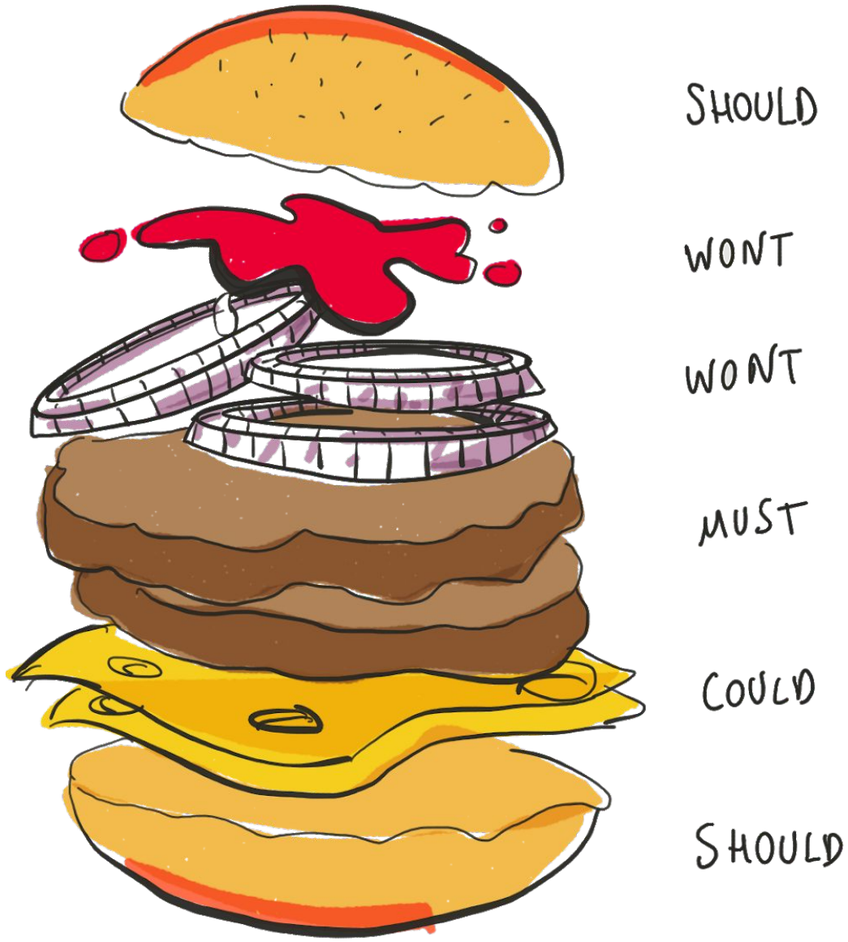
Técnica de priorización basada en la clasificación de funcionalidades de un producto o servicio con respecto a las expectativas de los clientes. El término es una regla mnemotécnica basada en un acrónimo formado por la primera letra de cada categoría.

UTILIZACIÓN

- La técnica consiste en clasificar los elementos que se consideran fundamentales para un producto o servicio frente a otros criterios que no son tan relevantes. Los criterios de mayor valor son considerados obligatorios para poner el producto o servicio en el mercado.

CATEGORÍAS MoSCoW

- **Must have** - Características obligatorias para el producto o servicio.
  Por ejemplo, que se oiga la música cuando demos al play en un reproductor.
- **Should have** - No obligatorias, pero deberían estar en la medida de lo posible.
  Por ejemplo, que el reproductor tenga un pequeño mando en los auriculares.
- **Could have** - Elementos adicionales, deseables aunque no se esperen.
  Por ejemplo, que el reproductor te recomiende canciones según tus preferencias.
- **Won't have** - Mejor si no encontramos estas características.
  Por ejemplo, que tenga estar continuamente enchufado a la corriente.



User Persona (arquetipo)

Es una representación de tu cliente/usuario ideal que realizamos en base a estudios de mercado y a la información que ya tenemos de él/ella. Para ello elaboramos una plantilla con la que tratamos de describir al cliente/usuario de nuestro producto o servicio, así como sus necesidades. Esta técnica nos servirá para identificar a los personajes de nuestras funcionalidades y al "humanizarlos", nos resultará más fácil empatizar con ellos y ajustarnos a sus necesidades reales.

¿QUÉ INFORMACIÓN RECOGEMOS?

- La información que recojamos de nuestros clientes/usuarios dependerá del tipo de producto/servicio que les vayamos ofrecer. Por ejemplo: si es un producto tecnológico, tendremos que explorar sus habilidades digitales y cuáles son sus dispositivos preferidos. Si queremos ofrecerle una experiencia gastronómica diferente, nos preguntaremos acerca de sus gustos culinarios, sus comidas e ingredientes favoritos, etc.

POSIBLES ASPECTOS A CONTEMPLAR

- Nombre y apellidos
- Ocupación
- Hobbies
- Personalidad
- Valores y Motivaciones
- Metas
- Frustraciones
- Conocimientos
- Evolución

autentia THINKING WITHYOU

Mapa de empatía

Se trata de una herramienta que nos ayuda a entender mejor fundamentalmente a nuestro cliente, pero también, por ejemplo, a un compañero de trabajo, miembro del equipo, etc. a través de un conocimiento más profundo de sus necesidades, su entorno y motivos.

PASOS PARA USAR EL MAPA DE EMPATÍA

- **Identifica** primero quiénes son tus clientes/usuarios y agrúpalos en base a atributos. Por ejemplo: rango de edad, nivel económico, etc.
- Escoge un segmento y "**humaniza**" al cliente que lo represente, es decir, "dale vida", eso te ayudará a entenderle mejor: cómo se llama, cuáles son sus hobbies, qué hace en sus ratos libres...
- **Empatiza** construye el mapa de empatía preguntándote cosas sobre sus necesidades y motivos.
- **Valida** con el cliente real si tus hipótesis sobre lo que le motiva y necesita de verdad son ciertas.

¿QUÉ PIENSA/QUÉ SIENTE?
¿Qué es lo que considera importante y qué no?
¿Cuales son sus preocupaciones y sentimientos?
¿Cuáles son sus expectativas y aspiraciones?

¿QUÉ ESCUCHA?
¿Qué es lo que escucha en su entorno?
¿Qué le dicen sus amigos, compañeros y familia?
¿Cuáles son sus canales de comunicación?
¿Quiénes son sus principales influencias?

¿QUÉ VE?
¿Cuál es su entorno?
¿A qué ofertas está expuesto?
¿Quienes son las personas clave de su entorno?

¿QUÉ DICE/HACE?
¿Cómo se comporta en público?
¿Qué dice que le importa?
¿Con quién habla?
¿Influencia a alguien?
¿Qué contradicciones tiene?

ESFUERZOS
Miedos
Frustraciones
Obstáculos

RESULTADOS
Deseos
Necesidades

El mapa de empatía es una herramienta diseñada por XPLANE y basada en la propuesta de Alex Osterwalder & Yves Pigneur.

Importante vs. Urgente

La importancia y la urgencia son cualidades; atributos que permiten describir una tarea. Igual que un coche puede ser pequeño y rápido a la vez, una tarea puede ser urgente e importante simultáneamente. Para poder planificarlas y priorizarlas adecuadamente es necesario tener claros los siguientes conceptos:

Urgencia

Se trata de una cualidad **asociada al tiempo.** Aumenta tanto, a medida que queda menos tiempo para la fecha límite como en función del volumen de la tarea.

- Entre dos tareas que requieran el **mismo tiempo** de realización, la más urgente es la que tenga **la fecha límite antes**.
- Entre dos tareas que tengan la **misma fecha límite**, la más urgente es la que lleve **más tiempo hacer**.
- Si se **aplaza la fecha límite** de una tarea, esta se vuelve **menos urgente**.
- Si descubres que una tarea será **más larga** de lo que pensabas, se volverá **más urgente**.
- Una tarea que **no tiene fecha límite** no será **nunca urgente**.

Importancia

La **importancia** es una cualidad **asociada a las consecuencias**. Aumenta en la medida en que las consecuencias de fracasar en ella aumentan. Es decir, una tarea es importante sólo si las consecuencias que se sufrirán al no hacerlas son graves.

- Dos tareas con **consecuencias similares**, independientemente del volumen de trabajo o dificultad, **son igualmente importantes**. (Ej: tan importante es escribir un informe como llevarlo a la persona indicada)
- Entre dos tareas, aunque no se parezcan en nada, **la más importante será siempre la que cause efectos más graves** en caso de no completarla.
- Aunque la tarea no cambie, las **consecuencias pueden cambiar**. Si lo hacen, la **importancia de la tarea cambia también** (aumenta o se reduce).

		URGENCIA	
		Urgente	No Urgente
IMPORTANCIA	Importante	Tareas que es **indispensable** dedicarle tiempo de manera **prioritaria y sin demora**.	Tareas que habrá que **aplazar,** pero asegurándose de que no sea por mucho tiempo, atendiendo siempre a su prioridad.
	No Importante	Tareas que habrá que tratar de **delegar,** en la medida de lo posible.	Dependiendo del contexto se recomienda una o otra acción

Gestión de Tareas

Tareas que es necesario hacer pero que se pueden **planificar** con tiempo.

Optimización del tiempo

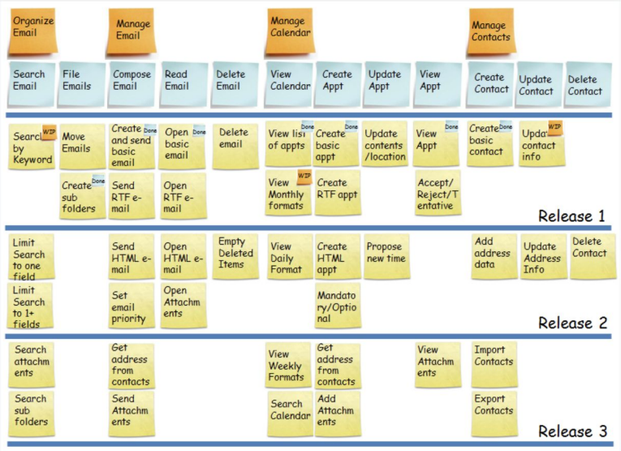
Actividades en las que se sugiere **no invertir** tiempo y si es posible deberían **descartarse**.

User Story Mapping

User Story Mapping es una técnica creada por Jeff Patton que consiste en organizar el Product Backlog visualmente en dos dimensiones: horizontal (releases) y vertical (funcionalidades). Esta técnica nos ayudará a mantener conversaciones y tomar decisiones útiles durante la construcción de nuestro producto.

PASOS DE UN USER STORY MAPPING

1. Definimos las funcionalidades a alto nivel que tendrá nuestro producto desde el punto de vista del usuario. Escribimos cada funcionalidad en un post-it para poder desplazarlo en el mapa.
2. En la línea horizontal distribuimos las acciones que hará el usuario con nuestro producto, empezando, por la izquierda, por las que el usuario utilice primero. Por ejemplo: el usuario entra en la plataforma, luego busca un producto, lo añade a la cesta de compra...
3. En la línea vertical, situamos capas de funcionalidad completa y coherente para los usuarios, situando arriba las más importantes y abajo las más prescindibles.
4. Después de marcar con una línea horizontal el MPV, definimos nuestras capas de funcionalidad. Tenemos que hacerlo incluyendo características que juntas constituyan una unidad lógica.



User Story Map creado por Steve Rogalsky.

BENEFICIOS

Además de los beneficios visuales por representarlo como una foto global, nos ayuda a diseñar un plan de construcción de nuestro producto de una manera iterativa e incremental. Ganamos claridad y ello repercute en una mayor calidad del producto.

¿Qué es eXtreme Programming (XP)?

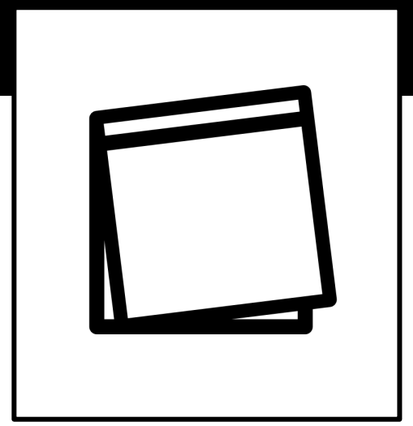
Es una **filosofía para el desarrollo de software**, basado en un conjunto de **prácticas útiles y técnicas probadas** para implementar código de forma que su diseño, arquitectura y codificación permitan incorporar modificaciones y añadir nuevas funcionalidades sin demasiado impacto en la calidad del mismo.

LAS PROMESAS DE XP

1. **Reducir el riesgo** del proyecto.
2. Mejorar la **respuesta ante cambios** en el negocio.
3. **Mejorar la productividad** a lo largo de toda la vida del software.
4. Los equipos XP producen software de calidad a un **ritmo sostenible**.

SE CARACTERIZA POR:

- Funciona con equipos de **cualquier tamaño**. Los valores y principios detrás de XP son aplicables a cualquier escala.
- Se adapta a **requisitos vagos o que cambian rápidamente**. Sin embargo, también aplica en proyectos en los que los requisitos no son tan volátiles.
- Sus **cortos ciclos de desarrollo**, que dan como resultado feedback tempranos, concretos y continuos.
- Su aproximación a la **planificación incremental**, que genera un plan general que se espera que evolucione a lo largo de la vida del proyecto.
- Su capacidad para **programar de manera flexible** la implementación de las funcionalidades, respondiendo a las necesidades cambiantes del negocio.
- Su **dependencia de pruebas automatizadas, desarrollada con TDD,** para monitorear el progreso del desarrollo y permitir que el sistema evolucione detectando los defectos lo más pronto posible.
- Su dependencia de la **comunicación oral**, **las pruebas y el código fuente** para comunicar la estructura y la intención del sistema. Para ello usa la figura del Sistema de Metáforas (System Metaphor) para describir el sistema de forma sencilla.
- Se basa en un proceso de **diseño evolutivo** que dura tanto como el sistema.
- Su dependencia de la **estrecha colaboración de personas** activamente comprometidas con talento y la excelencia.



VALORES

- Los **valores** son los criterios a gran escala que usamos para juzgar lo que vemos, pensamos y hacemos.
- Los **valores aportan un propósito a las prácticas** y deben hacerse explícitas para que las prácticas no se vuelvan rutinarias.

Ej. *Un desarrollador en una Retrospectiva da un feedback poco detallado y ambiguo. Ya ha dado un feedback pero este no es transparente y no revela lo que realmente opina por temor a ser juzgado y que sus palabras se vuelvan en su contra posteriormente. Realmente se está negando a dar un feedback más profundo de cómo ha visto el Sprint ya que no quiere tener responsabilidades por haber sido injustamente acusado en el pasado. En este caso el desarrollador valora la protección por encima de la comunicación.*

PRÁCTICAS

- Las **prácticas** son las actividades que se realizan dentro del equipo de forma continuada en el día a día y conforme a unas **reglas claras y objetivas**.
- Su ejecución da como resultado la adquisición de **nuevas habilidades y experiencia**.
- Las prácticas son la **evidencia de los valores** y pueden cambiar dependiendo de la situación.

Ej. *Si valoras la comunicación, algunas de tus prácticas probablemente sean: asistir a la Stand Up Daily Meeting a diario, o bien te preocuparás cuando recibas información confusa y preferirás una reunión cara a cara, al envío de un email para aclarar las dudas. Por tanto, darás soporte a todo tipo de prácticas que mejoren y fomenten la comunicación.*

PRINCIPIOS

- Los **principios** tienden un puente entre los valores y las prácticas.
- Son un **sistema de normas o leyes** que deben ser cumplidas para que un sistema funcione adecuadamente.

Ej. *Un jardinero que pretenda repoblar una montaña sabrá que si planta pinos, la hierba no crecerá ya que al competir por agua las raíces tan extensas de los pinos no les permitirán la supervivencia. Por tanto, se basará en el Principio de compañía para identificar qué plantas y árboles pueden crecer fuertes y sanos juntas.*

Se puede aplicar el mismo principio en la creación de un equipo de desarrollo a la hora de seleccionar los integrantes.

Valores XP

XP está basado en valores. No se trata de un conjunto de reglas, sino de una **forma de trabajar de acuerdo a tus valores personales y corporativos**. XP recomienda comenzar con los valores listados a continuación y posteriormente agregar aquellos que se alineen con el comportamiento del equipo.

COMUNICACIÓN	La comunicación **es el valor más importante** dentro de un equipo de desarrollo. Aunque algunos **problemas** son causados por la falta de conocimiento **la mayoría surgen por falta de comunicación**. Por tanto, cada uno es parte del equipo y **se comunican cara a cara** en la daily. **Se trabajará juntos en todo**, desde la definición de requisitos hasta la codificación y **se creará la mejor solución al problema entre todos.**
SIMPLICIDAD	La simplicidad es la esencia de XP, **es es el más intelectual de los valores de XP**. Por tanto, **nos centraremos en hacer lo que se necesita y se nos pide pero nada más**. Esto **maximizará el valor** creado para la inversión hecha hasta la fecha. Por esta razón, se tomarán pequeños pasos que nos hagan avanzar a nuestra meta y se mitigarán los fallos según se den. De modo que **crearemos algo de lo que estemos orgullosos** de mantener a largo plazo a un coste razonable.
FEEDBACK	El feedback es una **parte crítica de la comunicación**. Dado que **los cambios** son inevitables estos **generan la necesidad de obtener feedback**. Por consiguiente, usaremos este feedback **para acercarnos** más y más **a nuestros objetivos** y en consecuencia los equipos de XP se esforzarán por obtener la mayor cantidad de feedback que puedan manejar lo más rápido posible.
CORAJE	El coraje es la fuerza de **voluntad que puede desarrollar una persona**, sin miedo al fracaso, **para superar ciertos impedimentos** y que a veces se manifiesta como la tendencia a la acción. El coraje como valor sólo es peligroso pero junto con otros se hace poderoso. En XP se emplea para decir la verdad sobre el progreso y las estimaciones. No se teme a nada porque nunca se trabaja sólo.
RESPETO	Si los miembros de un equipo no se preocupan por el resto ni el trabajo que hacen, XP no funcionará. Por ello, **cada uno da y siente el respeto que ellos se merecen como un miembro valorado del equipo**. Cada uno aporta valor incluso si es simple entusiasmo. Los desarrolladores respetan la experiencia de los clientes y viceversa y la dirección respeta nuestro derecho a aceptar responsabilidad y recibe autoridad sobre nuestro propio trabajo.

Principios XP

Los principios permiten cerrar la brecha entre valores y prácticas, proveyendo pautas específicas de dominio para encontrar prácticas en armonía con los valores. **Los principios enumerados aquí no son los únicos posibles para guiar el desarrollo de software, pero estos son los principios que guían a XP.**

14 PRINCIPIOS

- **Humanidad**: El software lo desarrollan personas y es importante tener presente que los factores humanos son la clave para crear un software de calidad.
- **Economía**: El producto que se cree debe ser rentable tanto a corto como a largo plazo, debe producir beneficios.
- **Beneficio mutuo**: Uno de los principales principios de XP y de los más difíciles de poner en práctica. Propone pensar siempre en el beneficio de todas las partes implicadas en un desarrollo.
- **Auto-similitud**: Buscar soluciones similares en diferentes contextos. Los patrones de la forma de trabajo deben repetirse adaptándolos a la situación en la que estemos.
- **Mejora**: Excelencia en el desarrollo a través de la mejora. Se busca perfeccionar, no lo perfecto.
- **Diversidad**: Dos ideas sobre un diseño presentan una oportunidad, no un conflicto.
- **Reflexión**: Los buenos equipos piensan cómo y por qué hacen el trabajo, no esconden errores.
- **Flujo**: Entrega continua de valor.
- **Oportunidad**: Los problemas son oportunidades de cambio.
- **Redundancia**: Los problemas difíciles deben poder resolverse de diferentes maneras.
- **Fallo**: Un fallo no es un desperdicio si me sirve para aprender algo.
- **Calidad**: No vas a ir más rápido aceptando rebajar la calidad.
- **Pequeños pasos**: ¿Qué es lo menos que puedes hacer que sea reconocible y que vaya en la dirección correcta?
- **Responsabilidad aceptada**: El que acepta algo se hace responsable de ello.

Prácticas

Las prácticas son un elemento que influye en el comportamiento de las personas y las **hace ir en una dirección determinada desde donde están hasta donde pueden estar con XP**. Aunque aplicar una práctica es una elección, **las** que se indican son **prácticas** que **funcionan** y lo hacen aún **mejor cuando se aplican juntas**, ya que las interacciones entre las prácticas amplifican su efecto.

TIPOS DE PRÁCTICAS

Las prácticas se dividen en dos tipos: **13 prácticas primarias** y **11 corolarios.** Las **prácticas principales son útiles independientemente de lo que esté haciendo**. Cada una puede brindarle una mejora inmediata. Puedes comenzar con seguridad con cualquiera de ellas. Sin embargo, **los corolarios pueden resultar difíciles de aplicar sin dominar antes las prácticas primarias**. En las fichas posteriores se describe en qué consisten.

XP - Prácticas Primarias I

autentia

SENTARSE JUNTOS (SIT TOGETHER)

- Disponer de un **espacio abierto** en el que todo el equipo pueda trabajar.
- Una forma de satisfacer la necesidad de espacio privado es **limitando el número de horas de trabajo**.
- El espacio de trabajo debe promover que florezca la comunicación, **elimina los cubículos y las barreras que te impiden comunicarte** con las personas de tu alrededor.
- Cuantas **más reuniones cara-a-cara,** más humano y productivo se convierte el proyecto.

TODO EL EQUIPO (WHOLE TEAM)

- La gente necesita tener la **sensación de equipo**:
  - *Todos somos un equipo*
  - *Estamos en esto juntos*
  - *Apoyamos el trabajo, el crecimiento y el aprendizaje de todos*
- **Los equipos son dinámicos**. Si se requiere un conjunto de habilidades o actitudes, trae a una persona con esas habilidades al equipo. Si a alguien no se le requiere más, se libera y puede ir a otros lugares.

ESPACIO DE TRABAJO INFORMATIVO

- Hacer que el **espacio de trabajo se empape del proyecto** en el que estamos trabajando.
- Colocar las tarjetas con **las historias de usuario en las paredes**, agrupadas por estado.
  - *Proporcionan información rápidamente*
- Colocar también en un lugar visible distintas **gráficas de evolución del proyecto.**
  - *Información activa e importante*

PROGRAMA PRIMERO LAS PRUEBAS (TDD)

- Antes de escribir cualquier código escribir las pruebas o lo que es lo mismo **aplica un Desarrollo Orientado a Pruebas (Test Driven Development). Ventajas:**
  - ***Las pruebas sirven de especificación y aclaran el alcance*** *del código que tenemos que escribir.*
  - ***Cohesión y acoplamiento*** *- si es difícil escribir el test, es una señal de que tenemos un problema de diseño. El código débilmente acoplado y altamente cohesionado es más fácil de probar.*
  - ***Confianza*** *- escribiendo código limpio que funciona y demostrando tus intenciones con las pruebas construye una relación de confianza con tus compañeros.*
  - ***Ritmo*** *- es muy fácil perderse durante horas cuando se está programando. Con el enfoque de test-first, está claro lo que hay que hacer a continuación: o escribimos una prueba o hacemos que funcione una prueba rota (broken test). El ciclo que se genera se convierte pronto en algo natural y eficiente: test, code, refactor, test, code, refactor.*

TRABAJO ENERGIZADO (ENERGIZED WORK)

- **Trabajar** a pleno rendimiento, pero solo **tantas horas como se pueda ser productivo de forma sostenida.**
- Con suficiente **cafeína es muy fácil llegar al punto en el que el código que se escribe quita valor al proyecto** y (lo más peligroso) no darse cuenta de ello.
- Es posible aumentar la productividad **gestionando mejor el tiempo.**
  - *Recomendación de uso de la técnica Pomodoro*
  - ***Por ejemplo,*** *declarar una franja de dos horas como Tiempo de Programación, desconectar los e-mails y los móviles y dedicarse sólo a eso.*

XP - Prácticas Primarias II

autentia

PROGRAMACIÓN EN PAREJA (PAIR PROGRAMMING)

- Todo **el código** que va a producción **debe ser escrito por dos personas sentadas frente a la máquina.**
- Pair programming es un **diálogo entre dos personas** que están simultáneamente analizando, diseñando, probando e intentando programar mejor.
- **Ventajas**
  - *Se mantienen centrados mutuamente*
  - *Se clarifican ideas*
  - *Se vencen los bloqueos individuales*
  - *Se cumplen mejor los estándares del equipo*
- **Consejos**
  - *Rotar las parejas*
  - *No juntar dos programadores novatos*
  - *No invadir el espacio personal del otro (monitores grandes)*

HISTORIAS DE USUARIO (STORIES)

- Escribir las **historias de usuario en tarjetas pequeñas**, indicando un nombre, una descripción y una estimación del tiempo.
  - Recomendamos el uso de post it.
- **Es más adecuado hablar de "historias" que de "requisitos"** (palabra con connotaciones de "inmutabilidad" y "permanencia" que no son compatibles con "abrazar el cambio").
- **Al estimar cada historia es fácil** darse cuenta del coste de cada una de ellas y podemos combinarlas, **priorizarlas**.
- **Colocar las tarjetas** en la pared, **en un sitio visible**, no en un programa.
  - Recomendamos el uso de un tablero Kanban.

CICLO SEMANAL (WEEKLY CYCLE)

- **Reunión al comienzo de cada semana**.
  - *Revisión del progreso hasta la fecha, incluyendo si el progreso de la semana previa se corresponde con lo previsto*
  - *Hacer que el cliente escoja historias que sumen una semana de tamaño para la semana actual*
  - *Fraccionar las historias en tareas, los miembros del equipo se apuntan a las tareas y las estiman*
- **Comenzar la semana escribiendo las pruebas automáticas** que se realizarán cuando las historias estén completas y pasar el resto de la semana implementándolas e integrándolas en el proyecto.
- **Al final de la semana las nuevas historias deben estar disponibles para ser desplegadas.**
- **La semana es un ciclo corto para poder hacer experimentos** del estilo "vamos a usar la técnica del Pomodoro durante una semana" o "esta semana cambiamos de pareja cada hora".

CICLO TRIMESTRAL (QUARTERLY CYCLE)

- Es conveniente **hacer reuniones con un ciclo superior al semanal para ver la evolución del proyecto** en conjunto.
- Cada 3 meses:
  - *Identificar los cuellos de botella, especialmente los que no están controlados por el equipo*
  - *Planificar el tema o los temas para el cuatrimestre*
  - *Escoger historias que sumen un trimestre para cumplir con los temas escogidos*
  - *Centrarse en una visión general de cómo el proyecto encaja en la organización y del valor que añade*
- Los trimestres también **son un buen ciclo para reflexionar sobre el equipo, proponer y evaluar experimentos** que duren más de una semana.

XP - Prácticas Primarias III

autentia

TRABAJAR CON HOLGURA (SLACK)

- Trabajar con un cierto colchón que **suavice la tensión de cumplir compromisos imposibles.**
- Es importante **establecer una atmósfera de confianza** en la que los que piden y los que desarrollan el producto se comunican de forma clara y honesta.
- **Trabajar con plazos no realistas introduce errores inmanejables**, mina la moral y construye relaciones antagonistas.
- **Cumplir los compromisos**, incluso modestos, **elimina desperdicios** (waste), suaviza las tensiones, mejora la credibilidad y construye unas relaciones basadas en la comunicación honesta.

CONSTRUCCIÓN DE 10 MINUTOS (10' BUILD)

- **10 minutos para construir automáticamente todo el sistema y ejecutar todos los tests.**
- **Todo el proceso** de build y de paso de las pruebas **debería estar automatizado.**
- No **comprobar** sólo **las pruebas** de la nueva parte añadida al sistema, sino **todas** las anteriores.
- En **continuous delivery** se va un paso más allá y se automatiza también el despliegue y el lanzamiento de las nuevas funcionalidades en producción.

INTEGRACIÓN CONTINUA (CONTINUOUS INTEGRATION)

- **No dejar pasar más de dos horas sin integrar los cambios** que hemos programado.
- La programación en equipo es un problema de **"divide, vencerás e integrarás".**
- La **integración es un paso no predecible** que puede costar más que el propio desarrollo.
- **Integración síncrona:** cada pareja después de un par de horas sube sus cambios y espera a que se complete el build y se hayan pasado todas las pruebas sin ningún problema de regresión.
- **Integración asíncrona:** cada noche se hace un build diario en el que se construye la nueva versión del sistema. Si se producen errores se notifica con alertas de emails.
- **El sistema resultante debe ser un sistema listo para lanzarse sin demasiados problemas.** Si el objetivo es desplegar una web, hay que desplegar la web (en el entorno de staging).

DISEÑO INCREMENTAL (INCREMENTAL DESIGN)

- **Diseño gradual,** con pasos pequeños y seguros.
- **El diseño no debe hacerse sólo al principio**, sino conforme vamos adquiriendo experiencia.
- **Invertir cada día en el diseño del sistema,** luchando por ver cómo la parte que estamos construyendo puede mejorarlo en su conjunto.
- **Pensar todo el tiempo en el problema en conjunto**, buscando formas de diseñar mejor todo el sistema. Cuando tengamos claro un diseño mejor, trabajar gradual pero continuamente para alinear el sistema real con este diseño.
- **Los equipos XP trabajan** duro **para** crear las condiciones bajo las **que el coste de cambiar el software sea bajo.**
- Una regla sencilla para mejorar el diseño: **eliminar la duplicación**.

XP - Prácticas Corolarios I

autentia

PARTICIPACIÓN REAL DE LOS CLIENTES

- **Haz que las personas** cuyas vidas y negocios se vean afectados por tu sistema **formen parte del equipo.**
  - Tendrás resultados diferentes con clientes reales.
- En general, **cuanto más se acerquen las necesidades del cliente** a las funcionalidades del desarrollo, **más valioso se volverá el desarrollo.**
- La inclusión de clientes en el proceso de desarrollo **fomenta la confianza y la mejora continua.**

DESPLIEGUE INCREMENTAL

- Es una práctica corolario que **puede aplicarse tanto al reemplazamiento de sistemas Legacy como migraciones masivas de datos.**
- Consiste en **modificar primero algunas de sus funcionalidades** del producto ya existentes sobre el que se está trabajando **y poco a poco** ir **modificando el resto** de forma incremental.

CONTINUIDAD DEL EQUIPO

- **Un equipo de desarrollo debe mantenerse aunque se cambie de proyecto**. De esta forma se mantiene la efectividad del equipo.
- **Las relaciones y los logros del equipo aportan valor al software.** Ignorar el valor de las relaciones y la confianza para simplificar los problemas de agenda es una forma errónea de economizar.
- **Que los equipos se mantengan unidos no implica que sean completamente estáticos.** En cualquier momento nuevos miembros se pueden incorporar fomentando una rotación razonable dentro del equipo proporcionando a la compañía los beneficios de los equipos estables así como la propagación del conocimiento y la experiencia en ella.

REDUCIR LOS EQUIPOS

- Con el tiempo un equipo será cada vez más productivo. Manteniendo la carga de trabajo, se podrá enviar a alguno de sus miembros a formar nuevos equipos.
- **Evita crear equipos cada vez más grandes** ya que el rendimiento del mismo será tan pobre que valdrá la pena considerar otras alternativas.

XP - Prácticas Corolarios II

autentia

ANÁLISIS DE CAUSA RAÍZ

- Cada vez que se encuentre un defecto se debe **eliminar el defecto y su causa. El objetivo** no es sólo eliminar ese defecto sino **que el equipo no cometa el mismo tipo de error de nuevo.**
- Un ejercicio simple que se puede aplicar para ello es usar la **técnica de los 5 Porqués** de Taiichi Ohno.

NEGOCIAR EL ALCANCE DEL CONTRATO

- **Escriba contratos que fijen el tiempo, los costos y la calidad, pero no el alcance preciso del sistema.** El alcance debe negociarse de forma continua mediante contratos cortos en lugar de uno largo.
- **Los contratos grandes y largos se pueden dividir en la mitad o en tercios.** Y si la solicitud de cambio en un contrato es muy costosa, se puede redactar con menos alcance fijo por adelantado y menores costos para los cambios.
- En contratos para el desarrollo de software resulta más útil y menos arriesgado **negociar varios contratos de duración más corta** y fijar de esta forma el alcance para cada uno de ellos.
- **Negociar el alcance del contrato es un mecanismo para alinear los intereses de proveedores y clientes para alentar la comunicación y la retroalimentación**, y para tener la oportunidad de mostrar el coraje de hacer lo que parece correcto hoy y no hacer algo ineficaz sólo porque está en el contrato.

PROPIEDAD COLECTIVA DEL CÓDIGO

- **Todos los miembros del equipo sienten el código como suyo y se responsabilizan del mismo por igual**. Cualquier miembro del equipo debe ser capaz de modificar cualquier parte del código.

CODIFICACIÓN Y PRUEBAS

- Los elementos clave del proyecto que **no se pueden dejar de mantener y mejorar nunca.** Este mantenimiento es la **Refactorización, en todo momento, en todo lugar.**

CÓDIGO BASE ÚNICO

- Si por algún motivo se necesita trabajar en paralelo al desarrollo principal, esto no debería prolongarse más de varias horas.

DESPLIEGUE DIARIO

- **Cada día debe ponerse el código desarrollado en producción** para evitar un desfase entre la última versión de software y la de cada máquina local. Para poder desarrollar esta práctica Corolario es imprescindible el dominio de las primarias así como el Corolario Despliegue Incremental.

PAGAR POR FUNCIONALIDAD

- Si se paga por funcionalidad, **se sabrá con precisión hacia dónde dirigir el desarrollo.**

12 Principios del Manifiesto Ágil

Satisfacer al **usuario**

Aceptamos los **cambios**

Entrega **frecuente**

Negocio e IT juntos

Personas motivadas

Conversación **cara a cara**

Producto **funcionando**

Ritmo **sostenible**

Excelencia técnica

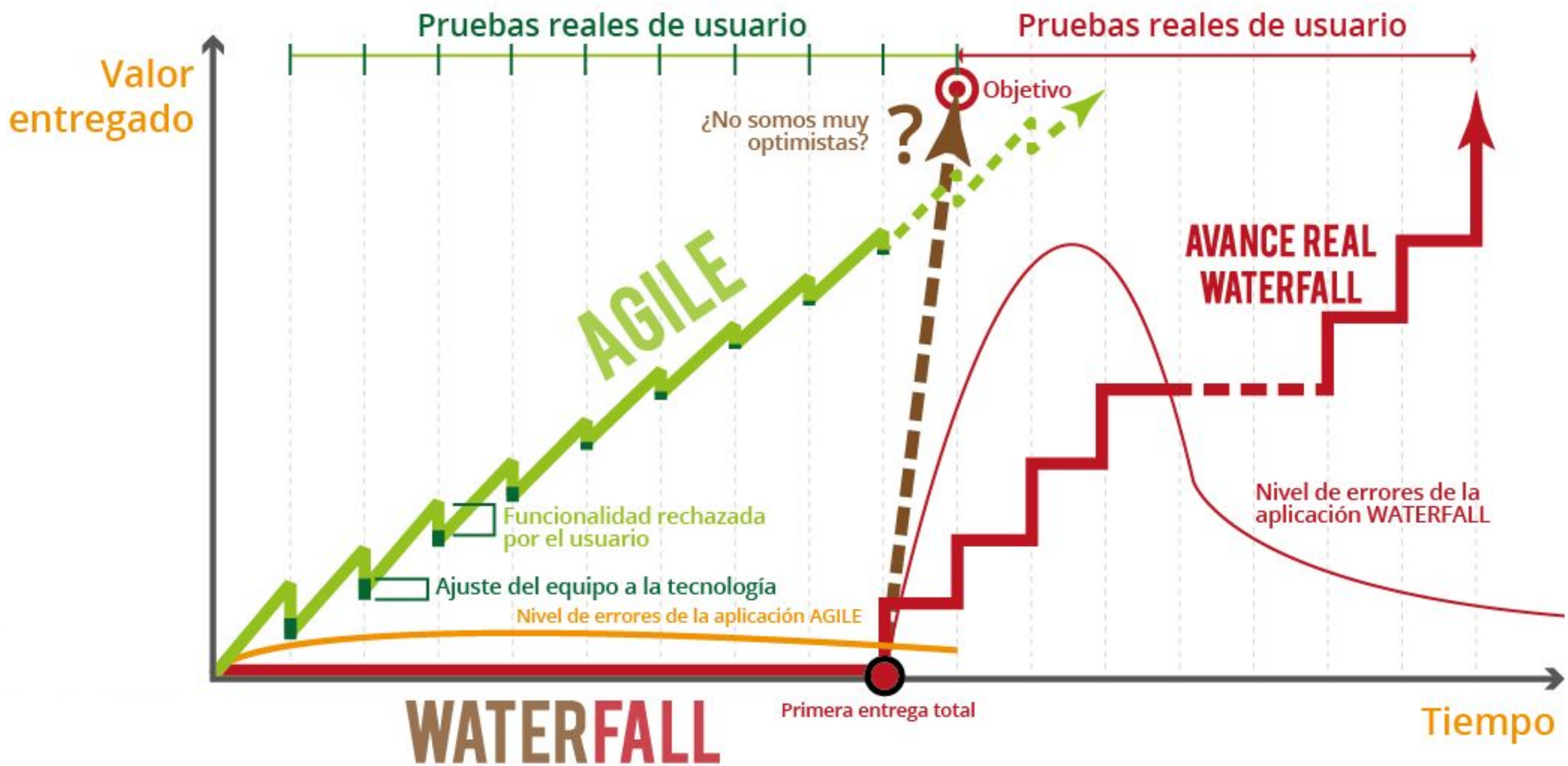
Sencillez

Emergen Arq y Diseño

El equipo **reflexiona**

http://agilemanifesto.org/iso/es/manifesto.html

Modelos tradicionales vs. ágiles



Manifiesto por el desarrollo ágil de software

Estamos descubriendo formas mejores de desarrollar software tanto por nuestra propia experiencia como ayudando a terceros. A través de este trabajo hemos aprendido a valorar:

Individuos e interacciones sobre procesos y herramientas
Software funcionando sobre documentación extensiva
Colaboración con el cliente sobre negociación contractual
Respuesta ante el cambio sobre seguir un plan

Esto es, aunque valoramos los elementos de la derecha, valoramos más los de la izquierda.

http://agilemanifesto.org/iso/es/manifesto.html

Manifiesto por el software Craftsmanship

Como aspirantes a Artesanos del Software estamos
elevando el listón de desarrollo de software profesional
practicando y ayudando a otros a aprender el oficio.
A través de este trabajo hemos llegado a valorar:

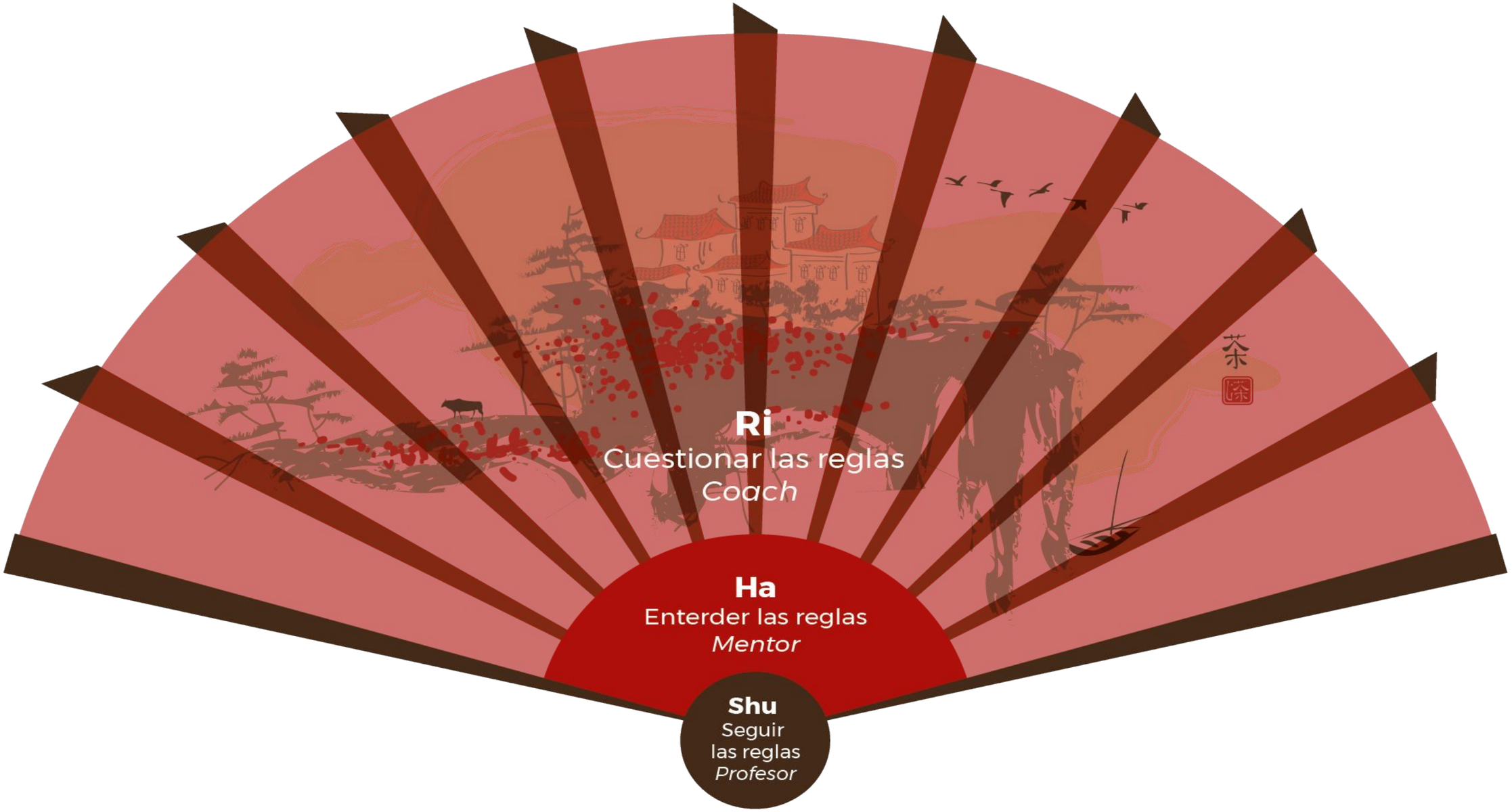
No sólo software que funciona, sino también software bien diseñado
No sólo responder al cambio, sino también agregar valor constantemente
No sólo individuos e interacciones, sino también una comunidad de profesionales
No sólo colaboración de clientes, sino también asociaciones productivas

Es decir, en la búsqueda de los elementos de la izquierda,
hemos encontrado indispensables los elementos de la derecha.

http://manifesto.softwarecraftsmanship.org/

autentia THINKING WITHYOU

Shu - Ha - Ri



改善

Kaizen, espacio de mejora

Un equipo debe ser **más potente** a nivel técnico y **mejorar** la forma en la que se coordina/comunica e**n cada iteración**.

DEFINICIÓN

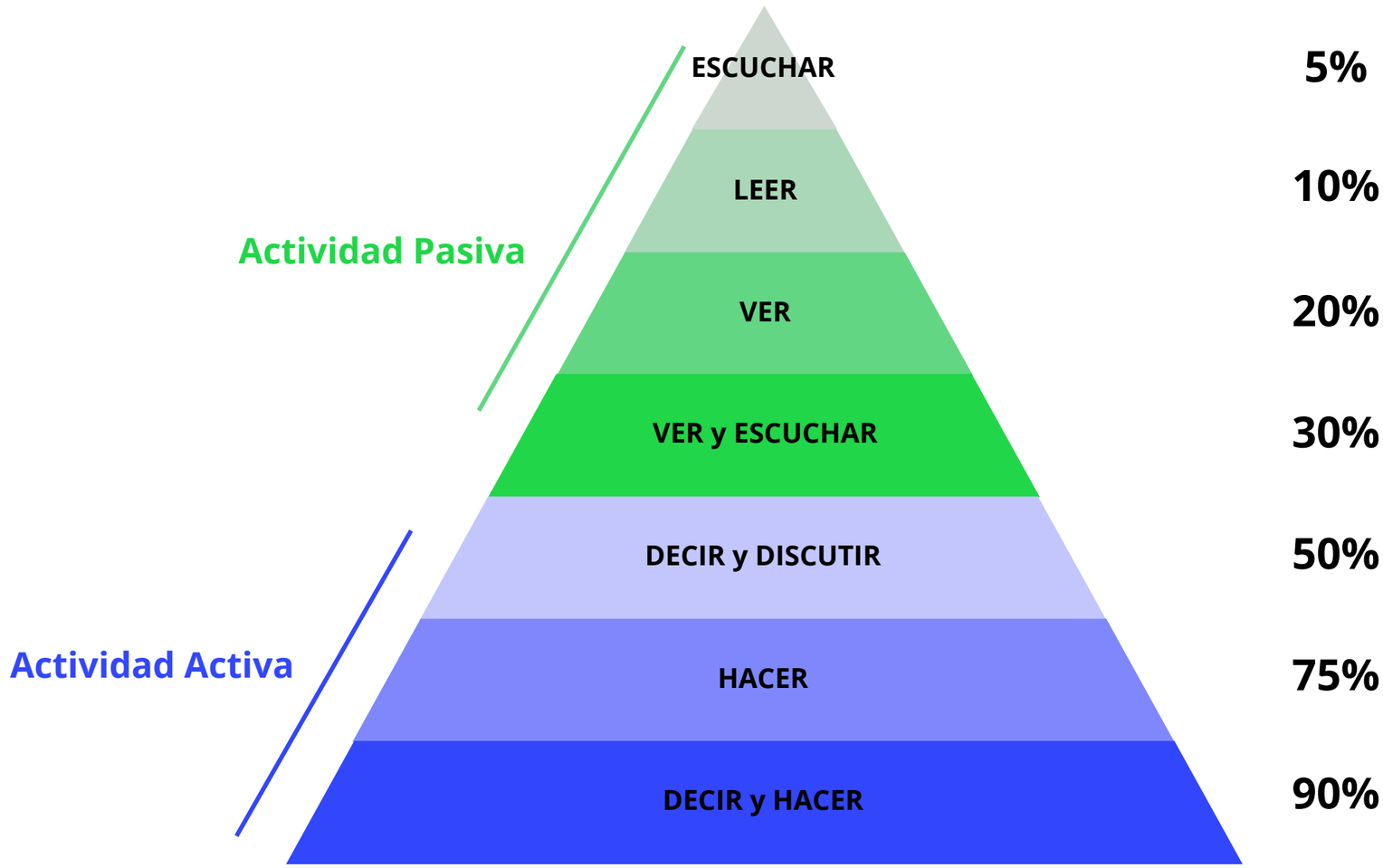
En el Backlog deben incorporarse items de mejora, como por ejemplo:

- Aprender a hacer reglas complejas de correo.
- Instalar integración continua.
- Aprender a crear campos tabulados en JIRA.
- Hacer un taller interno de TDD/BDD.
- Usar CardBoard para modelar paneles no estructurados.
- Darnos más feedback en las reuniones.
- Escucharnos activamente cuando hagamos una propuesta.

Podemos emplear técnicas como el Pair programming, los Workshops, etc para conseguir la evolución que necesitamos como equipo.

Pirámide del aprendizaje

Edgar Dale, pedagogo norteamericano, fue el creador en 1967 del denominado **cono de la experiencia**, de este deriva la **pirámide del aprendizaje**. Recordamos más lo que experimentamos que lo que escuchamos, leemos u observamos.



¡OJO! Edgar Dale no definió ningún porcentaje y no hay realmente un punto de acuerdo entre los expertos.

VUCA

Es un acrónimo empleado para describir la **volatilidad, incertidumbre, complejidad y ambigüedad** en una organización o equipo.

SIGNIFICADO

- **V** ⟶ **Volatilidad**. La naturaleza y dinámicas del cambio, la naturaleza y velocidad de las fuerzas y catalizadores del cambio.
- **U** ⟶ **Incertidumbre**. (Uncertainty en inglés). Falta de predictibilidad, perspectivas de sorpresa y sentido de consciencia y comprensión de los eventos.
- **C** ⟶ **Complejidad**. Multiplicidad de fuerzas, ruptura de causa-efecto y confusión que rodean a la organización.
- **A** ⟶ **Ambigüedad**. La distorsión de la realidad, potencial para malentendidos y los significados diferentes de las condiciones: confusión de causa y efecto. Ante una misma situación, hay diferentes interpretaciones.

OBJETIVOS

- Anticipar problemas que modifican condiciones.
- Entender las consecuencias de problemas y acciones.
- Entender la interdependencia de las variables.
- Prepararse para desafíos y realidades alternativas.
- Interpretar y aprovechar oportunidades.

LIDERAZGO

- **Integridad** del líder.
- **Humildad** y gestión del ego.
- **Empatía** y conexión emocional.
- **Determinación** y actitud enérgica.
- **Consciencia** y evolución del mapa mental.
- Generación de un **propósito y visión compartida**.

¿CÓMO SOBREVIVIR A UN ENTORNO VUCA?

- **Abordar la Volatilidad** con una visión de futuro, amplia y apoyada en los propósitos y valores más que en los beneficios a corto plazo.
- **Afrontar la incertidumbre** con conocimiento, formación, actualización constante, con habilidades que nos preparen para “gestionarnos” ante lo desconocido.
- **Aceptar la complejidad** multicausal, con espíritu crítico e indagatorio, aptitud para resolver problemas complejos, rompiendo silos y buscando la sencillez en la ejecución.
- **Reducir la ambigüedad** con agilidad y una rápida capacidad de reacción/adaptación a los cambios.

autentia THINKING WITHYOU

Agile Coach

Su objetivo es que la organización progrese en la implantación de metodologías ágiles a través del apoyo a personas y los equipos que estas conforman. Mientras que el scrum master se centra más en su equipo de desarrollo, el agile coach actúa desde una mirada más amplia y transversal a varios equipos y niveles de la organización.

A NIVEL INDIVIDUAL

- Poner la mirada en las personas, en sus percepciones, creencias limitantes, invitándoles a que salgan de su zona de confort.
- Mantener conversaciones 1:1 con los diferentes miembros, asistiéndoles ante sus dificultades en la implantación Agile.
- Formar a los miembros de los equipos en métodos o herramientas Agile.

A NIVEL DE EQUIPO

- Poner la mirada en los diferentes equipos, en sus mecanismos de comunicación y coordinación siguiendo los principios Agile.
- Eliminar impedimentos (por ejemplo, barreras jerárquicas, desafíos logísticos).
- Ayudar a poner foco en la detección de dependencias y riesgos de los equipos.
- Diseñar, facilitar y asistir a los equipos en las ceremonias y eventos.
- Promover la autoorganización, autonomía y responsabilidad en el equipo.

A NIVEL DE ORGANIZACIÓN

- Poner la mirada en las estructuras organizativas, en sus procesos, en los sistemas de trabajo.
- Brindar orientación ágil a las diferentes personas, áreas y niveles de la compañía.
- Guiar a la organización en la optimización de sus procesos y en la transformación Agile.

Ventana de Johari

La Ventana de Johari es una herramienta creada por los psicólogos Joseph Luft y Harry Ingham. Nos ayuda a conocernos mejor y a comprender **cómo varía la percepción** que tenemos de **nosotros mismos** y la **que tienen los demás**.

		YO	
		Conozco de mí	**No conozco de mí**
LOS DEMÁS	**Conocen de mí**	**ÁREA PÚBLICA** Es la parte de nosotros mismos que los demás también ven. **¿QUÉ HACER?** Mejorar la autoconsciencia de nosotros mismos, reflexionar más sobre nuestras capacidades, áreas de mejora, motivaciones, emociones...	**ÁREA CIEGA** Es la parte que otras personas perciben de nosotros y no nos cuentan. **¿QUÉ HACER?** Solicitar feedback a otras personas de nuestro entorno, pedirles que pongan ejemplos sobre nosotros: 3 cosas positivas y 3 áreas de mejora. Reflexionar sobre el feedback recibido.
	No conocen de mí	**ÁREA OCULTA** Es la parte de nosotros mismos que conocemos pero que no queremos compartir con los demás: miedos, sentimientos (pensamos que si los hacemos públicos nos harán daño, no nos apoyarán...). **¿QUÉ HACER?** Hacer explícita: mostrar primero a las personas de mayor confianza. Si son fortalezas, ponerlas en práctica en reuniones, proyectos...para hacerlas más visibles.	**ÁREA DESCONOCIDA** Es la parte que ni siquiera uno mismo ni los demás perciben. Capacidades y talentos ocultos: sentimientos reprimidos, miedos desconocidos, fobias... **¿QUÉ HACER?** Descubrir y explorar esa parte de nosotros a través de acompañamientos con profesionales.

Personal Maps

Es una herramienta que nos va a permitir conocer mejor a las personas de nuestro equipo. Consiste en explorar y representar gráficamente varios aspectos personales de nuestros compañeros mientras conversamos con cada uno de ellos.

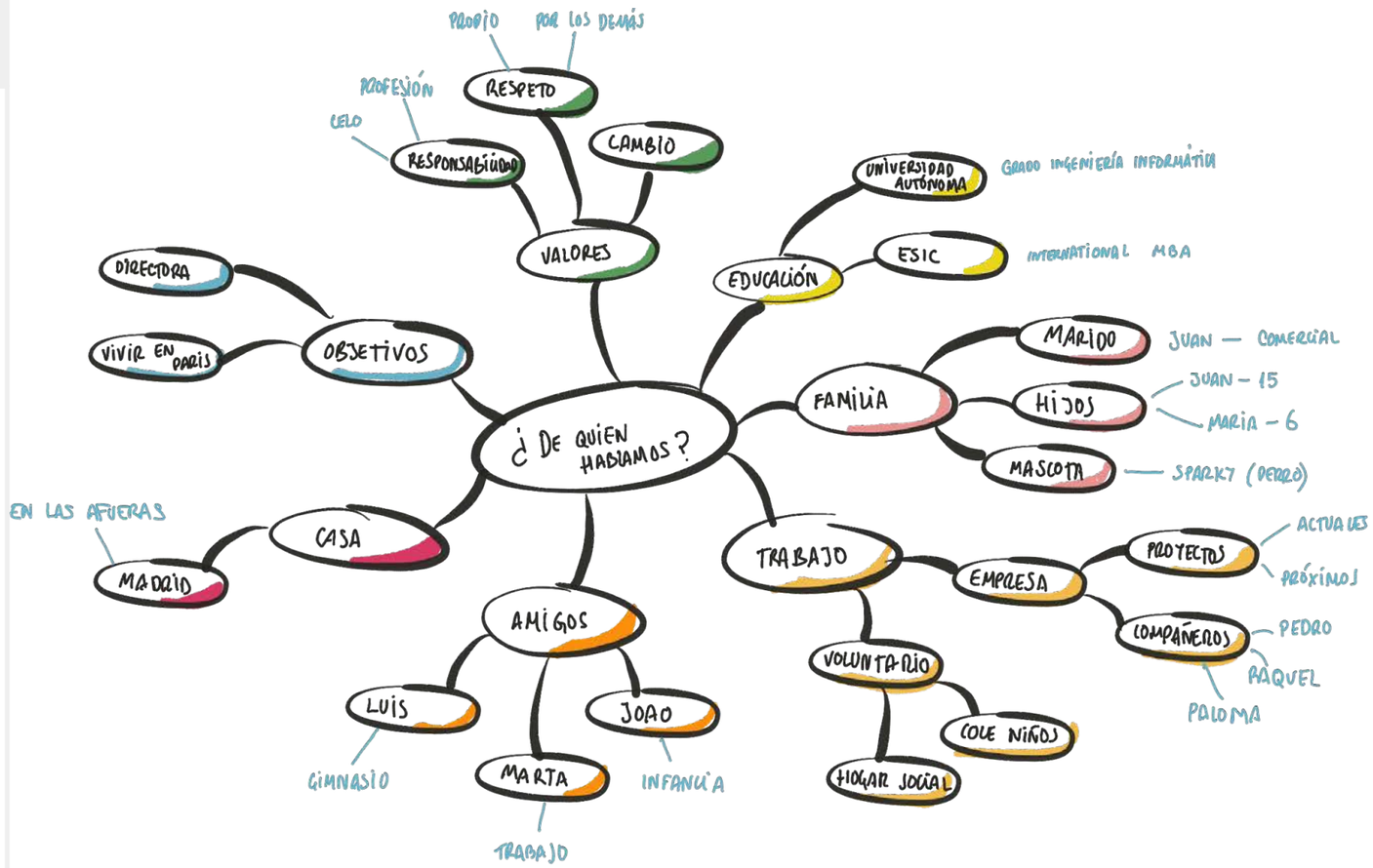
CONSTRUYE TU PERSONAL MAP

Pasos:

- Hoja en blanco, escribe el nombre de la persona en el centro.
- Escribe aquellos aspectos que quieras conocer de la persona.
- Hobbies, personalidad, metas, valores, familia, comida favorita...
- Inicia la conversación!

Concluye:

- ¿Qué sabes de esta persona que antes no sabías?
- ¿Ha cambiado tu percepción?
- ¿Qué nueva mirada tienes hacia esta persona?
- ¿Cómo te gustaría que fuera la relación con ella a partir de ahora?
- ¿Qué harás diferente?



Basado en Jurgen Appelo

Recompensas y Reconocimientos

El reconocimiento y la recompensa son una bonita forma de mantener motivado al equipo, además de reforzar positivamente comportamientos sanos.

CLAVES PARA RECOMPENSAR

- Recompensa justa: ni mucho ni poco.
- Recompensa de forma constante.
- Recompensa públicamente.
- Recompensa el cómo, no sólo el qué.
- Recompensa también entre compañeros no solo del líder hacia el colaborador.

CUIDADO CON LA MOTIVACIÓN EXTRÍNSECA

- Los refuerzos externos influyen en la motivación intrínseca satisfaciendo o frustrando necesidades de autodeterminación y competencia.
- Cuando recompenses, potencia la motivación intrínseca a través de lo siguiente:
  - La necesidad de relación: que la recompensa implique alguna socialización con el equipo.
  - La competencia: que la recompensa incluya un feedback sobre nuestro valor, lo que hemos aportado con nuestro trabajo.
  - La autonomía: sentir que elegimos la recompensa, por ejemplo, eligiendo el premio tras un trabajo bien realizado.

DIFERENTES MOTIVADORES INTRÍNSECOS

- Exploración: “me gusta aprender en el trabajo, investigar y averiguar cosas nuevas”.
- Contribución: “me gusta sentir que contribuyo a ayudar a otras personas y a la sociedad”.
- Logro: “me gusta superar los estándares, conseguir resultados ambiciosos”.
- Autorealización: “me gusta superarme a mí mismo y sentir que mi trabajo está alineado con mi propósito”.
- Autonomía: “me gusta tener libertad para definir partes de mi trabajo y tomar decisiones”.
- Poder: “me gusta tener prestigio e influencia en mi organización”.
- Afiliación: “me gusta tener buenas relaciones en el trabajo y un ambiente de equipo”.
- Seguridad: “me gusta trabajar en un entorno seguro que me de estabilidad”.

La matriz de confianza

La confianza está permanentemente en construcción. Se cimenta continuamente por ambas partes. No esperamos a que “se ganen nuestra confianza”. Somos responsables de construirla también nosotros mismos.

LAS 3 PATAS DE LA CONFIANZA

Competencia

- ¿Es la persona competente para cumplir con aquello con lo que se ha comprometido?

Sinceridad

- ¿Es sincera la persona? ¿Nos miente?

Histórico

- ¿Qué evidencias positivas y negativas de confianza tenemos con esta persona en el pasado?

MATRIZ DE CONFIANZA

Construye tu propia matriz de confianza con las personas de tu equipo.

Persona de tu equipo	Interés genuino por la persona (1 al 10)	Competencia (1 al 10)	Sinceridad (1 al 10)	Histórico (+ o -)

Pregúntate: ¿qué vas a hacer en las próximas semanas para que esa persona sea más confiable para ti?

Basado en idea de EEC

CONFIANZA 360°

- Confía en ti mismo.
- Gánate la confianza de los miembros del equipo.
- Confía tú en el equipo.
- Consigue que los miembros del equipo confíen entre ellos.

ESPEJAR: Reproducirse como una imagen en un espejo

CÓMO GENERAR CONFIANZA DESDE LA ESCUCHA

Espejamos con el lenguaje

- Mismas palabras.
- No juzgar.
- Hacer preguntas para entender.

Espejamos con el cuerpo

- El baile.

Espejamos con la emoción

- Adaptación de intensidad.

Delegación

Delegar no es perder el tiempo ni tampoco es perder el control, es una inversión. Delegar es compartir responsabilidades, es que cada uno haga lo que le corresponde para cumplir los objetivos propios y del equipo, para desarrollarse, para permitir que se desarrollen los demás, para avanzar y para conseguir mucho más de lo imaginado. ¡Esto va de equipos, no de héroes!

NIVELES DE DELEGACIÓN / AUTORIDAD

- **Informa:** anuncia las decisiones tomadas.
- **Vende:** toma las decisiones y luego cuentas las bondades al equipo para ganar su compromiso.
- **Consulta:** consultas a la gente antes de tomar la decisión. Pero dejas claro que serás tú quien tome la decisión.
- **Acuerda:** invita al equipo para tomar la decisión por consenso. Tu voz es igual que la del equipo.
- **Asesora:** trata de influir al equipo pero dejas claro que la decisión depende de ellos.
- **Pregunta:** deja que el equipo decida pero le pides que luego te convenzan/informen de la decisión.
- **Delega:** delegas la decisión plenamente en el equipo y utilizas tu tiempo para otra cosa.

CHECKLIST DE DELEGACIÓN PARA EL LÍDER

- ¿Tratas adecuadamente el factor de riesgo de delegar este trabajo?
- ¿Tienen las personas las habilidades y la disciplina de empoderamiento adecuadas?
- ¿Has considerado y seleccionado el nivel correcto de autoridad?
- ¿Tienen las personas las herramientas necesarias?
- ¿Las personas saben cuáles deberían ser los resultados?
- ¿Estableciste las condiciones de entorno para el trabajo (presupuesto, tiempo, recursos y calidad)?
- ¿Las personas saben cuándo se debe hacer el trabajo?
- ¿Las personas saben cómo se mide el progreso?
- ¿Las personas saben con qué frecuencia deben informarte sobre el progreso (adhiriéndose a los hitos intermedios)?

Basado en Jurgen Appelo

Escucha empática

La escucha empática es una **escucha limpia de prejuicios**, es tratar de comprender al otro a través de una **escucha atenta**, antes de tratar de ser comprendido. **No es** escuchar para tener razón, ni para **justificarnos o reprochar**. Una escucha empática implica querer comprender los sentimientos, necesidades y razones que hay detrás de las palabras que escuchamos. Es escuchar con la inocencia de un niño que se deja sorprender y transformar por las palabras del otro.

HERRAMIENTAS PARA MEJORAR LA COMUNICACIÓN

Como hablante

- Cotejamos - ¿me he explicado? - ¿qué es lo has interpretado? ¿Puedes resumir...?

Como oyente

- Verificamos - déjame verificar si te he entendido bien - déjame que te repita -
- Indagamos - dame más detalles - puedes ser más específico - ¿a qué te refieres?

CONSEJOS PARA LIMPIAR NUESTRA ESCUCHA

- ¿Qué tipo de escucha estoy haciendo?
- ¿Para qué estoy escuchando?
- ¿Qué juicios interfieren en mi escucha?
- ¿Qué me está faltando para preguntar desde la inocencia?

ESCUCHAR ES / ESCUCHAR NO ES

Escuchar **NO ES**:

- Interrumpir
- Completar las frases
- Distraerse
- Dar soluciones

Escuchar **ES**:

- Mirar a los ojos
- Estar en silencio
- Cotejar lo que dice el interlocutor
- Escuchar su cuerpo y lenguaje - lo que dice y lo que no dice
- Aceptar al otro como sin quererle controlar ni cambiar
- Estar abierto a la transformación

Basado en idea de EEC

Escucha empática

autentia THINKING WITHYOU

Para poder comprender la escucha empática es imprescindible conocer los diferentes niveles de escucha..

EMPÁTICA

Sentimos un interés genuino por la otra persona. Escuchamos para comprender sus inquietudes y emociones y no para responder. Generamos empatía con la otra persona. Escuchamos desde su paradigma. Escuchamos para dejarnos transformar por sus palabras.

ACTIVA

Cuando escuchamos con nuestros sentidos puestos. Estamos presentes. No nos distraemos ni interrumpimos a la otra persona. Estamos en silencio. Nos interesa lo que la otra persona nos está contando.

SELECTIVA

Cuando escuchamos, pero solo lo que nos interesa oír. Por ejemplo, aquello que nos da la razón sobre lo que pensamos. No tenemos una escucha global ni atenta.

FINGIDA

Cuando parece que estamos escuchando a alguien pero nuestra mente está en otro lado. Aunque pretendamos que estamos interesados, no nos quedamos con lo que nos están diciendo. No nos importa entender a la otra persona.

BIOLÓGICA

Cuando escuchamos un ruido, por ejemplo.

La pregunta

La pregunta es una herramienta fundamental para el desarrollo de personas y equipos en las organizaciones. Muchas veces los conflictos y los problemas de comunicación aparecen porque no generamos espacios suficientes para conversar entre nosotros. Las preguntas ayudan a reflexionar, a plantearse nuevos puntos de vista, fomentan el descubrimiento y el aprendizaje en la persona.

CARACTERÍSTICAS DE LAS PREGUNTAS PODEROSAS

- Abiertas, cuya respuesta no sea si/no.
- Cortas.
- Sin juicio.
- Reflejan una escucha activa.
- Ayudan a tomar consciencia del observador que somos de la realidad.
- Estimulan la reflexión.
- Se enfocan en el futuro.
- Exploran posibilidades, nuevos aprendizajes, nuevas acciones.

LAS 3 PREGUNTAS DE UN LÍDER-COACH

- ¿Cuál es tu objetivo?
- ¿Qué te falta?
- ¿Qué puedes hacer para conseguirlo?

ALGUNAS PREGUNTAS PODEROSAS

- ¿Cuáles son mis habilidades y fortalezas?
- ¿Cuál es mi objetivo?
- ¿Hasta qué punto estoy comprometido con él?
- ¿Dónde me veo de aquí a 3 años?
- ¿Qué me gustaría mejorar?
- ¿Qué tengo que soltar? (Malos hábitos, creencias limitantes...)
- ¿Cuáles son mis necesidades y expectativas?
- ¿Qué no estoy dispuest@ a cambiar?
- ¿Qué me impide actuar?
- ¿A qué me estoy resistiendo?
- ¿Qué haré de diferente manera la próxima vez?
- ¿Qué apoyos tengo?
- ¿En quién o qué me podría ayudar?
- ¿Qué cosas me dan energía cada día y me mantienen motivad@?

autentia THINKING WITHYOU

Ejemplos de preguntas poderosas

PREGUNTAS SOBRE OBJETIVOS Y VISIÓN

- ¿Qué te gustaría conseguir?
- ¿Qué metas tienes?
- ¿Dónde te ves de aquí a 3 años?
- ¿Qué te gustaría que pasara al final de este proyecto?

PREGUNTAS SOBRE EMOCIONES Y NECESIDADES

- ¿Cómo te sientes?
- ¿Cómo te gustaría sentirte?
- ¿Cómo te hace sentir esta relación / este conflicto?
- ¿Cuáles son tus necesidades?

PREGUNTAS PARA TOMAR CONSCIENCIA

- ¿De qué te has dado cuenta?
- ¿Qué sabes ahora que antes no sabía?
- ¿Qué has aprendido en la sesión?
- ¿Cuál es la conclusión que sacas de todo esto?

PREGUNTAS PARA EL PLAN DE ACCIÓN

- ¿Cuál será tu próximo paso? ¿Qué pequeño paso puedes dar?
- ¿Qué cosas te ayudarían a conseguir tu objetivo?
- ¿En qué o quién te podrías ayudar?
- ¿A qué te compromete específicamente?
- ¿Qué vas a hacer de aquí a la próxima sesión?

RECUERDA: ¿Cuál es tu objetivo? ¿Qué te falta? ¿Qué puedes hacer para conseguirlo?

Las claves del feedback

El feedback es toda la información que nos devuelve el entorno sobre nuestra forma de ser y de actuar. Es una de las mejores herramientas que tenemos para crecer. El feedback nos da información sobre nuestras áreas ciegas que sólo otros pueden ver. El feedback es un regalo, nosotros decidimos con qué parte quedarnos y si nos ayuda para ser mejores personas y profesionales.

PASOS PARA DAR FEEDBACK

- Piensa para qué vas a dar el feedback, ¿cuál es tu objetivo? ¿Qué no te gustaría que pasara?
- Piensa también qué quieres para la otra persona, ¿qué te gustaría que se llevara de la conversación?
- Escoge un buen momento. Agenda la conversación. Evita las conversaciones de pasillo. Pide permiso.
- Genera contexto y un entorno de seguridad y confianza.
- Haz explícito lo valiosa que es la relación para los dos. “Eres valioso para mí, por eso te quiero dar este FB”.
- Establece el objetivo común para la conversación.
- Básate en hechos concretos, pon ejemplos que funden la opinión.
- Añade “es mi punto de vista” y habla también de tus emociones.
- Ayuda a la persona ofreciéndole sugerencias de mejora y estableciendo compromisos mutuos.

REGLAS DEL FEEDBACK

- Da feedback siempre al “balón”, nunca del “jugador”.
- Evita el “siempre” y el “nunca”.
- Sustituye el “si pero” por el “si y además”.
- Recuerda que es tu opinión.
- Reconoce tus emociones y sentimientos y las de la otra persona.
- Focálizate en el futuro: refuerza la posibilidad de cambiar, da sugerencias de mejora.
- ¡El FB es un regalo! Cada uno decide lo que hace con él, ¿con qué FB quieres quedarte?
- Se retribuye diciendo: “gracias”.

EVITAR...

- El ataque.
- La ironía.
- La amenaza.
- Querer tener razón a toda costa.
- Las preguntas desde el juicio.

Preguntas y recursos efectivos para el feedback

PREGUNTAS PARA PREPARAR UNA CONVERSACIÓN DE FEEDBACK

- ¿Cuál es tu motivación para mantener dicha conversación?
- ¿Qué quieres que ocurra en la conversación?
- ¿Qué te aporta la otra persona? ¿Qué tenéis en común?
- ¿A qué dificultades te enfrentas? ¿Cómo las podrías resolver?
- ¿Qué emoción sientes al pensar en la conversación? ¿Qué emoción te gustaría sentir? ¿Qué necesitarías para ello?
- ¿Qué necesitarías para que te apeteciera mantener la conversación?

PREGUNTAS PARA FINALIZAR UNA SESIÓN DE FEEDBACK

- ¿Hay alguna pregunta que no nos hayamos hecho?
- ¿Qué hemos aprendido?
- ¿Qué queremos que pase a partir de ahora?
- ¿A qué acciones nos comprometemos los dos?
- ¿Cómo sabremos que lo hemos conseguido?

OTRAS PREGUNTAS

PREGUNTAS PARA PENSAR

- ¿Cómo llegas a esa conclusión?
- ¿Qué te hace pensar eso?
- ¿Podría haber otra explicación?

PARA CONOCER SU PUNTO DE VISTA / EMOCIONES

- ¿Cómo lo ves tú?
- ¿Qué piensas?
- ¿Cómo te sentiste ante ese hecho?

PARA GENERAR CLARIDAD

- ¿Cuánto es “mucho” o “poco” para ti?
- ¿Qué significa para ti “ser carismático” para ti?

PARA QUEJAS

- ¿Cómo harías para?
- ¿Qué propones tú?
- ¿Qué está en tu mano / en tu círculo de influencia?

PARA ABRIR POSIBILIDADES

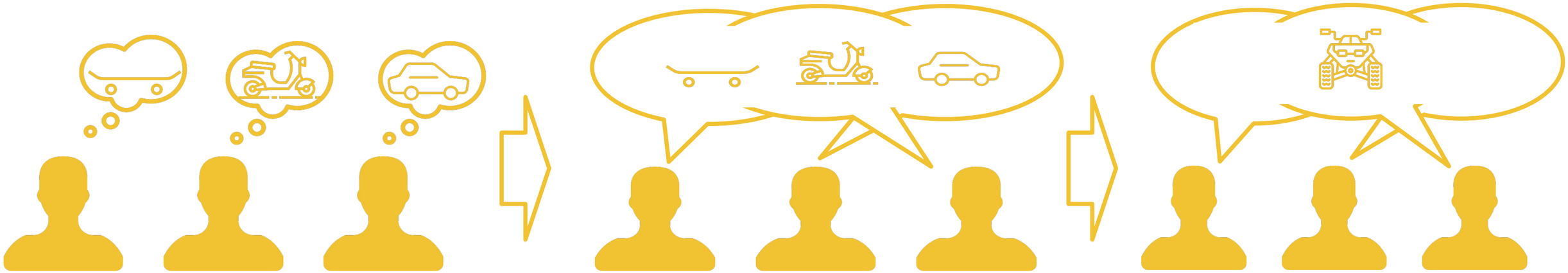
- ¿Qué pasaría si?
- ¿Qué no has intentado?

Agile Inception Deck

Jonathan Rasmusson, en su blog The Agile Warrior, presenta The Agile Inception Deck como una baraja de opciones orientada a evitar malentendidos y fomentar el diálogo en la fase inicial de un proyecto.
https://agilewarrior.wordpress.com/2010/11/06/the-agile-inception-deck/

¿ESTAMOS REALMENTE HABLANDO DE LO MISMO?

Un aspecto fundamental en todos los proyectos es el de **dialogar**. Cuando el proyecto se va a iniciar, todo el mundo **cree estar alineado con respecto al objetivo del proyecto o producto**. Luego empezamos a trabajar, a construir, a aprender y entonces es cuando realmente **nos damos cuenta que todos teníamos una concepción distinta en nuestras cabezas**. Esto no deja de ser **una forma de desperdicio** que puede dañar al proyecto o incluso acabar con él, el Agile Inception Deck nos ayuda a trabajar esta comunicación.
Mejor hacerse estas 10 preguntas o ejercicios antes de comenzar el proyecto que optar por lo hacerlo. Alinearse, va de estar todos en la misma página en relación a porqué estamos aquí, qué queremos hacer y cómo vamos a llegar al objetivo.
¿Cómo sabes exactamente lo que tu cliente realmente necesita?. **Un buen punto de inicio es... preguntarles.**



https://agilewarrior.wordpress.com/2010/11/06/the-agile-inception-deck/

AHORA SÍ QUE ESTAMOS HABLANDO DE LO MISMO

Los objetivos son dos: Alineamiento y Control de Expectativas.

El alineamiento es asegurarse de que todo el mundo entiende lo mismo sobre por qué estamos aquí, qué intentamos hacer y cómo vamos a llegar al final. Sencillo, ¿no?.

Establecer expectativas es comunicar con claridad al equipo y a los stakeholders lo que va a costar llevar al proyecto al éxito, definiendo las reglas para el compromiso y la implicación.

Este diálogo hay que tenerlo muy al principio, antes de arrancar el proyecto, y asegurarse que sabemos lo que necesita el cliente, sin asunciones, siendo claros y haciendo las preguntas necesarias aunque sean difíciles.

- ¿Por qué estamos aquí?
- Crear un Elevator Pitch
- Diseñar una Caja de Producto
- Crear una lista de NOes
- Conoce a tus vecinos
- Haz ver la solución
- ¿Qué nos quita el sueño?
- Tómale las medidas
- Ser claros en lo que vamos a dar
- Muestra lo que va a costar

1. ¿Por qué estamos aquí?

No se puede crear un gran producto, si en primer lugar, no sabes por qué lo estás creando. Preguntarnos a nosotros mismos el por qué de lo que vamos a hacer o estamos haciendo, nos da como equipo el contexto que necesitamos para tomar decisiones necesarias.

MOTIVACIÓN

Por ejemplo, imaginemos que una empresa nos contrata para crear una aplicación. Una pregunta obvia que nos ayudaría, a nosotros y al equipo para crear la solución es, **¿Por qué?** ¿Por qué esta empresa se gasta dinero en esto? ¿Es por un tema de seguridad o es un requisito de regulación estatal? ¿O es sencillamente para tener un servicio de gestión eficiente para mejorar las condiciones y el servicio que se han estado dando hasta ahora?

Conociendo y **entendiendo** el motivo principal de nuestro **proyecto**, y los **beneficios** que se esperan (sean económicos o no) nos va a dar el conocimiento necesario para ayudarnos a tomar las decisiones y realizar las negociaciones oportunas. Esto no lo podremos hacer bien si no conocemos el porqué.

https://agilewarrior.wordpress.com/2010/11/06/the-agile-inception-deck/

2. Elevator-Pitch

El elevator pitch consiste en transmitir, de una forma cautivadora, la idea de un proyecto o de una solución en lo que dura un viaje de ascensor, 2 minutos, para que al terminar, nuestro interlocutor desee escuchar más, es decir hayamos capturado su atención sobre nuestro producto, proyecto o servicio. Si conseguimos empatizar con sus necesidades, será mucho más fácil capturar su atención.

ELEMENTOS DE UN ELEVATOR PITCH

Dolor/Necesidad

- ¿Qué necesidad resuelves? ¿Quiénes serían tus clientes?

Solución

- ¿Cómo pretendes resolver la necesidad? Explicación breve de cómo funciona la solución

Mercado

- ¿Cuál es el mercado y la tendencia futura? Por qué el mercado merece la pena.

Competidores

- ¿Cuáles son tus competidores y tu ventaja competitiva frente a ellos? ¿Por qué lo puedes hacer mejor que ellos?

Equipo

- ¿Cuál es el equipo? ¿Cuál es su magia?

Call to action

- ¿Qué le pides a tu interlocutor? Petición de un siguiente paso concreto y un compromiso por su parte.

CARACTERÍSTICAS

Conciso

- Entre 15 y 90 segundos

Menos es más

- ¿Podrías explicar tu negocio en una frase?

Claro

- Pocas frases y atractivas.
- Sin conceptos vacíos y complejos.

Específico

- Adaptado a cada situación y receptor.

Sugerente

- Que consiga que la persona quiera conocer más información.

DESTINATARIOS

Es importante modular el discurso y los mensajes en función de nuestro destinatario.

- Inversores, socios, clientes, auditorio general, jurados de un concurso...

3. Diseñar una caja de producto

Diseñar una caja de producto es un concepto de marketing que nos invita a imaginar la caja de un producto con los mensajes e imágenes necesarias como si nuestro producto estuviera empaquetado y fuera a estar expuesto de forma que resulte lo más atractivo posible para el cliente.

DISEÑAR UNA CAJA DE PRODUCTO

Pensar profundamente en el producto desde el punto de vista del cliente, siempre está bien. Crear un envase no sólo te mete en la cabeza del cliente si no que esta tarea llevada a cabo por todo el equipo es una actividad de *teambuilding* muy agradable y enriquecedora. No hace falta que obtenga algo tremendamente complejo o elaborado, simplemente preguntarse unas cuantas cosas:

ESTRUCTURA DEL MENSAJE

¿Cuáles son las tres principales razones para que la gente quiera comprar este producto?

Conseguir un slogan que capte el espíritu del producto o servicio

https://agilewarrior.wordpress.com/2010/11/06/the-agile-inception-deck/

4. Crear una lista de NOes

Felicitar es fácil pero dar condolencias, no.
Una lista de **Noes** lo que hace es poner límites a la hora de empezar a hablar, estableciendo un punto de referencia a las expectativas sobre lo que no se va a considerar como parte del proyecto.

DENTRO DEL ALCANCE

Relatar las funcionalidades o áreas que se consideran viables de realizar. Es muy común que aquí se identifiquen las partes básicas del sistema como la gestión de usuarios o disponer de informes personalizados o funcionalidades automatizadas o complejas en esta fase.

FUERA DEL ALCANCE

Obtener una relación de las funcionalidades o áreas que tienen que quedar fuera del alcance inicial. Normalmente en este área caen las funcionalidades que representan mayor complejidad o sofisticación.

SIN DECIDIR

En este área estaría incluido todo lo deseable aunque técnicamente inviable por duración o por precio. Lo que todos querríamos que incorporara el sistema pero más adelante.

EN RESUMEN

Decir lo que no vas a hacer tiene un efecto muy potente. Elimina un montón de desgaste permitiendo que el equipo se focalice en lo que está **DENTRO** del alcance e ignorando lo que está fuera. Es del área **DENTRO** de donde emergerán todas las características y funcionalidades. Tampoco es inusual que haya cosas que, a pesar de estar **DENTRO,** por razones como falta de tiempo o dinero se queden **FUERA**.

https://agilewarrior.wordpress.com/2010/11/06/the-agile-inception-deck/

5. Conoce a tus vecinos

A veces nos centramos tanto en nosotros y en nuestro equipo que nos olvidamos de todo lo demás.
El entorno que nos rodea es tan amplio, sobre todo en grandes empresas que, a veces, no somos capaces de identificar a cuantas personas de nuestro entorno necesitamos para que nuestro proyecto sea un éxito.

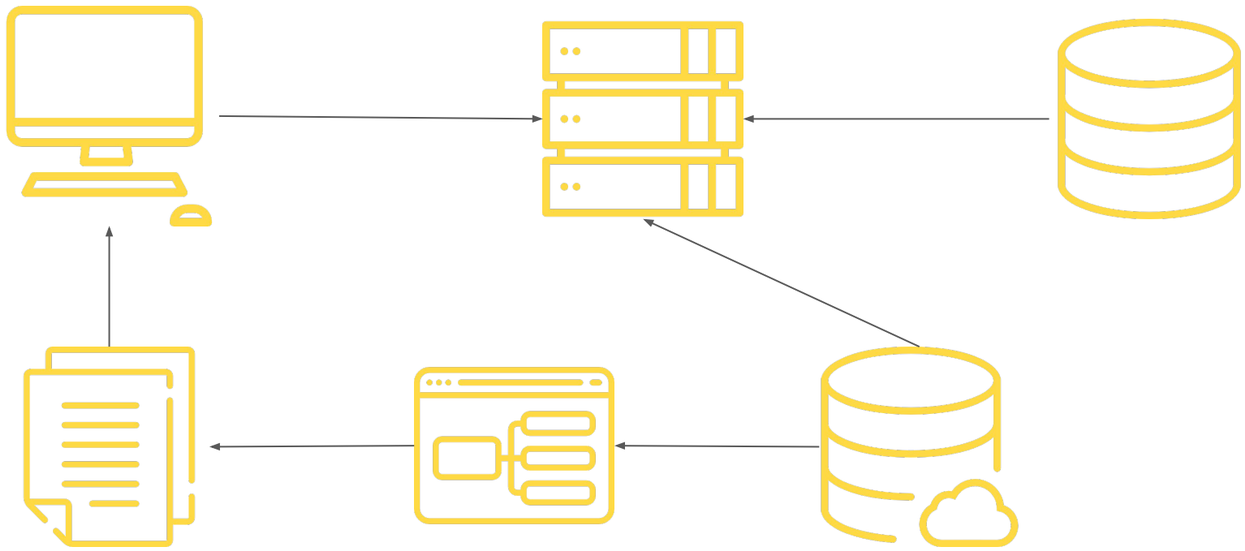
MECÁNICA

Como equipo tenemos que pensar con quién necesitamos vernos o reunirnos y establecer una relación antes de salir a producción. Buscamos la implicación de todos los actores y que, llegado el momento, conozcan nuestro producto para poder colaborar. Por ejemplo, si vamos a comercializar un nuevo producto, Ventas tendrá que conocerlo para poder ofrecerlo y atención al cliente tendrá que saber cómo funciona para ayudar a los usuarios cuando tengan problemas.

https://agilewarrior.wordpress.com/2010/11/06/the-agile-inception-deck/

6. Haz ver la solución

A la hora de diseñar la Arquitectura la composición del equipo es fundamental. Hay que asegurarse que todo el equipo está alineado con la solución antes de empezar por lo que mostrarle la arquitectura y las herramientas del proyecto es una necesidad.



Arquitectura del Proyecto

MECÁNICA

Este ejercicio va de que tú y tu equipo penséis por adelantado con qué arquitectura y herramientas se va a acometer el proyecto no vaya a ser que el equipo no tenga todos los skills o nos estemos saltando un estándar corporativo

https://agilewarrior.wordpress.com/2010/11/06/the-agile-inception-deck/

7. ¿Qué nos quita el sueño?

Hay mil cosas que pueden ir mal en nuestro proyecto. Algunas se pueden manejar y otras no. El objetivo es identificar riesgos y centrarnos en resolver aquellos en los que podemos influir y no preocuparnos por los que están fuera de nuestro alcance (esto además nos permite ser conscientes de que estos últimos existen).

RIESGOS EN LOS QUE PODEMOS INFLUIR

- Tener un equipo equilibrado en skills.
- Que el equipo esté junto y no separado.
- Tener un equipo con una dedicación total al proyecto.
- Disponer de entornos de pruebas y test.
- Tener suficiente conocimiento en las tecnologías que utilizaremos.
- Accesibilidad del cliente y stakeholders.

RIESGOS EN LOS QUE NO PODEMOS INFLUIR

- La red social que usamos para identificarnos decide hacer de pago el registro de usuarios.
- Una autoridad regulatoria externa de la que dependemos cambia sus políticas.
- Un tercero compra un partner con el que trabajamos y finaliza nuestra relación comercial.

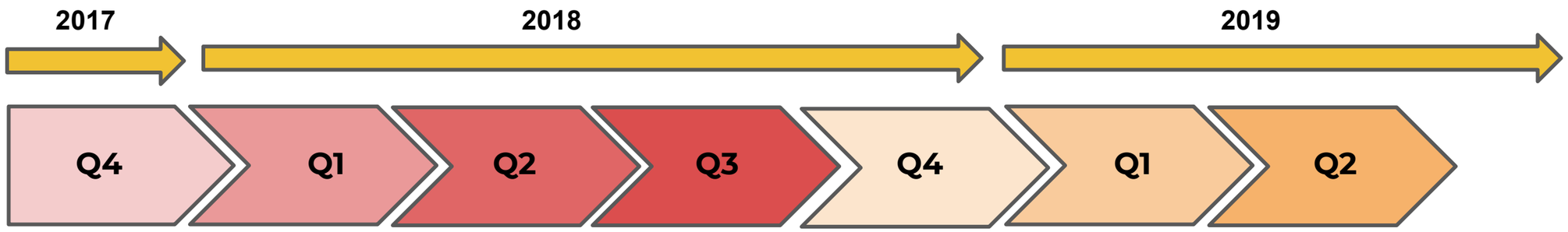
MECÁNICA

Como equipo tenemos que pensar en los riesgos que están cercanos y para los que nos podemos preparar y no pensar en si la economía mundial va a ir mal o si puede haber cambios internos en el futuro. Es mejor defender aquellos objetivos que influyen directamente en el éxito de nuestro proyecto que frustrarnos pensando en aquellos sobre los que no tenemos control.

https://agilewarrior.wordpress.com/2010/11/06/the-agile-inception-deck/

8. Tómale las medidas

Se trata de un ejercicio de estimación, de establecer una medida grosso modo o de orden de magnitud de lo que va a durar o el esfuerzo que puede suponer. No se puede saber exactamente cuánto va a durar el proyecto antes de empezar pero sí si estamos hablando de tres meses, un año o más.



“Esto es una estimación, no un compromiso, no deja de ser una fantasía de control”.

MECÁNICA

Como equipo tenemos que hacer algún tipo de estimación aunque sea a alto nivel. La lista de los NOes viene muy bien aquí. El objetivo es conseguir una estimación aproximada para que los stakeholders se puedan hacer una idea. Lo importante aquí no es la precisión, sino hacernos una idea de si el proyecto es remotamente alcanzable con los recursos de los que disponemos. El objetivo es establecer un control de las expectativas.

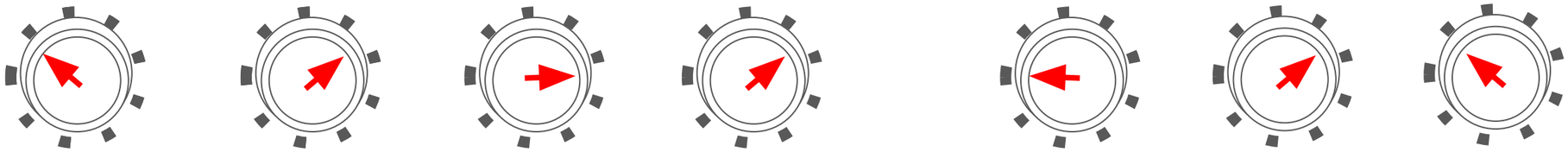
https://agilewarrior.wordpress.com/2010/11/06/the-agile-inception-deck/

9. Ser claros en qué vamos a dar

Los proyectos tienen sus **factores clave** de ejecución: **tiempo, alcance, dinero** y, por supuesto, **calidad**. Es fundamental saber cuáles de estos factores son inamovibles y en cuáles hay flexibilidad. Desde luego, no todos pueden ser fijos.

Identificar riesgos en los que podemos influir

Todos los proyectos tienen palancas del tipo dinero, fechas, alcance y calidad. Mejor saber de todos cuales tienen más peso y cuales son flexibles. En unos proyectos el dinero es intocable, en otro las fechas de entrega. Lo mejor es tener esta conversación al principio, para cuando llegue la hora de apretar, si hay flexibilidad en el alcance (preferible) o darle vueltas a la fecha de entrega. Otro punto a tener en cuenta es que cosas podrían poner este proyecto en entredicho: Simplicidad, facilidad de uso, seguridad, rendimiento, etc. Todo esto hay que anotarlo y que todo en mundo lo tenga en el radar antes de empezar.



Presupuesto Alcance Tiempo Calidad Facilidad Seguridad Simplicidad

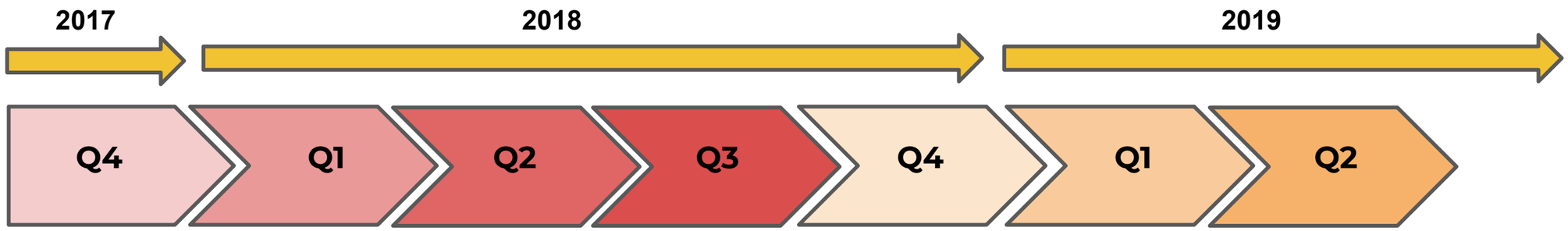
MECÁNICA

Es importantísimo tener esta conversación al principio para llegado el momento saber si hay flexibilidad en la duración o la hay en el alcance. Y otra cosa igual de importante a tener en cuenta es qué aspectos críticos en el proyecto cuya falta pondría en riesgo la continuidad. Por ejemplo, facilidad de uso, simplicidad, seguridad, rapidez...

https://agilewarrior.wordpress.com/2010/11/06/the-agile-inception-deck/

10. ¿Cuánto va a costar?

La pregunta del millón: ¿Cuánto tiempo y cuánto dinero?. Si el budget está ya cerrado y es inamovible, se puede hacer unos cálculos básicos que nos dará una idea general.



5 personas, 5 meses, 400k€

MECÁNICA

Esta es la mejor oportunidad para valorar el tipo de equipo que se va a encargar de ello. En este momento se pueden establecer las expectativas respecto al tamaño del equipo, sus skills, sus capacidades crossfunctional necesarias para afrontar el proyecto.

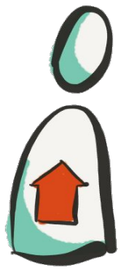
https://agilewarrior.wordpress.com/2010/11/06/the-agile-inception-deck/

Conclusiones

Hacer un Agile Inception Deck no es magia. Consiste en juntar a la gente adecuada y hacerse las preguntas difíciles para luego poder compartir esa información con los stakeholders, con el equipo y con todos los que lo deban saber.

MECÁNICA

Este proceso puede durar entre un par de días y un par de semanas y puede cubrir un periodo cercano al medio año. Habría que aplicarlo de forma recurrente al acercarse el final de lo que hayamos cubierto.



Product Owner Proxy

Su objetivo es ayudar al PO cuando su disponibilidad no es total, participando en la definición y el detalle de los requisitos.
Suele asociarse con los roles de business analyst, funcional, etc, posee un gran conocimiento de la solución o del área de negocio específica, y complementa la visión del Product Owner a un nivel de detalle mayor (incluso técnico).

¿CUÁLES SON SUS FUNCIONES?

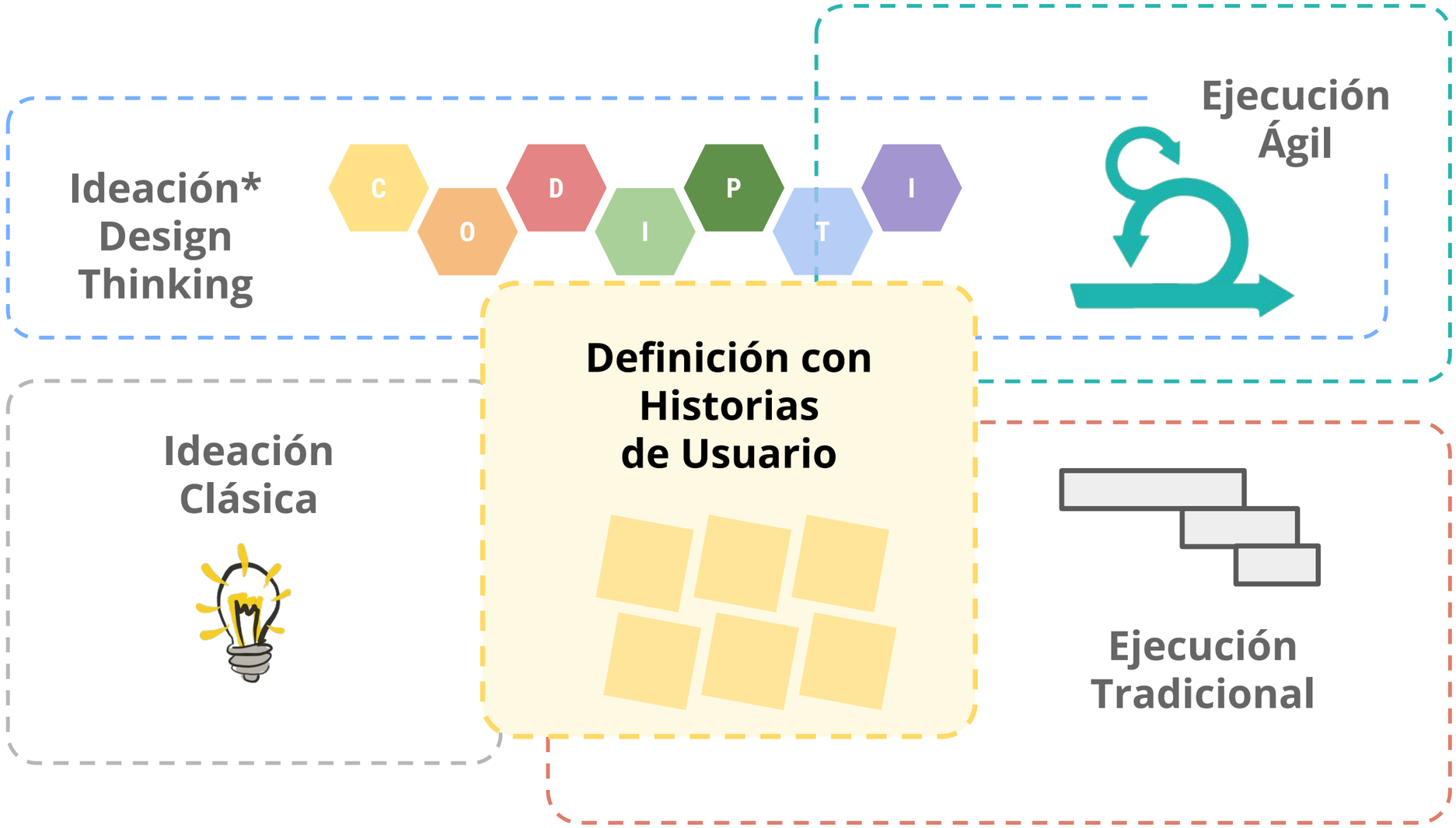
- Suplir la falta de disponibilidad del Product Owner.
- Asistir a las ceremonias especificadas por Scrum para el Product Owner.
- Realizar las tareas especificadas por Scrum para el Product Owner.
- Tener cierta capacidad de toma de decisiones para no convertirse en un cuello de botella
- Trasladar y sincronizar las decisiones tomadas con el Product Owner en su ausencia.

¿QUÉ NO ES?

- Un lacayo del PO
- Un escriba del PO

autentia

Ideación, definición y ejecución de un proyecto/producto



**Design Thinking engloba un espectro más amplio que la simple ideación.*

Independientemente del sistema de ideación de nuestro producto/proyecto y de la forma en la que construyamos nuestro software, las historias de usuario nos permiten mejorar la definición y reducir el paraguas de oportunidades.

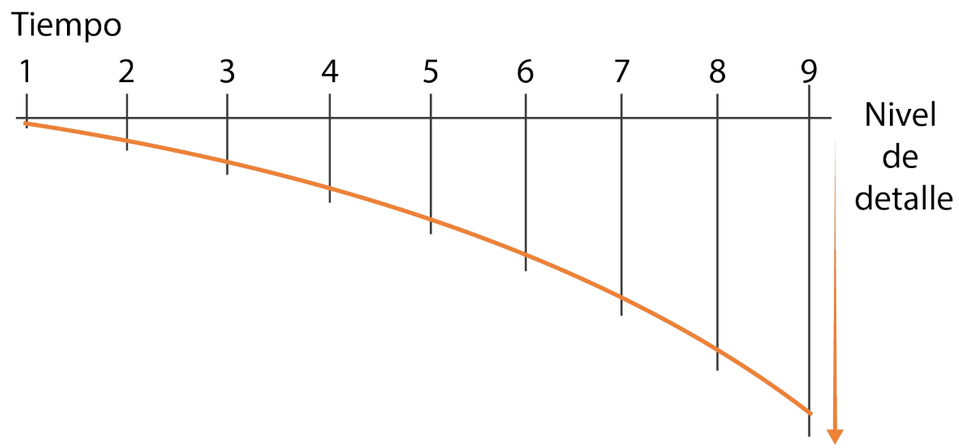
Modelo de definición de un proyecto con Historias de Usuario

PASOS PARA REALIZAR LA DEFINICIÓN

1. Definir objetivos + KPIs (1 o 2 no más).
2. Identificar actores.
3. Dibujar diagrama de procesos (desechable).
4. Redactar las historias (card).
5. Modelar mediante User Story Mapping.
6. Podar funcionalidad.
7. Priorizar historias.
8. Estimar de forma relativa.

---

9. Descomponer historias y obtener agrupadores.
10. Detallar historias y agrupar funcionalmente.
11. Identificar dependencias.
12. Estimar considerando los detalles.



Hasta el punto 8 nos encontramos en una curva de aceleración, esto es, la inversión de tiempo vs el nivel de detalle es muy baja. Nos ayuda a tener un orden de magnitud.

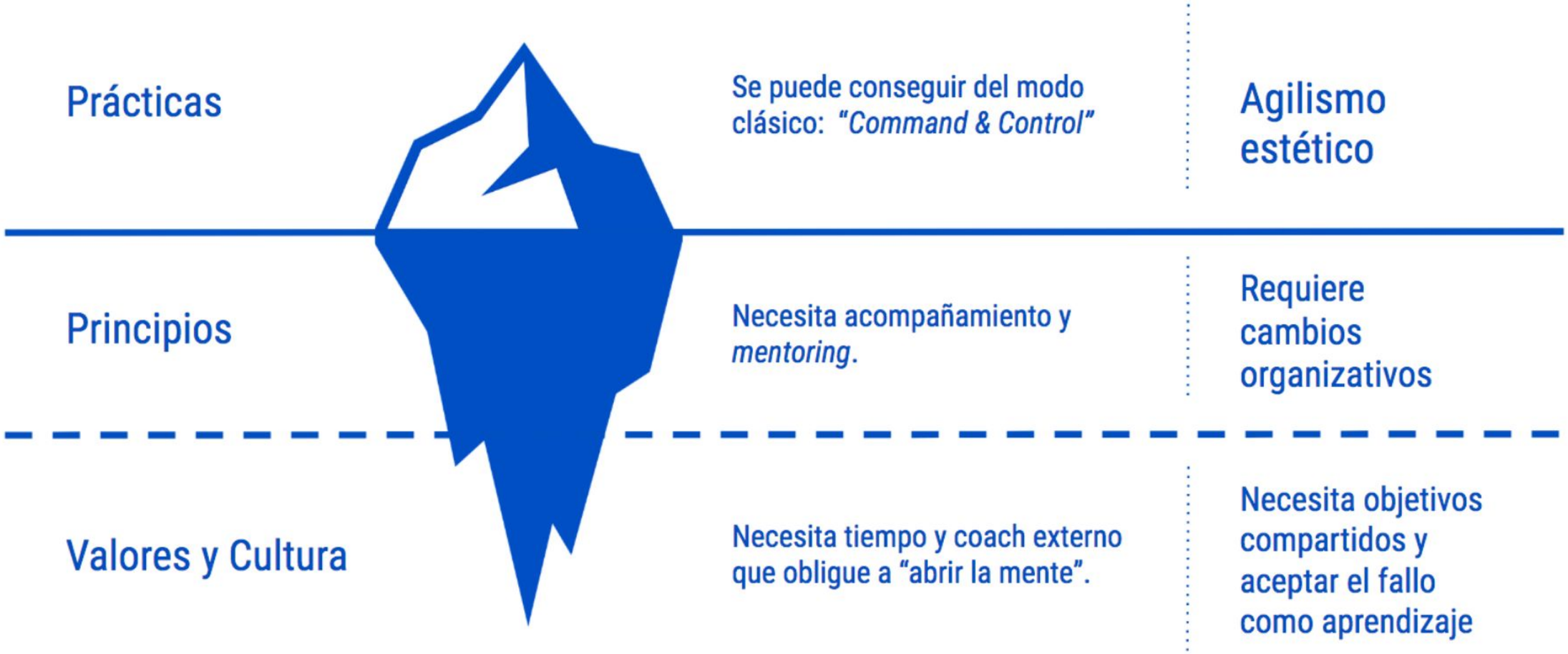
A partir del punto 9, comenzamos a ganar detalle y granularidad, la velocidad se ve reducida si comparamos en amplitud y no en detalle. Mejoramos definición y predictibilidad.

El mayor reto es mantener un equilibrio entre necesidad de información y exceso de información. Estaríamos generando desperdicio.

Un poco de humor

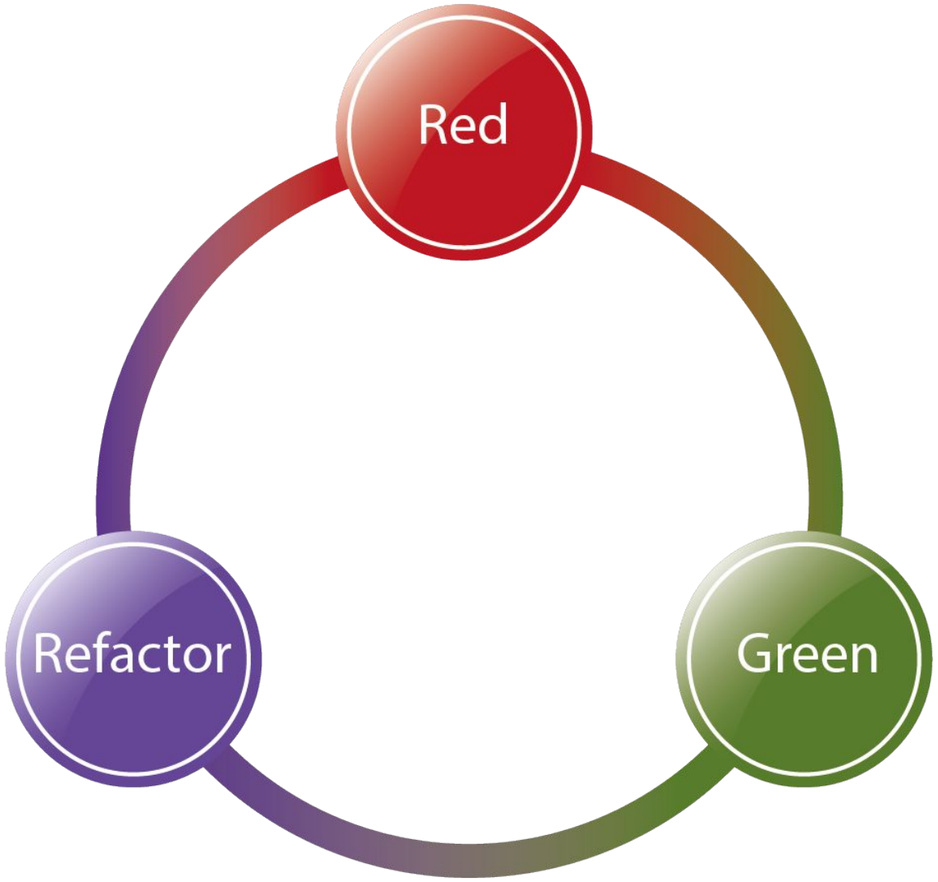
autentia THINKING WITHYOU

Adopción del agilismo



TDD

El desarrollo dirigido por pruebas o **TDD** (Por sus siglas en inglés **Test Driven Development)** es una técnica de programación que se centra en escribir las pruebas antes de programar la funcionalidad, siguiendo el ciclo: “falla, pasa, refactoriza” [red, green, refactor] intentando así mejorar la calidad del software que producimos.



CÓDIGO DE COLORES

ROJO: existe la prueba, pero no se ha codificado la funcionalidad.

VERDE: código mínimo de la funcionalidad programado para pasar el test.

REFACTOR: eliminamos redundancia y mejoramos la implementación, con técnicas de refactoring y principios SOLID.

ALGUNOS APUNTES

- Se podría considerar la suma entre **TFD (test-first development)** y **refactoring**.
- Es necesario como pre-condición disponer de una definición formal de los **requisitos**.
- Ayuda a **diseñar** los módulos que no existen todavía.
- Promueve la **automatización** de pruebas**.**
- Ayuda a **perder el miedo** a tocar el código.

Integración continua

La integración continua es una práctica de desarrollo de software mediante la cual los desarrolladores combinan los cambios en el código en un repositorio central de forma periódica, tras lo cual se ejecutan versiones y pruebas automáticas. La integración continua se refiere en su mayoría a la fase de creación o integración del proceso de publicación de software.

PROCESO

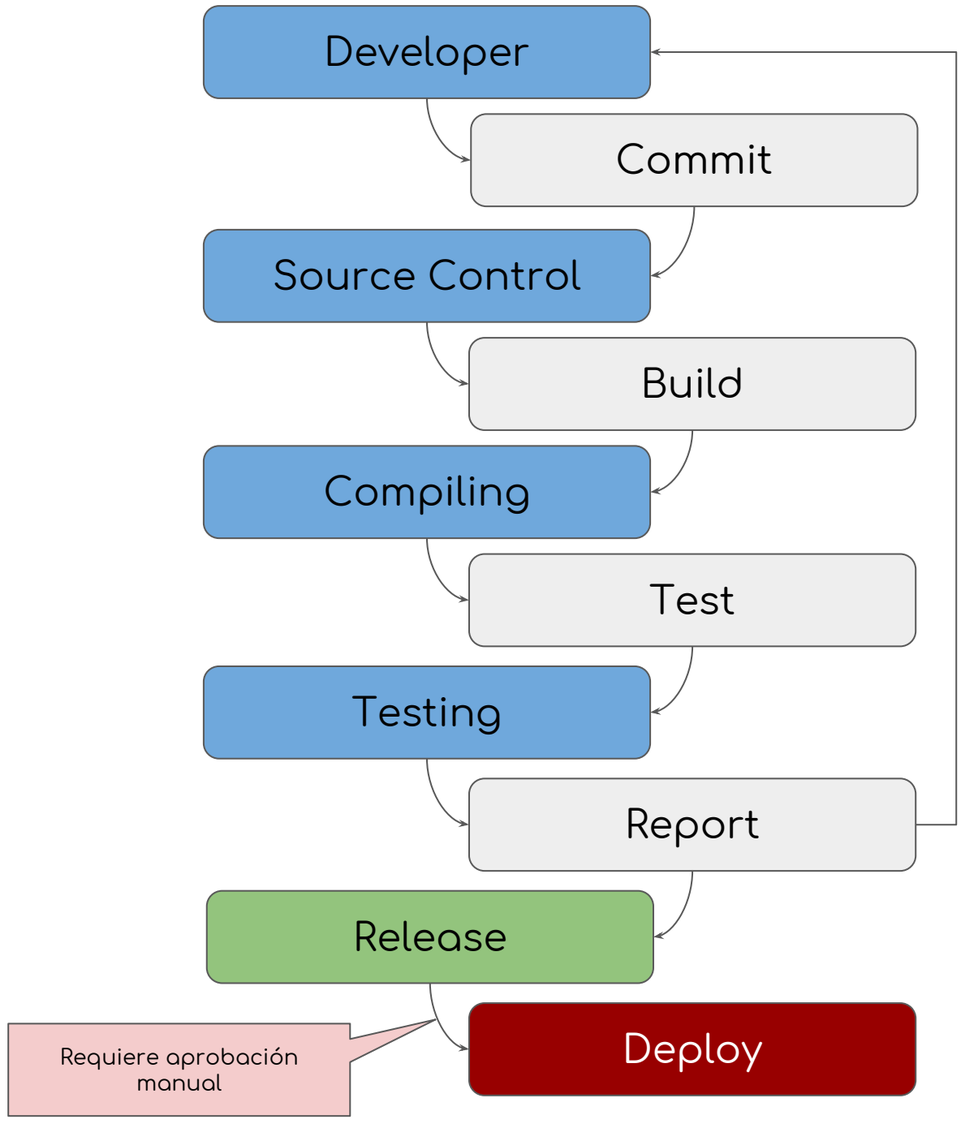
1. Se descargan las fuentes desde el control de versiones (Git, Subversion, Mercurial, etc.) cada cierta cantidad de horas.
2. Se compila nuevamente.
3. Se ejecutan las pruebas.
4. Se generan informes.

VENTAJAS

- Los desarrolladores pueden detectar y solucionar problemas de integración de forma continua, evitando el caos de última hora cuando se acercan las fechas de entrega.
- Disponibilidad constante de una versión para pruebas, demos o lanzamientos anticipados.
- Ejecución inmediata de las pruebas unitarias.
- Monitorización continua de las métricas de calidad del proyecto.

Entrega continua

La entrega continua es una práctica de desarrollo de software mediante la cual se crean, prueban y preparan automáticamente los cambios en el código y se entregan para el entorno previo a producción. Amplía la integración continua al implementar todos los cambios en el código en un entorno de pruebas y/o de pre-producción después de la fase de creación. Cuando la entrega continua se implementa de manera adecuada, los desarrolladores dispondrán siempre de un artefacto listo para su implementación que se ha sometido a un proceso de pruebas estandarizado y deja el desarrollo listo para ser desplegado en producción.



VENTAJAS

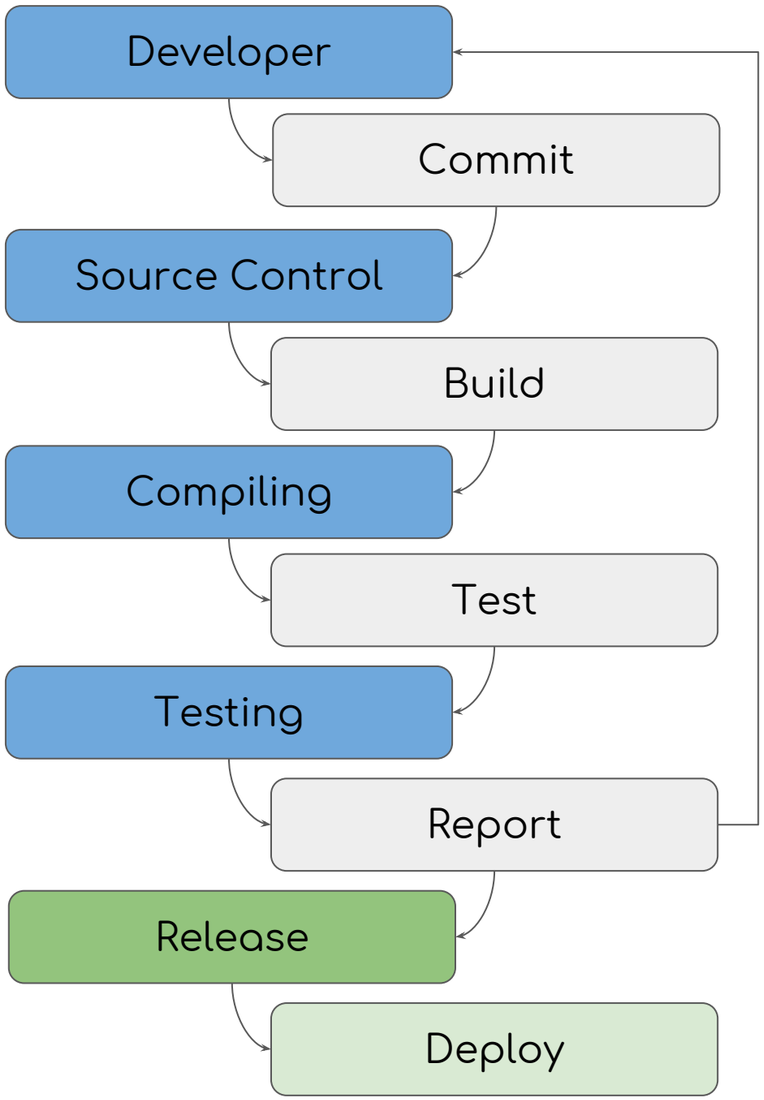
1. Automatiza el proceso de publicación de software.
2. Mejora la productividad del desarrollador.
3. Encuentra y arregla los errores con mayor rapidez.
4. Entrega las actualizaciones con mayor rapidez

OBSERVACIONES

Con la entrega continua, todos los cambios en el código se crean, se prueban y se envían a un entorno de almacenamiento o pruebas de no producción. Pueden efectuarse varias pruebas al mismo tiempo antes de la implementación en producción. En el último paso, el desarrollador aprueba la actualización para su envío a producción cuando está listo.

Despliegue continuo

El despliegue continua es una práctica de desarrollo de software mediante la cual se crean, prueban y preparan automáticamente los cambios en el código y se entregan para el entorno de producción. Amplía la entrega continua al implementar todos los cambios en el código en un entorno de pruebas y/o de producción después de la fase de creación. Cuando se utiliza de manera adecuada, los desarrolladores dispondrán siempre de un entorno listo para entregar cambios de manera automática.



VENTAJAS

1. Automatice el proceso de publicación de software.
2. Mejore la productividad de desarrollo.
3. Encuentre y arregle los errores con mayor rapidez.
4. Entregue las actualizaciones con mayor rapidez

OBSERVACIONES

El proceso se diferencia de la implementación continua en que en el segundo caso el envío a producción se efectúa automáticamente, sin aprobación explícita.

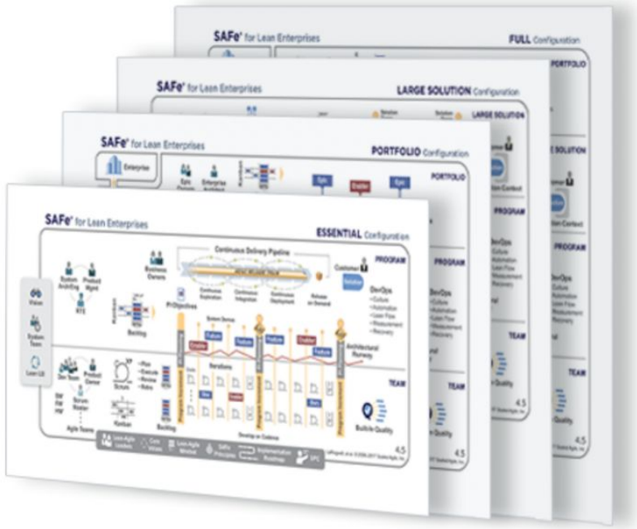
SAFe

Según su creador Dean Leffingwell, **SAFe, Scaled Agile Framework** es una base de conocimientos libre para la aplicación de patrones probados para desarrollar software implementando Lean-Agile en proyectos a escala empresarial. Su objetivo principal es proporcionarle a las organizaciones una guía general para aumentar la productividad en el desarrollo de software a todos los niveles de una organización.

CONFIGURACIONES DE SAFe

La nueva versión de SAFe está pensada para ser un marco configurable y orientado a satisfacer las necesidades de cualquier producto o servicio. Se definieron cuatro configuraciones que permiten proporcionar un enfoque ajustado a las características de cada empresa:

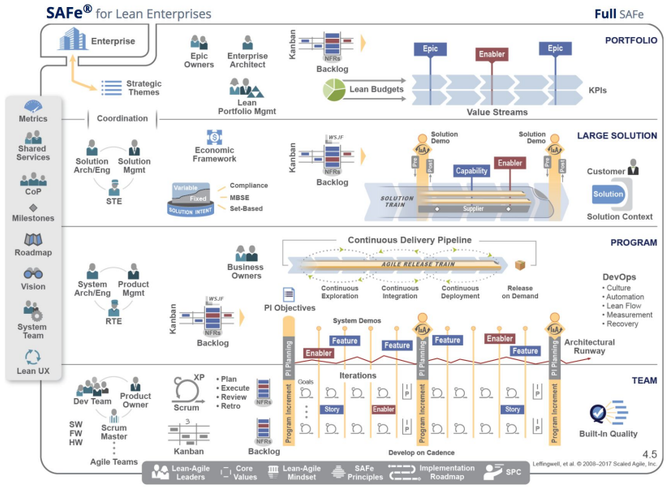
- **ESSENTIAL SAFe:** Es la configuración más básica del framework y provee de los elementos mínimos.
- **LARGE SOLUTION SAFe:** Pensada para empresas que construyen soluciones grandes pero que no requieren de las bondades del nivel Portfolio para gestionarlas.
- **PORTFOLIO SAFe:** Es la configuración ideada para corporaciones con múltiples productos con un mínimo número de dependencias, pero que requiere de un nivel de Portfolio para la coordinación, estrategia, inversión y gobernanza.
- **FULL SAFe:** Representa la configuración más compleja de SAFe dado que es compatible con la construcción de grandes soluciones que normalmente requieren integrar a cientos de personas para su desarrollo.



NIVELES DE SAFe

En la versión más reciente de SAFe se proporcionan cuatro niveles sobre los que se pueden aplicar prácticas Lean-Agile:

- Un nivel para los **Equipos** donde se proporciona un modelo para equipos ágiles basados en Scrum y prácticas XP.
- Un nivel de **Programa** donde los esfuerzos de múltiples equipos ágiles se integran para ofrecer valor a la empresa en un Agile Release Train.
- Un nivel de **Portfolio**, donde los programas se alinean con las estrategias del negocio y su intención de inversión.
- Y finalmente un nivel opcional para **Flujo de Valor** donde se orquestan la entrega de valor de distintos Agile Release Train.



Less

Large-Scaled Scrum es un framework de desarrollo de productos que extiende Scrum con directrices y reglas de escalado sin perder los objetivos originales de Scrum. Es importante diferenciar entre, LeSS y LeSS Huge.

CARACTERÍSTICAS

LeSS mantiene muchas de las prácticas e ideas de Scrum de un solo equipo:

- Un solo Product Backlog, un Product Owner y un Sprint.
- Una *Definition of Done (DoD)* para todos los equipos.
- Un incremento con un producto potencialmente entregable al final de cada Sprint.
- Varios equipos multifuncionales.

En LeSS, todos los equipos trabajan juntos en un Sprint común para entregar un producto común, cada Sprint. ¿Qué es diferente en LeSS?

- Sprint Planning 1 que incluye a miembros de todos los equipos.
- Daily Scrum con la presencia de algún miembro de otros equipos para aumentar la colaboración entre equipos.
- PRB global y opcional. Reunión de Product Backlog Refinement con el Product Owner y miembros de todos los equipos.
- Sprint Reviews con miembros de todos los equipos.
- Retrospectiva global con el Product Owner, los Scrum Masters y miembros de todos los equipos.

LeSS Huge, recomendable para adopciones de más de ocho equipos, añade:

- Un nuevo rol **Product Owner de Área**. Un artefacto, **Área Backlog** y la ejecución de LeSS Sprints en paralelo.

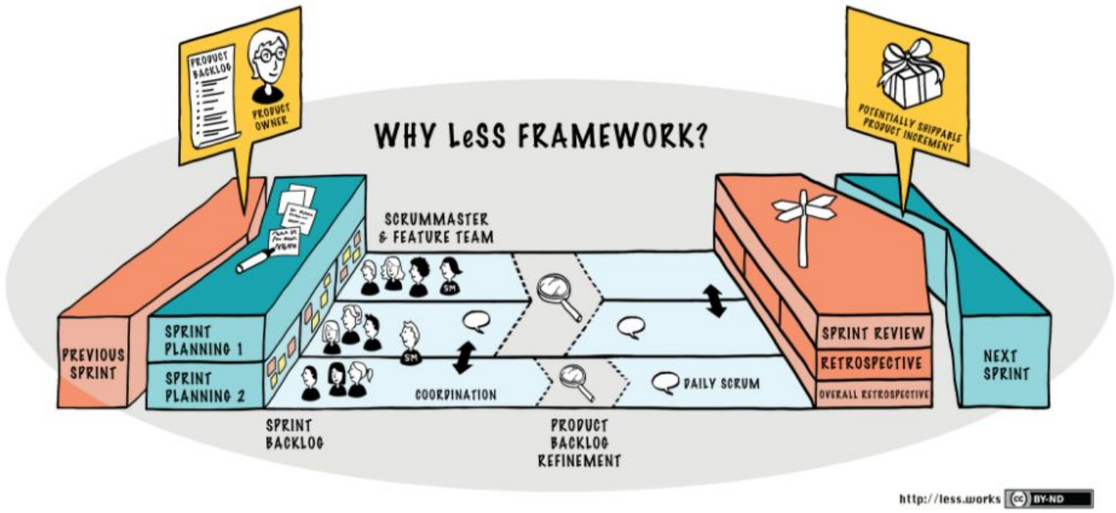


Imagen de The LeSS Company B.V.

Nexus

Nexus es un framework de escalado Agile propuesto por Scrum.org, consistente en roles, eventos, artefactos y reglas que unen y entrelazan conjuntos de tres a nueve equipos trabajando en un único Product Backlog para construir un incremento integrado que cumple los objetivos.

CARACTERÍSTICAS

Nexus consiste en múltiples equipos Scrum multifuncionales trabajando juntos para liberar un incremento integrado potencialmente entregable al final de cada Sprint como mínimo. Basándose en sus interdependencias, los equipos se auto-organizan y eligen los miembros más adecuados para hacer el trabajo.

NOVEDADES

El Nexus incorpora unos nuevos roles que se integran en un equipo adicional: El Nexus Integration Team, que se compone de:

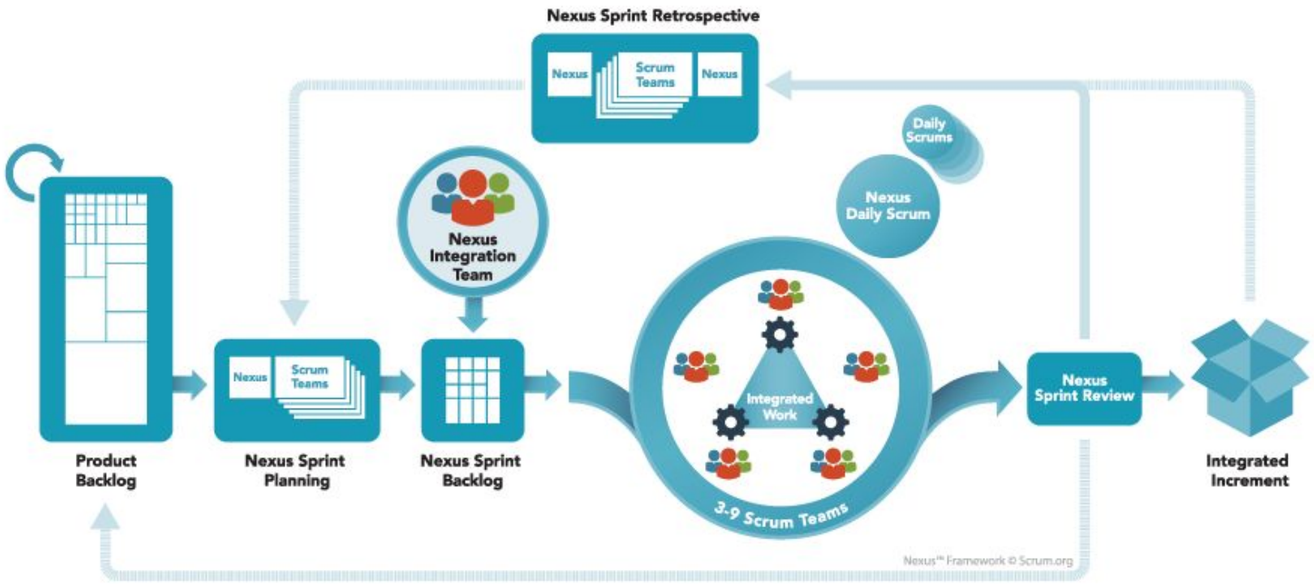
- Un Product Owner, un Scrum Master y los miembros del **Integration Team**.

Este equipo realiza la integración de Nexus resolviendo problemas técnicos y no técnicos de los equipos. Estos miembros también son miembros de equipos individuales.

FLUJO

Flujo de procesos Nexus con equipos Scrum entrelazados:

- Refinar el Product Backlog.
- Nexus Sprint Planning.
- Trabajo del equipo de desarrollo.
- Nexus Daily Scrum.
- Nexus Sprint Review.
- Nexus Sprint Retrospective.



Scrum Studio

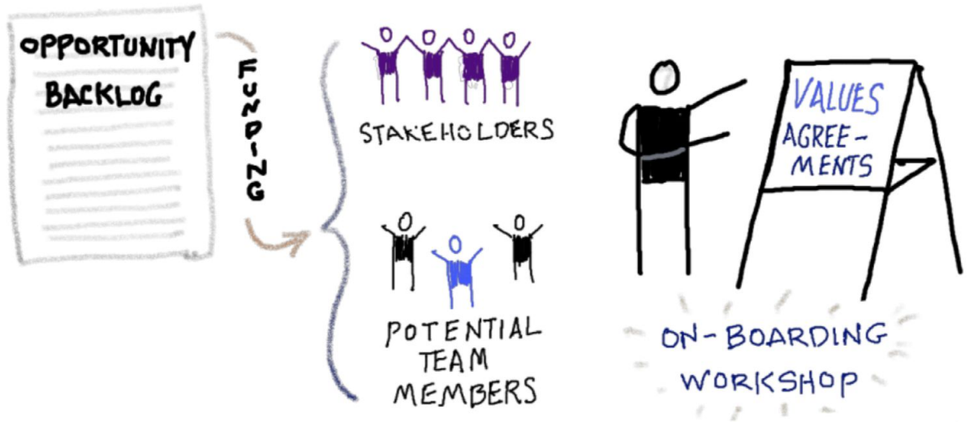
Scrum Studio no es un proceso ni un framework, es un conjunto de patrones adaptable que ha ido evolucionando junto a Scrum y Nexus a lo largo de los años. Se basa en la escisión de un equipo de la organización matriz como piloto que posteriormente irá creciendo y absorbiendo la organización para realizar un escalado horizontal que no esté condicionado por la estructura organizativa.

CARACTERÍSTICAS

Es un modelo **flexible** basado en la **experimentación** que favorece el que los equipos puedan crear **productos innovadores** con **Scrum** y **Nexus**. Según va creciendo la necesidad, *Scrum Studio* se amplía añadiendo nuevos grupos y productos, y si es necesario, añadiendo otros *Scrum Studios* diseñados específicamente para conjuntos de productos concretos.

Si un producto ya no necesita el enfoque empírico porque ya satisface a los clientes o porque llega a su fin de ciclo de vida, puede trasladarse desde el modelo Scrum Studio a una parte de la organización que ya no necesita ese enfoque.

- Es un modelo basado en **resultados**.
- Fuertemente enfocado en **equipos de alto rendimiento**.
- Apoyando y protegiendo a esos equipos para que se puedan centrar en la innovación basada en el **empirismo**.
- Permitiendo el **crecimiento** de los individuos y de los equipos invirtiendo en comunidades.



Spotify

Es el modelo de escalado ágil que se desarrolló e implantó en la creación del producto y la empresa Spotify. Se basa en la organización de equipos auto-organizados bajo el concepto de Squads, equiparable a un Scrum Team. Los Squads suelen fomentar la aplicación de principios Lean Startup cuyo lema es “Think it, bulid it, ship it, tweak it”.

CARACTERÍSTICAS

- Operan autónomamente en contacto directo con StakeHolders y sin bloquearse entre squads como si fueran una mini-startup.
- Dedican el 10% del tiempo en “hacking days”, promoviendo la innovación y el aprendizaje.
- Cada Squad dispone de un Product Owner y además, un agile coach les apoya para desarrollarse.
- Los squads se agrupan en tribus que trabajan en áreas relacionadas y operan como si fueran unidades capaces de generar valor por sí mismas.
- Para coordinar experiencia y conocimiento entre squads existe el concepto de Chapter, donde los miembros de squads con skills similares se agrupan y comparten información y conocimiento.
- El Guild, es un concepto similar pero a nivel de toda la organización.

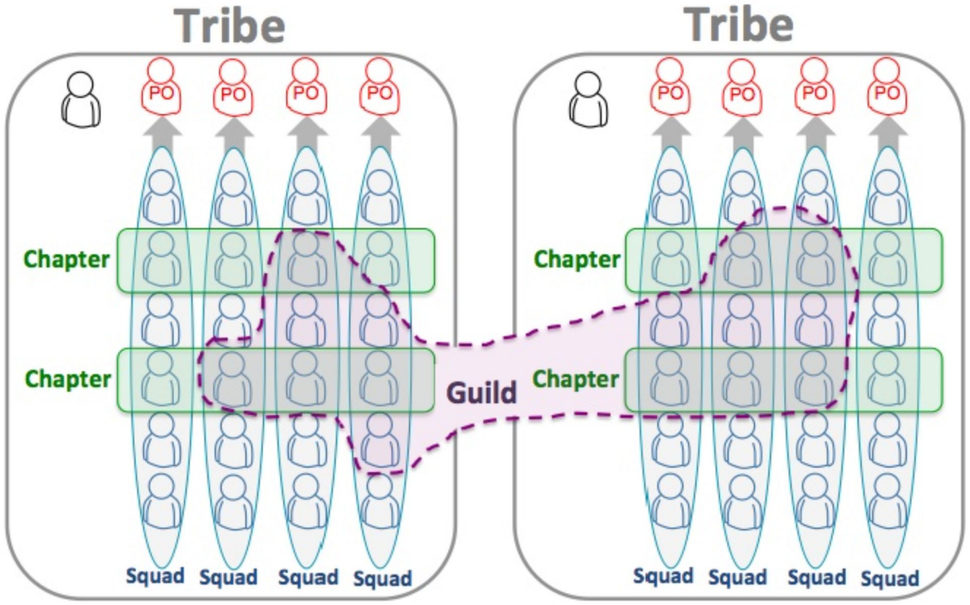


Imagen de Henrik Kniberg & Anders Ivarsson Scaling @ Spotify with Tribes, Squads, Chapters & Guilds

Lean Manufacturing

Es un método ágil centrado en la estrategia y su origen está en la empresa de manufacturación y la posterior adaptación al desarrollo de software. Este método tiene **tres objetivos** principales: **reducir** drásticamente el **tiempo de entrega de un producto**, reducir su **precio** y reducir el **número de defectos** o *bugs*.

PRINCIPIOS EN LOS QUE SE BASA LEAN

1. Elimina desperdicios	**Elimina todo lo que no aporte valor**. Es crucial no hacer algo que no sea específicamente lo que se espera que se haga. Si el cliente no lo necesita o no es lo que quiere en ese momento, directamente puede y debe ser eliminado. El principal objetivo de este principio es ayudar al equipo a **no perder el foco**.
2. Optimizar el todo	**El cliente necesita un todo**. No le aporta mucha información ver pequeños trozos de lo que espera sin saber cómo va a encajar el puzle final. Necesita, desde un principio, ir viendo una foto global de lo que va a recibir.
3. Construir con calidad	**La calidad no es opcional**. No debemos esperar a probar nuestro producto en la fase final ya que la resolución del problema será, en sí misma, un problema.
4. Aprender constantemente	**La creación es un proceso experimental**. La clave está en ir aprendiendo y entendiendo lo que se necesita a medida que se construye ya que no podemos adivinar el futuro.
5. Reaccionar Rápido	Para poder reaccionar rápido es necesario **trabajar con iteraciones cortas** de forma que los comentarios del cliente sean frecuentes y se cubran sus expectativas y necesidades en cada momento.
6. La mejora continua	Para mejorar constantemente , **no se debe tolerar ningún fallo**, por pequeño que éste sea. Se debe sacar a la luz rápidamente para ser corregido y de esta forma se estará fomentando aprender de estos errores y evitar que se repitan y que se hagan mayores.
7. Cuidar el equipo de trabajo	**Un equipo de trabajo debe estar motivado** y esto se consigue proporcionándole cierto grado de autonomía para tomar decisiones, ofreciendo a cada persona la posibilidad de aprender y mejorar y asegurándose de que su trabajo es valioso en todo momento.

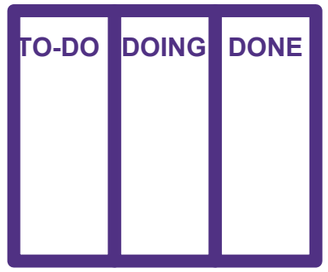
Lean: Elimina desperdicios

Dentro del primer principio (*"Elimina desperdicios"*), Lean define como un desperdicio todo aquello innecesario, es decir, todo aquel añadido del que se puede prescindir. Asi, Lean destaca ocho (7+1) tipos de desperdicios:

TIPOS DE DESPERDICIOS

1. Espera	Esperar en una reunión a que lleguen todos. Esperar porque alguien apruebe todo del diseño.
2. Sobre-producción	Hacer esquemas y presentaciones para explicar los mismos conceptos de muchos modos distintos.
3. Sobre-ingeniería	Definir el sistema para cuando quiera ampliarlo, por aprovechar el tiempo. Añadir capas de complejidad que no son necesarias al principio.
4. Stock	Detallar requisitos que no vamos a construir en meses. Hacer entregas de muchos cambios a la vez.
5. Transporte	Cambios de contexto de proyectos a fuego en producción.
6. Movimiento	Falta de automatismos para mejorar la entrega continua.
7. Defectos	Poco cuidado por la calidad.
+1 Talento	El trabajo lo debe realizar la persona más adecuada teniendo en cuenta la habilidad y el impacto para el resto del equipo. Los managers tiene que ser líderes al servicio del equipo fomentando el desarrollo personal y profesional, y simplemente ser los mejores ejecutores.

http://blog.jorgemuria.com/?p=443



Principios Básicos

Es una **técnica** que se creó en **Toyota** para optimizar la producción de automóviles. Se utiliza para gestionar cómo se van completando las tareas y avanzando el **trabajo** de una manera **visual** a través del movimiento de **tarjetas**.

QUÉ ES

- Una herramienta.
- Un método para mejorar la entrega de valor.
- Una metodología para la gestión del cambio.

PRINCIPIOS DE GESTIÓN DEL CAMBIO

- Empieza donde estés.
- Establece cambios evolutivos.
- Promueve el liderazgo en todos los niveles.

PRINCIPIOS DE GESTIÓN DEL SERVICIO

- Entiende y focalízate en las necesidades y expectativas de los CLIENTES.
- Deja que la gente se auto-organice alrededor de las tareas.
- Evoluciona las POLÍTICAS para mejorar los resultados.

LAS 6 PRÁCTICAS

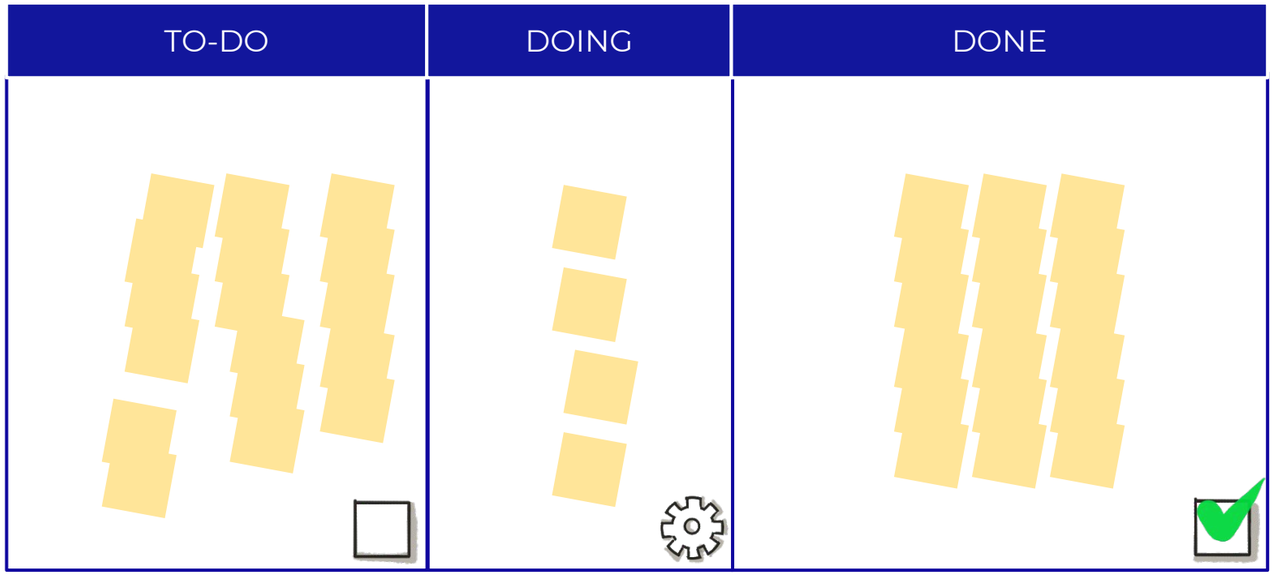
1. Visualiza el trabajo.
2. Limita tu WIP (work in progress).
3. Gestiona el flujo.
4. Haz explícitas las políticas.
5. Practica la mejora continua.
6. Implementa ciclos de feedback.

QUÉ NO ES

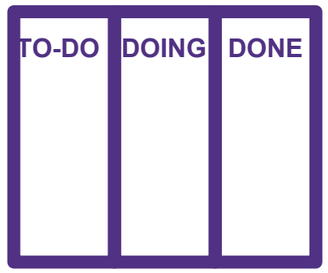
- Un procedimiento
- Un proceso

Kanban

autentia THINKING WITHYOU



TEMAS/ HISTORIAS	TAREAS PENDIENTES	EN PROCESO				
		IMPEDIDAS	N DÍAS	MAÑANA	HOY	TERMINADAS
		IMPEDIDAS				



Roles (no oficial de Kanban)

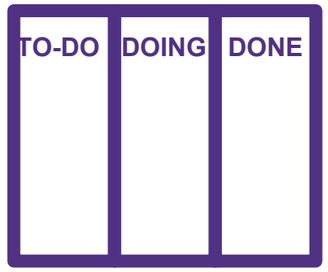
Kanban es y sigue siendo el método **“empieza donde estés”**, donde en un inicio nadie recibe nuevos roles, responsabilidades o cargos. **No existen nuevos roles** pero la práctica común ha propiciado la aparición de ellos y ahora se definen en el método en sí mismo.

PROPÓSITO

Más que asignarle a alguien un cargo, lo importante es el **propósito** del rol. Se debe ver el rol como un “gorro” que cualquiera se puede poner y quitar, de modo que cuando se asume uno de los siguientes roles, se pueden llevar a cabo una serie de funciones asignadas a cada rol.

ROLES EN KANBAN

- **Service Request Manager:** Se encarga de gestionar la demanda y los requisitos dentro del sistema Kanban, manejando las relaciones con los stakeholders y fomentando la transparencia dentro del sistema en torno a la priorización del trabajo. Además, facilita, selecciona y ordena los elementos de trabajo en la Reunión de revisión de la cartera de trabajo (Replenishment Meeting). Alternativamente, este rol se puede llamar *Product Manager*, *Product Owner* o *Service Manager*.
- **Service Delivery Manager:** Es responsable del flujo de trabajo dentro de un sistema Kanban y/o determinados ítems de trabajo y facilita el Kanban Meeting y el Delivery Planning. Algunos nombres alternativos son *Flow Manager*, *Delivery Manager* o incluso *Flow Master*.



Políticas explícitas

Una práctica imprescindible en Kanban es hacer **explícitas** las políticas. Las políticas *son decididas por aquellos cuyo trabajo se ve impactado en el tablero*, sin haber un rol que las decida. Son reglas que indican cómo se trabaja (Criterio Pull, Priorización, Límite WIP) y se deben ubicar a la vista de todos.

TIPOS DE POLÍTICAS

Las políticas explícitas pueden ser variadas y de muchos tipos, tenemos pues:

- Políticas por columna → ¿Quién mueve qué? ¿Cómo se deciden las prioridades?
- Políticas para todo el sistema → ¿Qué se escribe en un ticket? ¿Cómo funciona el tablero? ¿Cómo se revisan las replenishment?
- Políticas por clases de servicio

EJEMPLOS DE POLÍTICAS

Políticas por columna:

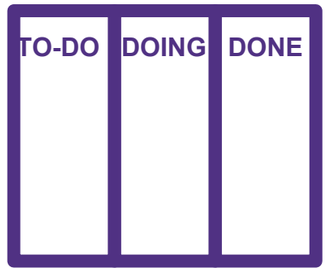
- Todo pasa por el Backlog primero *(Criterio Pull)*
- Toda tarjeta en la parte superior de la columna es más prioritaria que otra tarjeta que esté debajo *(Priorización)*

Políticas para todo el sistema:

- Tareas cuya duración sea de 15’ o menos no se agregan al tablero
- El tablero se actualiza una vez al día y se hará durante la Kanban Daily Meeting

Políticas por clases de servicio:

- Expedite
- Standard
- Fixed Date
- Intangible



Clases de Servicio

Las Clases de Servicio (o Riegos) son una clasificación de los posibles ítems de trabajo que pueden surgir basadas en diferentes expectativas del cliente, valor relativo, riesgo o **Cost of Delay**. En función de esta clasificación a cada Clase de Servicio se le asigna un **conjunto de políticas**.

CLASES DE SERVICIO (O RIESGOS)

La práctica es identificar aquellos ítems que sean más críticos basados en su **Cost of Delay**. La definición de *Cost of Delay* la podemos ver desde dos perspectivas diferentes, la del coste y la del valor siendo el significado el mismo:

1. **Perspectiva del Coste**

¿Cuánto me costaría si este ítem, que tengo que entregar en una fecha determinada, estuviera retrasado? ¿Que ocurriría si no termino el trabajo a tiempo?.

2. **Perspectiva del Valor**

La cantidad de valor del elemento que se pierde debido al retraso en la implementación durante un periodo de tiempo.

La clasificación más utilizada de Clases de Servicio es: *Expedite, Standard, Fixed Date, Intangible.*

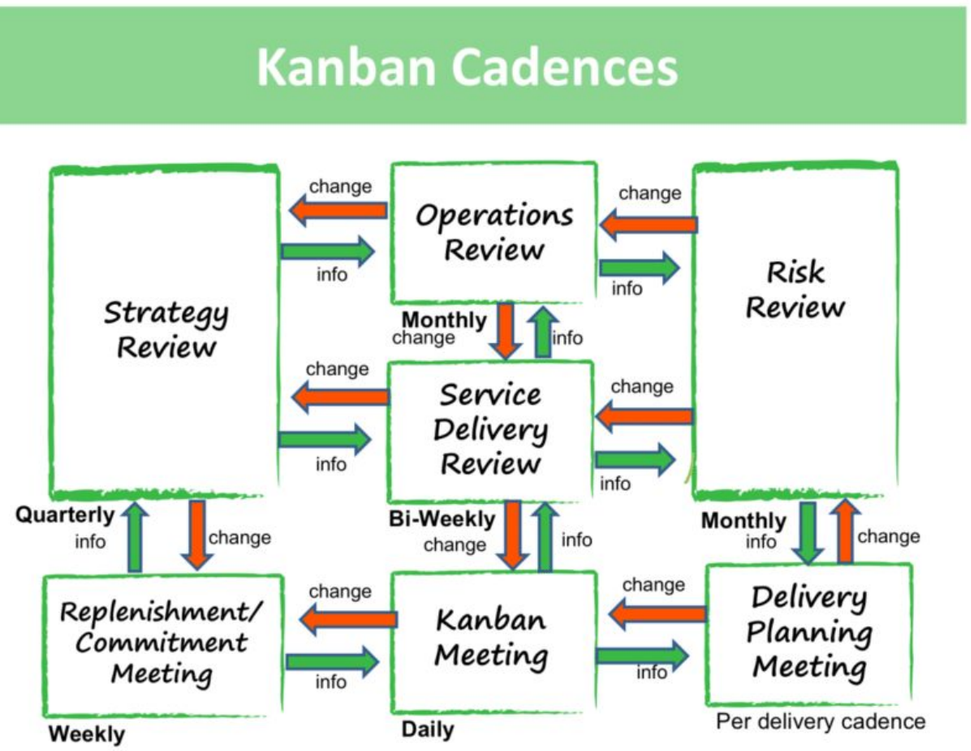
Otra clasificación diferente puede ser: *Urgent, High, Normal y Low*

En determinados casos las **Clases de Servicio** pueden afectar a la **prioridad**.

Cadencias

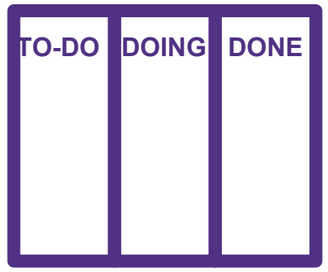
En Kanban se definen 7 cadencias o reuniones, interconectadas entre sí formando circuitos de retroalimentación para la evolución y gestión del flujo de trabajo. **“Cadencia”** también puede referirse al periodo de tiempo entre cada revisión – un día laboral o un mes, por ejemplo. Elegir la cadencia adecuada depende del contexto y es crucial para un buen resultado.

CADENCIAS



Source: Kanban Coaching Professional Masterclass, David J. Anderson, Barcelona 2017

Esquema de las siete cadencias en Kanban: muestra las frecuencias aconsejables para las revisiones en un contexto típico de empresa o servicios múltiples.



Cadencias - Getting Things Done

De todas las reuniones que se definen en Kanban existen 3 que por sus objetivos se deben priorizar. Al realizar dichas reuniones conseguiremos que los ítem que se inician se terminen (Getting Things Done). La descripción de las reuniones son sólo configuraciones típicas, debes ajustar la cadencia de las reuniones, duración, asistentes basándote en las necesidades de tu empresa pudiéndose ajustar con el tiempo.

CADENCIAS PARA CONSEGUIR EL "GETTING THINGS DONE"

- **Replenishment Meeting**

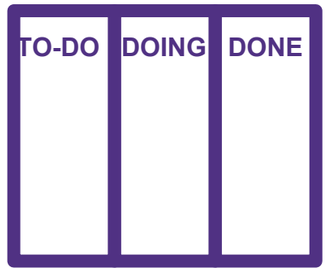
Con periodicidad **semanal**, el objetivo de esta reunión es establecer prioridades y seleccionar cuales son las siguientes prioridades de trabajo. Se mueve trabajo a los *commitment point* del sistema Kanban, por servicio. Dicho de otro modo, son los compromisos que el equipo toma con negocio/usuario final; los elementos con los que se compromete a entregar tras una cadencia determinada (vendrían a ser los elementos de un Backlog).

- **Daily Kanban**

Con periodicidad **diaria**, es una reunión en la que los miembros del equipo se reúnen para evaluar el trabajo en curso y acciones a tomar. Esta cadencia se produce para cada uno de los servicios dentro de un sistema. Preguntas como: ¿Qué impedimentos tenemos? ¿Cómo es el flujo? ¿Qué podemos mejorar? son las que se hacen en esta reunión.

- **Service Delivery Meeting**

Esta reunión, cuya cadencia **depende de cuando se entregue software al cliente**, está orientada a examinar y mejorar la efectividad del servicio.



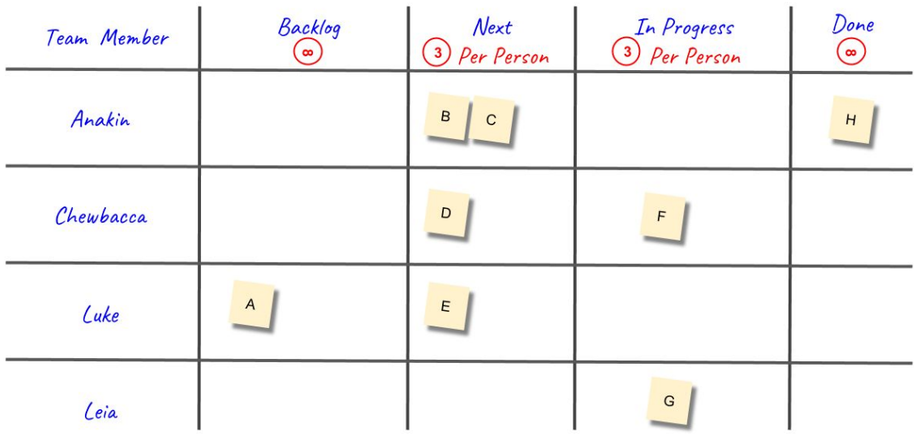
Proto Kanban

Un **Proto Kanban** es un precursor de un tablero Kanban donde existe un tablero visual pero aún está emergiendo un **sistema pull** *(sistema que limita el trabajo en progreso con señales visuales)* y el **flujo de trabajo para la entrega**. Podemos identificar varios tipos de Proto Kanban.

TIPOS

Aggregated Personal Kanban (en la oficina)

Un único tablero con filas por cada kanban personal de cada miembro del equipo.

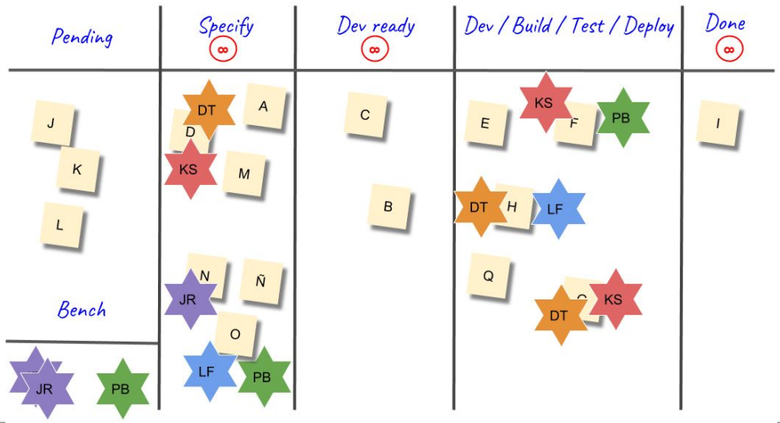


Team Kanban

Tablero kanban personal para un pequeño grupo de personas (de 3 a 4) quienes colaboran juntos en las mismas tareas.

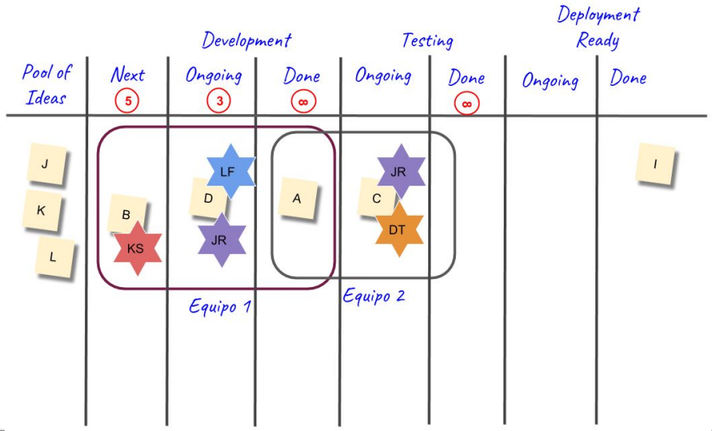
Per Person WIP Limit

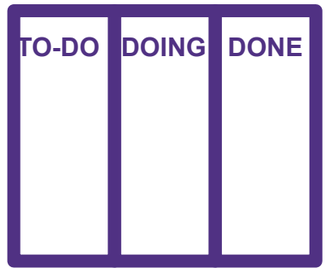
Visualizan parte del flujo de trabajo pero aún no es un sistema pull.
A menudo se implementa con avatares.



Aggregated Team Kanban

Encadena tableros To-Do, Doing y Done.
Muestra el flujo de trabajo pero aún no es un sistema pull de extremo a extremo.





WIP LIMIT

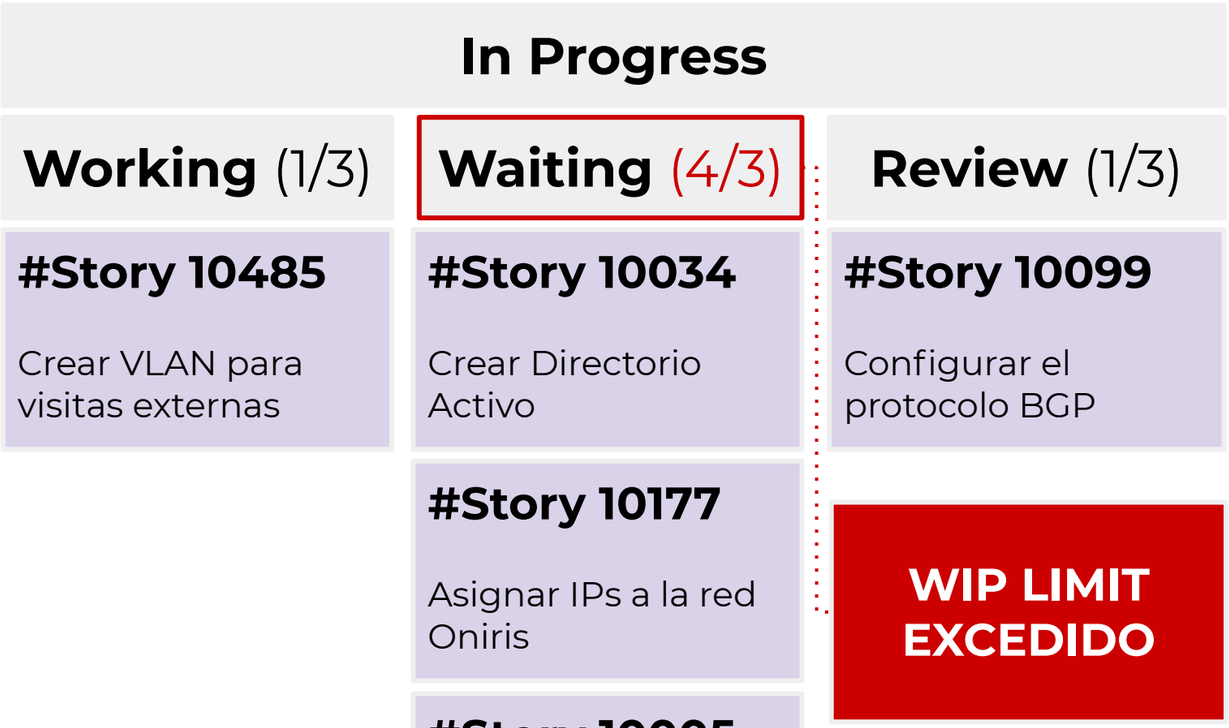
Las **métricas** en Kanban son una serie de medidas que nos van a permitir **conocer mejor nuestro progreso** entregando valor al cliente, osea, **entregando producto o parte de él terminado**. Dichas métricas van a darnos información sobre las desviaciones con respecto al objetivo principal.

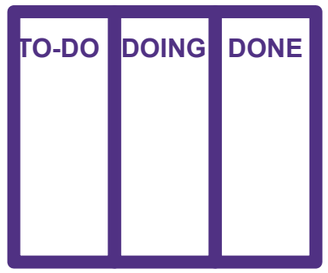
WIP LIMIT

Una de las principales métricas y también práctica del Método Kanban es el **WIP** que mide el trabajo en curso (Work In Progress) en cualquier momento. El **WIP Limit** en el *Diagrama de Flujo Acumulado (CFD),* abarca el trabajo pendiente **"Por hacer"** hasta el trabajo que todavía no cumple el *Definition of Done,* en el CFD fase **"Hecho"**.

Debido a que en entornos ágiles son de gran importancia las *entregas frecuentes*, en pos de obtener un rápido feedback del cliente que nos permita reajustar nuestros objetivos, el **WIP** debe definir cuántas tareas como máximo pueden realizarse en cada fase del ciclo de trabajo.

El WIP Limit favorece el método **GTD (Get Things Done)** debido a que impide iniciar tareas hasta que no se hayan finalizado aquellas en las que el equipo está trabajando.





Modelo de Madurez Kanban (KMM)

El propósito del **Modelo de Madurez Kanban** es apoyar el desarrollo de las siguientes capacidades organizacionales: alivio de la sobrecarga, cumplimiento con las expectativas del cliente, agilidad organizacional, supervivencia, resultados económicos predecibles y robustez financiera.

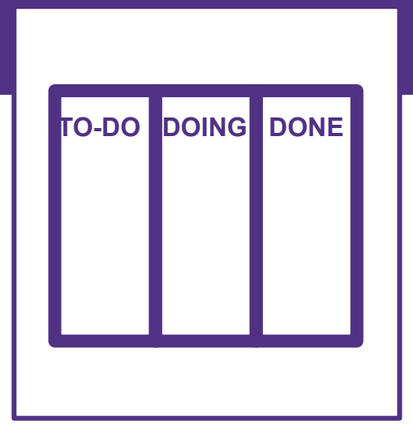
¿POR QUÉ EVALUAR EL NIVEL DE MADUREZ EN EQUIPOS KANBAN?

Porque tanto los equipos como los gerentes y coaches pueden evaluar rápidamente la capacidad del equipo y en función de ella pueden:

- *Ver cómo según la **capacidad del equipo** ésta se compara con el nivel recomendado por la agilidad.*
- *Guiar al equipo sobre **qué aspectos deberían ser los siguientes a mejorar.***
- *Ver, a lo largo del tiempo, la **evolución de la agilidad** (tendencia positiva o negativa).*
- *Ver el **impacto del trabajo de mejora del equipo**.*
- *Ver el **impacto de los cambios organizativos y estructurales** que están fuera del control del equipo (reducción de costos, equipos distribuidos, etc.).*
- ***¡Motivar!** Consiguiendo logros y reconocimientos por alcanzar mayores niveles de agilidad.*

¿Qué podemos medir?

- La implantación de los **Ciclos de Feedback.**
- La **Mejora.**
- La **Gestión del Flujo.**
- La **Visualización.**
- La **Limitación del WIP.**
- Las **Políticas Explícitas.**
- Los **Efectos.**



Modelo de Madurez Kanban (KMM)

Según la *Lean Kanban University* el modelo de madurez de Kanban presenta **7 niveles**, numerados del 0 al 6. Para que las empresas tecnológicas que ya están familiarizadas con el CMMI tengan facilidad de adopción, los niveles del 1 al 5 están alineados con los 5 niveles del CMMI, con una ligera diferencia en el nombre.

NIVELES DE MADUREZ

NIVEL 0 - INCONSCIENTE

La empresa es inconsciente a la necesidad del proceso. Los individuos completan las tareas con poca colaboración del equipo. Los tableros kanban personales básicos evolucionan hacia la concienciación de que hay demasiado trabajo en progreso.

NIVEL 1 - ENFOCADA EN EQUIPO

La definición del proceso está emergiendo pero puede aplicarse de forma inconsistente. Hay un concepto de equipos y colaboración pero mala alineación. Las heroicidades individuales (apaga fuegos) son comunes. Puede haber un tablero kanban básico o un tablero con tareas separadas por miembro del equipo.

NIVEL 2 - ORIENTADA AL CLIENTE

Los procesos básicos, el flujo de trabajo y las políticas se establecen y se siguen de manera coherente, pero los resultados deseados son inconsistentes. La calidad y la entrega del servicio son desiguales. Las mejoras tales como los límites de WIP por persona son interiorizados. El comportamiento es reactivo, no proactivo.

Niveles de Madurez

0 Inconsciente

- Kanban personal.
- Impredecible.

1 Enfocada en equipo

- Resultados inconsistentes.
- Personas felices.
- Proceso emergente.
- Kanban equipo.

2 Orientada al cliente

- Proceso consistente.
- Rutina.
- Resultados inconsistentes.
- Kanban de principio a fin.
- Kanban de aguas arriba.
- Gestión del flujo completo.

Modelo de Madurez Kanban (KMM)

NIVELES DE MADUREZ

NIVEL 3 - APTA PARA SU PROPÓSITO

Las definiciones y acuerdos de servicio son claros y definidos. Existe un Sistema Pull en el kanban. Los resultados deseados se logran de manera consistente. La entrega del servicio encaja según el propósito. Los clientes están satisfechos.

NIVEL 4 - PROTEGIÉNDONOS DE RIESGOS

Uso de métricas y sofisticados ciclos de retroalimentación. Clases de servicios gestionadas y recursos compartidos. La capacidad se asigna deliberadamente y se ajusta intencionalmente. Resultados económicos consistentes. Los gerentes están satisfechos.

NIVEL 5 - LÍDER DEL MERCADO

Optimizado para la eficiencia y mejora de los resultados económicos. Emerge una cultura de mejora continua. Uso del análisis cuantitativo, ciclos de feedback, experimentos y mejoras basadas en modelos.

Nivel 6 - CONSTRUIDA PARA LA SUPERVIVENCIA

Se alinea continuamente la estrategia con las capacidades operativas. Robustez frente a los cambios del mercado. Más allá de adaptable, capaz de reinventarse. "Construido para durar".

Niveles de Madurez

3 Apta para su propósito

- Resultados consistentes.
- Clientes satisfechos.
- Proceso consistente.
- Organización apta para su propósito.
- Organización expuesta a riesgos.
- Organización resiliente.

4 Protegiéndonos de riesgos

- Resultados predecibles.
- Accionistas satisfechos.
- Riesgos gestionados.
- Gestión basada en modelos predictivos.

5 Líder del mercado

- Mejora continua.
- Más apto para el propósito.
- Mejoras económicas.

6 Construida para la supervivencia

- Construida para perdurar.
- Sustentabilidad del negocio.
- Alineación consistente.
- Más apto para el propósito continuo.

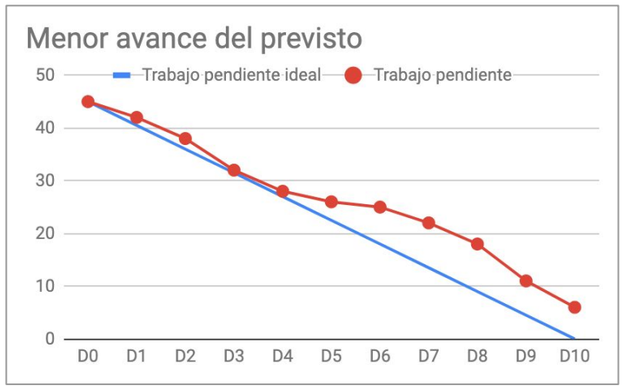
Burn-down chart

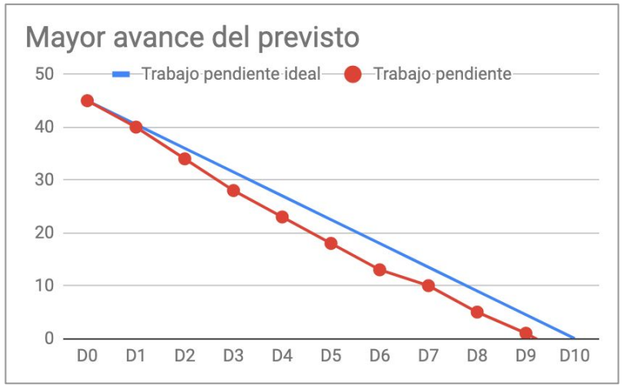
El objetivo de este gráfico facilitar el seguimiento del **trabajo** (esfuerzo) **que queda por realizar** en un periodo de tiempo concreto (iteración). En los equipos de desarrollo ágiles, el gráfico Burn-down se suele **aplicar por sprint (o release)** y ayuda al equipo a la **visualización** del **esfuerzo realizado** y el que queda por realizar hasta la finalización del sprint (o release).

UTILIZACIÓN

- En el eje **X se representa el tiempo**, bien en semanas, en días, etc, dependiendo de la duración de la iteración.
- En el eje **Y se representa el esfuerzo** a realizar, siendo lo más extendido el uso de puntos de esfuerzo (también podrían utilizarse horas, días-hombre, etc.).
- Si el gráfico representa una iteración, entonces la Y debería ser los **puntos totales de esfuerzo** para esta iteración, y la X la **duración de la iteración**.
- Se han tomado como referencia para el ejemplo una iteración de 2 semanas (10 días) y una estimación de 45 puntos de esfuerzo a realizar por el equipo.

- Podemos trazar la **línea de esfuerzo ideal** uniendo dos puntos:
  1. Inicio de iteración, total de esfuerzo estimado (x = D0, y = 45 en el ejemplo).
  2. Final de iteración, esfuerzo a cero (x = D10, y = 0).
- El gráfico se construye añadiendo **un punto al final de cada día**, con las coordenadas x = día e y = puntos de **esfuerzo** estimados **para finalizar**. Tras esta acción, se debe unir este punto con el del día anterior trazando una recta.
- Si estamos **por debajo de la línea de esfuerzo ideal**, el avance es **mayor del previsto** (queda menos por hacer), y si estamos **por encima**, el avance es **menor**.



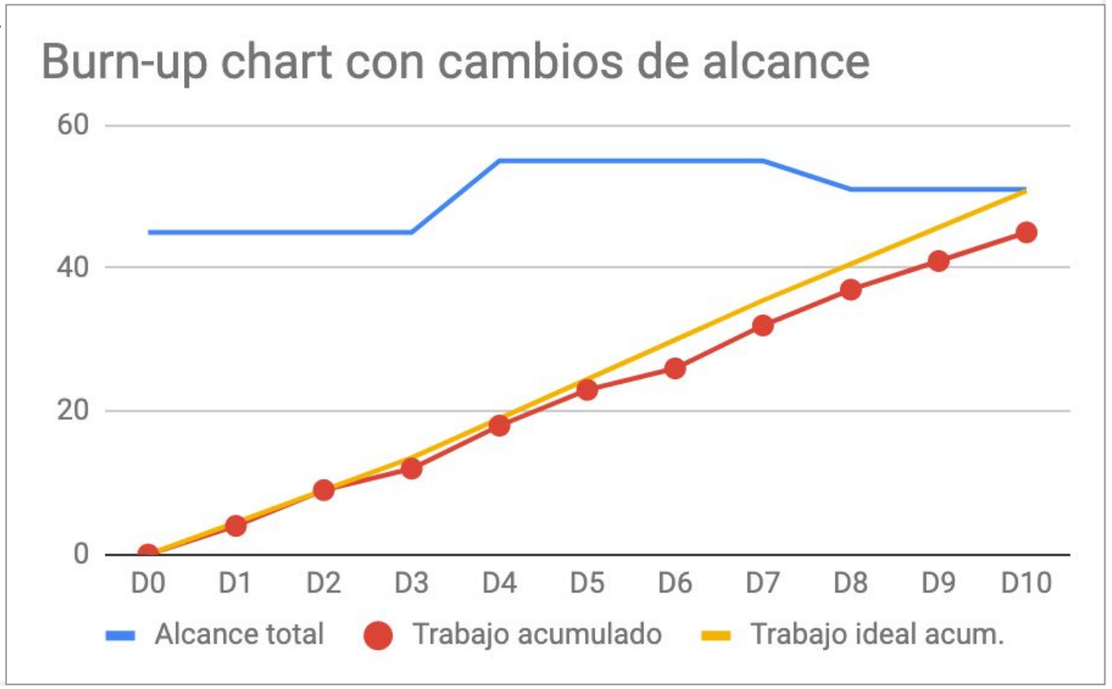


Burn-up chart

El objetivo de este gráfico es facilitar el seguimiento del **trabajo** (esfuerzo) **realizado** en un periodo de tiempo concreto (iteración). Para ello, muestra por separado el progreso del trabajo realizado y el alcance (trabajo total a realizar). Con ello, permite ver claramente los **cambios de alcance** en la iteración. Al igual que el Burn-down chart, se suele utilizar a nivel de iteración o release.

UTILIZACIÓN

- En el eje **X se representa el tiempo**, bien en semanas, en días, etc, dependiendo de la duración de la iteración.
- En el eje **Y se representa el esfuerzo** a realizar, siendo lo más extendido el uso de puntos de historia (también podrían utilizarse horas, días-hombre, etc.).
- Si el gráfico representa una iteración, se dibuja una línea con **puntos totales de esfuerzo** para esta iteración. Los incrementos y reducciones de alcance se reflejan en la misma.
- Se han tomado como referencia para el ejemplo una iteración de 2 semanas (10 días) y una estimación de 45 puntos de esfuerzo a realizar por el equipo. El alcance el D4 sube a 55 puntos y el D8 baja a 51.



- Podemos trazar la **línea de esfuerzo ideal** uniendo dos puntos:
  1. Inicio de iteración, esfuerzo a cero (x = D0, y = 0 en el ejemplo).
  2. Final de iteración, total de esfuerzo estimado (x = D10, y = 51).
- El gráfico se construye añadiendo **un punto al final de cada día**, con las coordenadas x = día e y = puntos de **esfuerzo** estimados **para finalizar**. Tras esta acción, se debe unir este punto con el del día anterior trazando una recta.
- Si estamos **por encima de la línea de esfuerzo ideal**, el avance es **mayor del previsto**, y si estamos **por debajo**, el avance es **menor**.

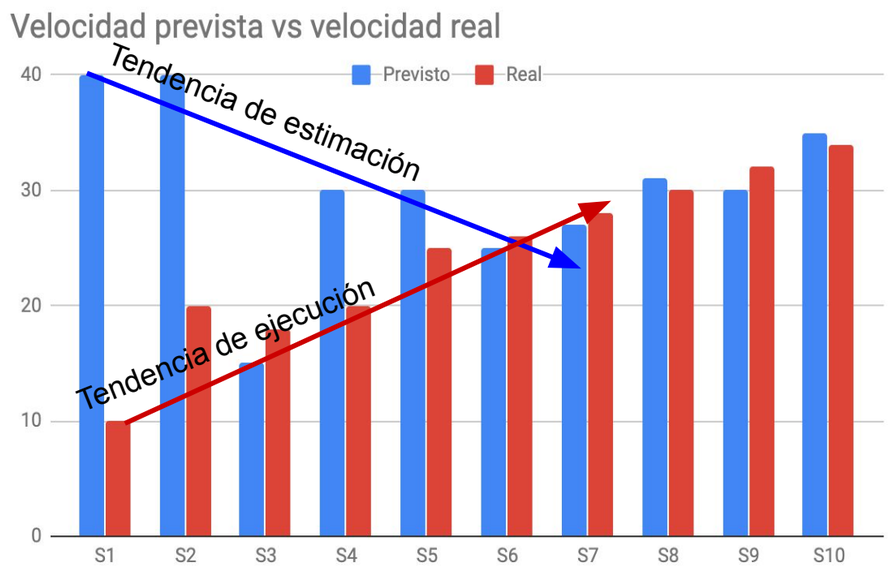
Velocity chart

El objetivo de este gráfico es poder visualizar la **evolución de la velocidad del equipo** a lo largo de múltiples iteraciones. Para ello, muestra (para cada iteración) los puntos de esfuerzo estimados del equipo, así como los realmente alcanzados al finalizar cada iteración.

UTILIZACIÓN

¡Atención! La velocidad es una métrica que no se puede utilizar para comparar el rendimiento de equipos diferentes. Se debe limitar a la evolución del propio equipo.

- En el eje **X se representan las iteraciones**.
- En el eje **Y se representa el esfuerzo**, siendo lo más extendido el uso de puntos de esfuerzo (también podrían utilizarse horas, días-hombre, etc.).
- Se ha tomado para el ejemplo un equipo que lleva realizados 10 sprints o iteraciones. El esfuerzo lo reflejan en puntos de esfuerzo.



- Para crear el diagrama son necesarios dos datos **por cada iteración**:
  1. **Estimación de esfuerzo total** a realizar por el equipo, obtenido al planificar el trabajo al comienzo de la iteración (en la Sprint Planning para un equipo Scrum).
  2. **Esfuerzo total del trabajo realizado** durante la iteración, obtenido al final de la misma (en la Sprint Review para un equipo Scrum).
- El gráfico puede reflejar grandes **divergencias entre estimado y realizado**, siendo más acentuadas en las primeras iteraciones. Esto sucede especialmente en equipos nuevos con inmadurez tecnológica o metodológica.
- Según avanzan las iteraciones, las divergencias deberían reducirse.
- Un **equipo maduro** en **mejora contínua** se quedará unas veces por encima y otras por debajo de la estimación, pero siempre cerca de la misma. Además **mostrará una tendencia creciente** hasta llegar a su **umbral de eficiencia**. Así sabremos que ni ha caído en la zona de confort ni está saturado de trabajo.

Throughput chart

El **throughput** indica la cantidad de **elementos de trabajo realizados** (no puntos de esfuerzo) en un periodo de tiempo concreto. Este gráfico muestra su **evolución** con la cadencia que nos interese (cada día, semana, mes, ...). Esto permite a un equipo comprometido con la mejora continua visualizar sus avances en términos de productividad y predecir las fechas de finalización de sus elementos de trabajo.

UTILIZACIÓN

- En el eje **X se representa el tiempo**, bien en semanas, en días, etc., dependiendo de la cadencia que nos interese.
- En el eje **Y se representan los elementos de trabajo realizados**.
- Se han tomado como referencia para el ejemplo un equipo que contabiliza el trabajo entregado cada 15 días.

- Para crear este gráfico, se deben hacer dos cosas:
  1. **Determinar la cadencia** que queremos usar.
  2. **Contabilizar** los elementos de trabajo **finalizados** en dicho periodo temporal.
- Se puede hacer una **previsión de fechas de finalización** de un determinado trabajo. Para ello se debe:
  1. Dividir el trabajo en n elementos de trabajo homogéneos.
  2. Determinar el throughput medio, una vez el equipo sea estable.
  3. Calcular el tiempo que llevará completar el trabajo aplicando la **Ley de Little** (tiempo = nº elementos de trabajo / throughput).
  4. Sumar el tiempo obtenido a la fecha de inicio del trabajo.

- Para que el throughput sea una métrica consistente y precisa, es necesario que el **tamaño medio** de los elementos de trabajo sea **homogéneo**.
- Aunque el throughput debería mostrar la mejora del equipo, tiene un **umbral de eficiencia**. En el ejemplo gráfico, el equipo se estabiliza con un throughput de 120 elementos de trabajo realizados cada 15 días.

Cumulative Flow Diagram (CFD)

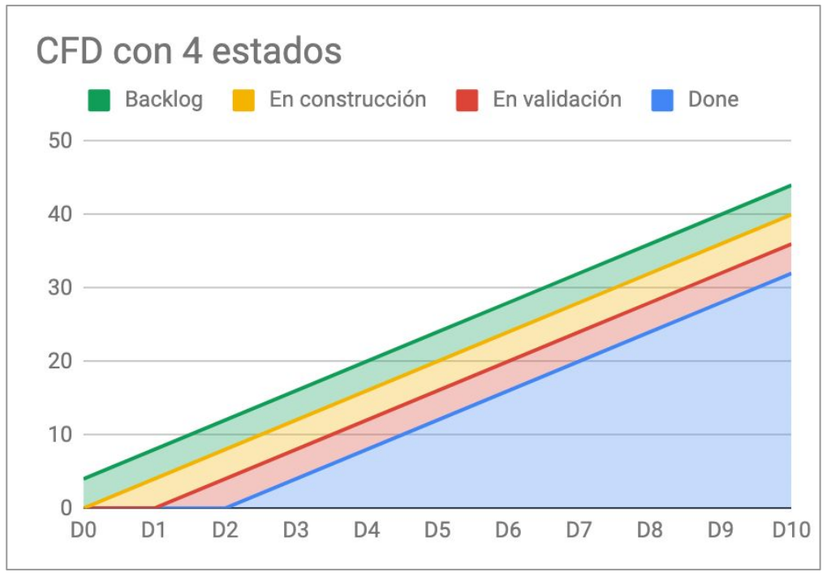
El diagrama de flujo acumulado o **CFD** muestra la **evolución en el tiempo del flujo de trabajo** de un equipo. Para ello, refleja la **cantidad de trabajo en curso (WIP) en cada estado**, acumulando o apilando estas cantidades gráficamente, en periodos regulares.

UTILIZACIÓN

- En el eje **X se representa el tiempo**, bien en semanas, en días, etc., siendo lo habitual hacerlo en días.
- En el eje **Y se representa el número de elementos de trabajo en cada estado** del flujo de trabajo del equipo. Estos se apilan en orden inverso respecto al flujo. Alternativamente, se podrían reflejar puntos de esfuerzo.

- Para construir este gráfico es necesario disponer de la cantidad de elementos de trabajo que hay en cada estado cada día:
  1. Se comienza por el último estado del flujo (Done) y se marca el número de elementos.
  2. Se toma el anterior estado del flujo (En validación) y se marca el número de elementos apilándolo.
  3. Se continua realizando el paso 2 (En construcción) hasta llegar al estado inicial del flujo (Backlog).

  Finalmente, se unen los puntos de cada estado y se colorean las distintas áreas resultantes.
- Cuando las bandas del CFD varían suavemente y su altura es estable, es síntoma de un buen equilibrio entre demanda y capacidad.
- Aunque este gráfico es ampliamente utilizado con **Kanban**, también es una herramienta potente para otros frameworks ágiles.



Métricas ágiles - Cumulative Flow Diagram

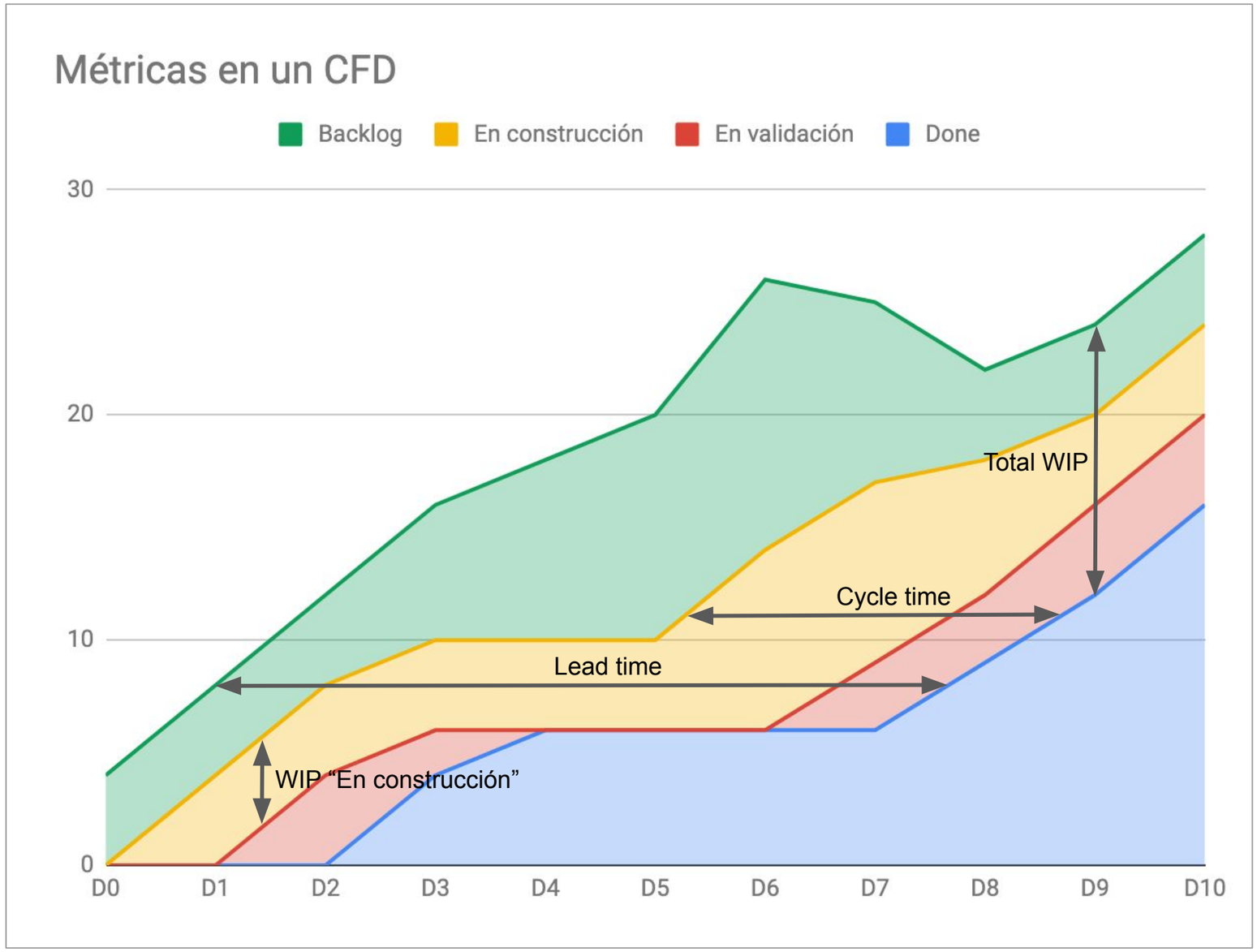
autentia

Métricas en un CFD

Un CDF muestra una **gran cantidad de información** de manera muy compacta: el **burn-up** chart, **lead time**, **cycle time**, el trabajo en curso (**WIP**) e incluso **cuellos de botella**.

Definición de las métricas:

- **Cycle time**: es el tiempo que pasa desde que se inicia un trabajo hasta que se entrega. Para el ejemplo: desde que entra en el estado "En construcción" hasta que llega a "Done".
- **Lead time**: es el tiempo que pasa desde que se solicita un trabajo (entra en el Backlog) hasta que se entrega (Done).
- **WIP (Work in Progress)**: es la cantidad de elementos en progreso para uno o varios estados del flujo de trabajo.
- **Burn-up**: la evolución del último estado del flujo (Done) muestra cómo se va completando el trabajo, al igual que en un burn-up chart.



¿Por qué el Total WIP incluye el Backlog?

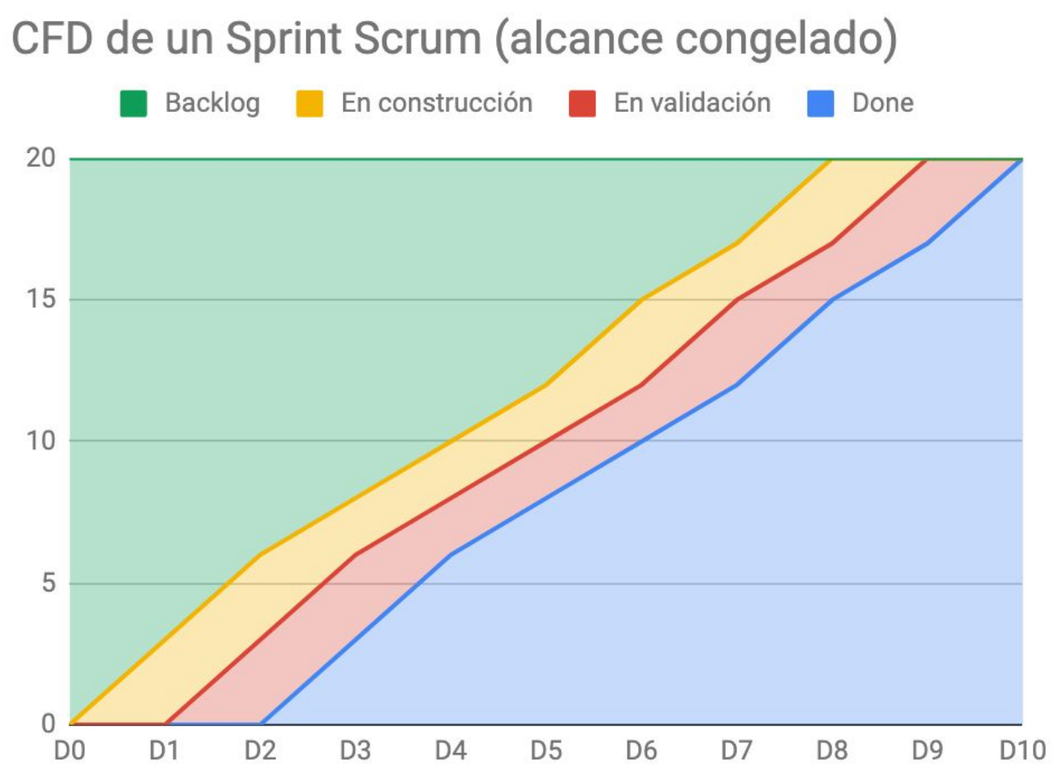
Para entenderlo hay que ver el sistema **desde fuera**, como una caja negra. Entran elementos (Backlog), se procesan y se entregan (Done). Todo lo que no se ha entregado está en proceso (Total WIP).

Entendiendo los "cortes" en un CFD

- Los **cortes verticales** muestran la **cantidad de elementos de trabajo** de cada estado para un momento dado.
- Los **cortes horizontales** muestran los **tiempos medios** que los elementos de trabajo pasan en cada estado.

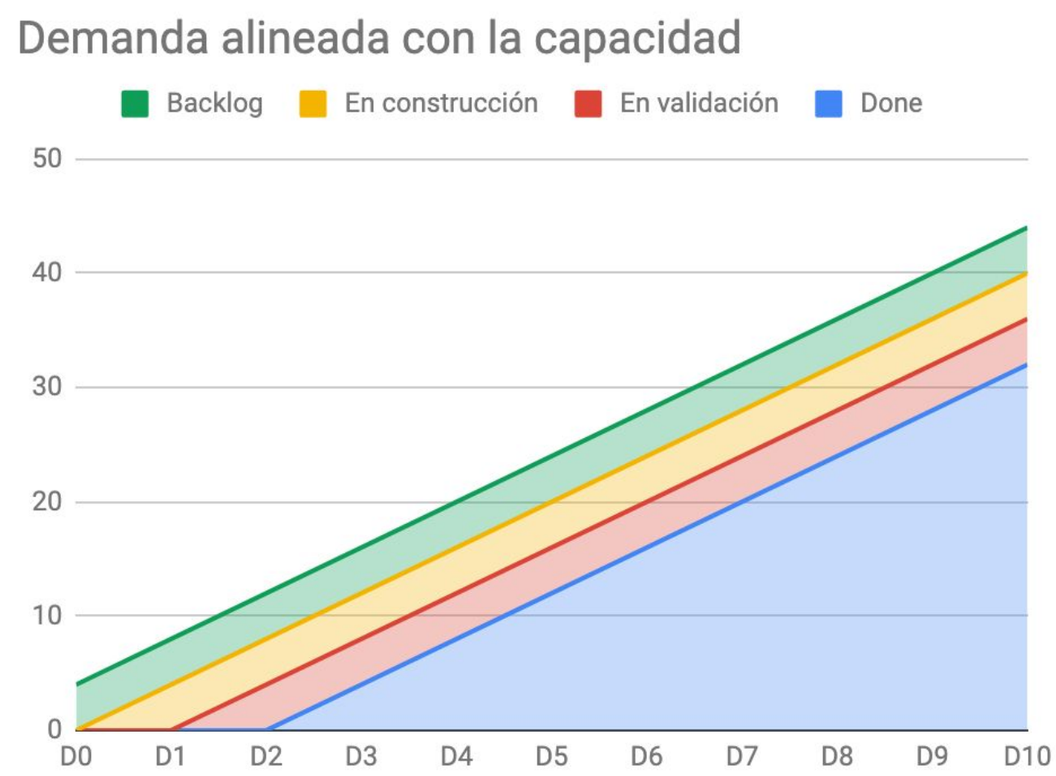
EJEMPLO - Sprint Scrum

- Un CFD puede reflejar perfectamente la evolución de los elementos de trabajo en Scrum.
- En el ejemplo se refleja un Sprint, lo cual tiene algunas particularidades:
  - El alcance, se limita al Sprint Backlog, confeccionado en la sesión de Sprint Planning.
  - El eje X (tiempo) se limita a la duración del Sprint (10 días).
  - Al partir de un alcance dado (20), el Backlog WIP comienza en su máximo y se reduce conforme pasa el tiempo.



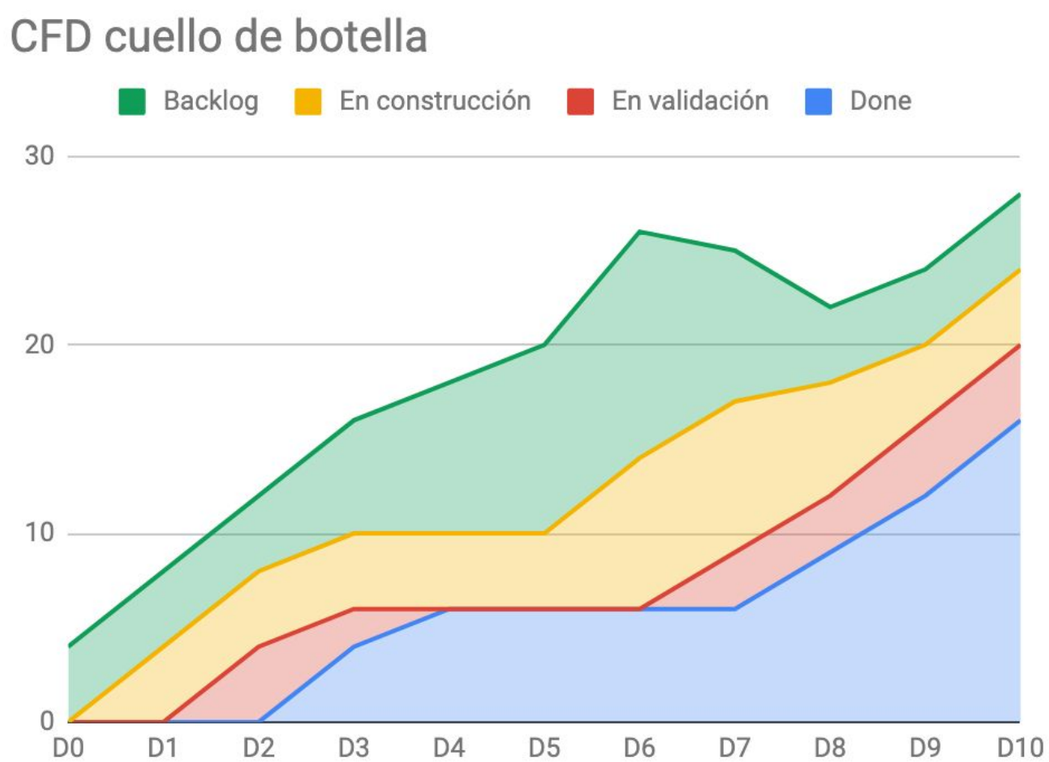
EJEMPLO - Flujo continuo equilibrado

- Un CFD habitualmente muestra un flujo continuo.
- La demanda se refleja con el primer estado del flujo de trabajo (en el ejemplo, el Backog).
- La capacidad global se refleja con el último estado (Done).
- Cuando las pendientes de las distintas rectas son iguales (son paralelas), significa que demanda y capacidad están alineadas.
- En el ejemplo se ve una situación ideal.



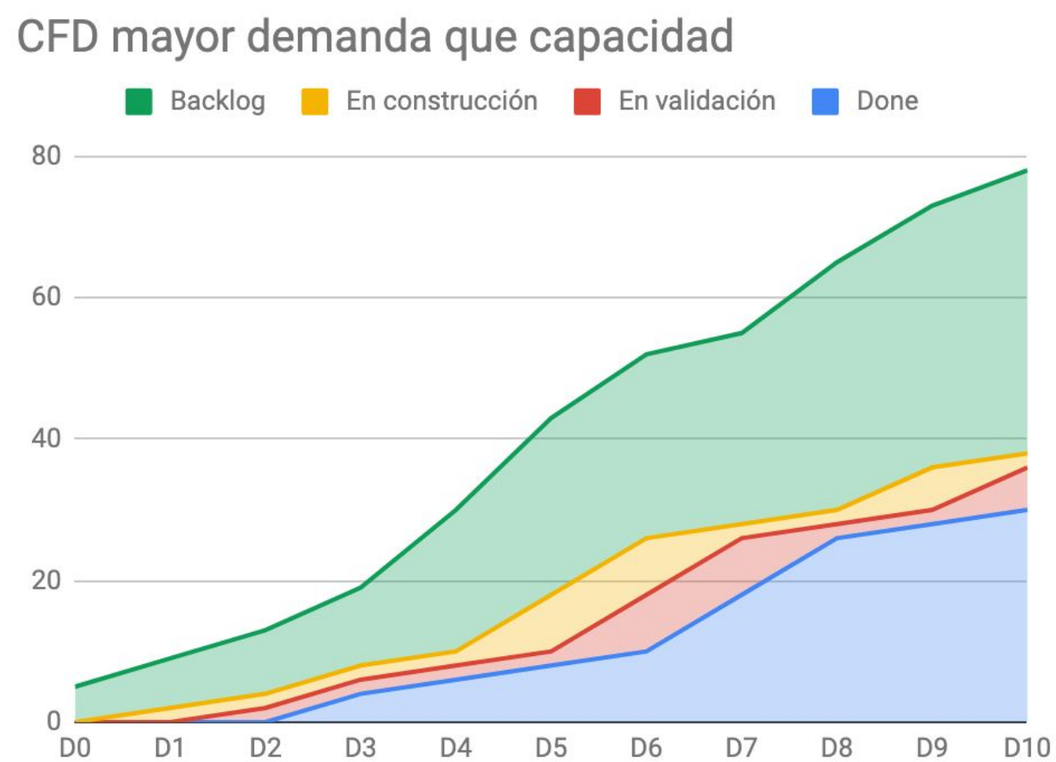
EJEMPLO - Cuello de botella

- Hasta el día 2 todo avanza equilibradamente.
- A partir de ahí algo se atasca en el estado “En construcción”. El tamaño del backlog comienza a crecer y en el estado “En validación” se van quedando sin trabajo.
- La situación se reconduce poniendo el foco en el estado donde hay problemas (se aumenta su capacidad), hasta que vuelven a equilibrarse capacidad y demanda.



EJEMPLO - Equipo saturado

- Este CDF muestra como la demanda (representada en el backlog) crece a ritmo superior a la capacidad disponible.
- Se puede apreciar que el backlog va creciendo de tamaño, a pesar de que el equipo de trabajo tiene varios puntos de sobreesfuerzo para tratar de controlar la demanda.
- Si esta situación se mantiene, el equipo se saturará y probablemente baje la calidad de las entregas.



¿Qué es un Sesgo Cognitivo?

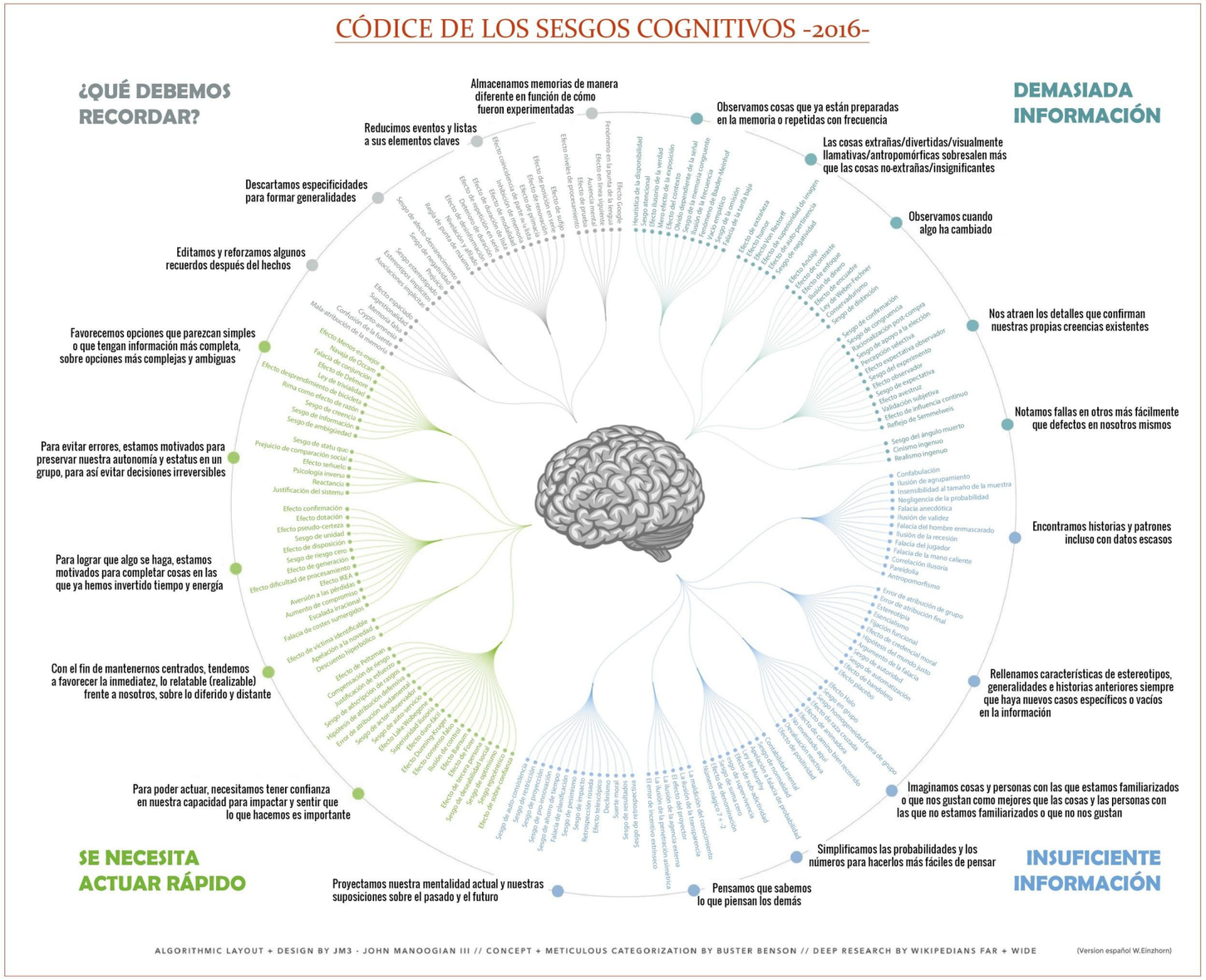
Un **sesgo cognitivo** es el *efecto psicológico* que produce una desviación en el procesamiento mental, lo que lleva a una distorsión, juicio inexacto o interpretación ilógica, que **se da sobre la base de la interpretación de la información disponible**, aunque los datos no sean lógicos o no estén relacionados entre sí.

Fuente: https://es.wikipedia.org/wiki/Sesgo_cognitivo

EL CÓDICE DE LOS SESGOS COGNITIVOS

Actualmente **existen identificados aproximadamente 188 sesgos cognitivos**. Nosotros nos basamos en la agrupación realizada por ***Buster Benson*** quien, agrupando sesgos similares, **obtuvo una lista de 20 tendencias estratégicas exclusivas** que usamos por razones muy específicas.

Además, hace una **agrupación** adicional en la que se analizan los sesgos **por el problema que están tratando de resolver**, lo que hace mucho más fácil de entender por qué existen, por qué son útiles y qué compensaciones introducen.



Sesgos Cognitivos - Agrupaciones por problema que tratan de resolver

autentia

DEMASIADA INFORMACIÓN

- Observamos cosas que ya están preparadas en la memoria o repetidas con frecuencia.
- Las cosas extrañas/divertidas/visualmente llamativas/antropomórficas sobresalen más que las cosas no extrañas/ insignificantes.
- Observamos algo cuando ha cambiado.
- Nos atraen los detalles que confirman nuestras propias creencias existentes.
- Notamos fallas en otros, más fácilmente que defectos en nosotros mismos.

INSUFICIENTE INFORMACIÓN

- Encontramos historias y patrones incluso en datos escasos.
- Rellenamos características de estereotipos, generalidades e historias anteriores siempre que haya nuevos casos específicos o vacíos en la información.
- Imaginamos cosas y personas con las que estamos familiarizados o que nos gustan como mejores que las cosas y las personas con las que no estamos familiarizados o que no nos gustan.
- Simplificamos las probabilidades y los números para hacerlos más fáciles de pensar.
- Pensamos que sabemos lo que piensan los demás.
- Proyectamos nuestra mentalidad actual y nuestras suposiciones sobre el pasado y el futuro.

NECESIDAD DE ACTUAR RÁPIDO

- Para poder actuar, necesitamos tener confianza en nuestra capacidad para hacer un impacto y sentir que lo que hacemos es importante.
- Con el fin de mantenernos enfocados, tendemos a favorecer la inmediatez, lo realizable frente a lo diferido y distante.
- Para lograr que algo se haga, estamos motivados para completar cosas en las que ya hemos invertido tiempo y energía.
- Para evitar errores y decisiones irreversibles, estamos motivados para preservar nuestra autonomía y estatus en un grupo.
- Preferimos opciones que parezcan simples o que tengan más información completa, sobre opciones más complejas.

¿QUÉ DEBEMOS RECORDAR?

- Editar y reforzar algunos recuerdos después del hecho.
- Descartamos especificidades para formar generalidades.
- Reducimos eventos y listas a sus elementos clave.
- Almacenamos recuerdos de manera diferente en función de cómo fueron experimentados.

Sesgos Cognitivos - Selección de Sesgos I

autentia

SESGO DE CONFIRMACIÓN

El sesgo de confirmación o sesgo confirmatorio es la **tendencia a favorecer, buscar, interpretar, y recordar, la información** que confirma las propias creencias o hipótesis, dando desproporcionadamente menos consideración a posibles alternativas.

Ejemplo: *Si tenemos la creencia que al bañarnos inmediatamente después de comer podemos sufrir un corte de digestión. Al buscar información, no buscamos refutar nuestra creencia con búsquedas como: "Bañarse después de comer produce un corte de digestión" sino que haremos una búsqueda tal como: "¿Por qué bañarse después de comer puede producir un corte de digestión".*

SESGO DE PROYECCIÓN

Es la **tendencia inconsciente a asumir** que los demás comparten los mismos pensamientos, creencias, valores o posturas parecidas a las nuestras.

Ejemplo: *En un proceso de selección el personal, el headhunter tiende a favorecer la contratación del candidato que comparta sus mismos valores familiares o hobbies. No significando esto que sea el mejor candidato.*

EFECTO DE CONSENSO FALSO

Es un sesgo cognitivo por el que **muchas personas tienden a sobreestimar el grado de acuerdo que los demás tienen con ellos.** Las personas tienden a presuponer que sus propias opiniones, creencias, predilecciones, valores y hábitos están entre los más elegidos y apoyados. Esta creencia **es un sesgo que exagera la confianza de los individuos en sus propias creencias,** aun cuando éstas sean erróneas o minoritarias.

Ejemplo: *Los bebedores creen que el alcohol le gusta más a la gente que lo que creen los abstemios.*

DESCUENTO HIPERBÓLICO

Dadas dos recompensas similares, las personas muestran mayor preferencia por aquella que llegue más pronto. La tendencia se incrementará conforme ambos beneficios estén más cerca al presente. En este sentido, se descuenta el valor del beneficio posterior por un factor que aumenta con la duración del tiempo de espera.

Ejemplo: *En EEUU al ganar la lotería te dan dos opciones, recibir el 60% del premio al momento o recibir el premio al 100% distribuido en un 20% al momento y el resto fraccionado durante 20 años. Casi todo el mundo elige la 1ª opción a pesar de perder el 40% del premio ganado.*

Sesgos Cognitivos - Selección de Sesgos II

autentia

EFECTO DE SOBREATRIBUCIÓN

Es la **tendencia a sobre-dimensionar los motivos personales internos a la hora de explicar un comportamiento observado en otras personas**, dando poco peso por el contrario a motivos externos como el rol o la situación, para este mismo comportamiento.

Cuando las personas piensan sobre sí mismas atribuyen su éxito a una cualidad personal interna (inteligencia, bondad, fuerza...) y sus fracasos a circunstancias externas (mala suerte, desventaja, manías...). Cuando la gente piensa en otros, atribuyen los aciertos de los demás a circunstancias externas (tuvo suerte, tuvo ventaja) y los fallos a debilidades o características internas (tiene mucho carácter, él no es suficientemente fuerte, inteligente).

MALDICIÓN DEL CONOCIMIENTO

Se produce cuando un individuo, en comunicación con otras personas, sin saberlo, **supone que los otros tienen los antecedentes necesarios para entender lo que dice.**

Ejemplo: *Cuando hablas utilizando tu jerga laboral, el receptor puede no estar familiarizado con esas palabras. Habla a tu madre en términos de métodos, arrays o merges, probablemente, no te entienda.*

EFECTO DE ENCUADRE

Es la **tendencia a alterar las decisiones según sea presentada una misma opción**. En concreto, las personas tienden a seleccionar opciones, dependiendo de si la cuestión está enmarcada para concentrarse en las pérdidas o en las ganancias.

Ejemplo: *Un virus se ha expandido entre una pequeña población, como médico tienes 2 opciones. A.- Salvar a 200 personas con las vacunas disponibles o bien B.- Usar un medicamento experimental que asegura ⅓ de la supervivencia de un grupo de 600 personas. La mayoría eligió A, la perspectiva de salvar seguro 200 vidas es más atractiva que la B con el mismo valor esperado.*

SESGO DE RETROSPECTIVA

Es la **tendencia a recordar las decisiones propias como mejores de lo que realmente fueron**.

Ocurre también cuando los individuos, una vez que conocen el resultado final, modifican su recuerdo al pensar que siempre supieron lo que iba a ocurrir. Esto se observa fácilmente cuando se registran las predicciones de los individuos, y se da cuenta de que objetivamente nadie había acertado el resultado antes de conocerlo.

Ejemplo: *Este sesgo está muy presente especialmente en historiadores cuando describen el resultado de una batalla y en el sistema jurídico cuando se imputa una responsabilidad.*

Sesgos Cognitivos - Selección de Sesgos III

autentia

EFECTO DE VÍCTIMA IDENTIFICABLE

Se refiere a la **tendencia de los individuos a ofrecer una mayor ayuda cuando se observa una persona en dificultades, en comparación con un grupo grande,** vagamente definido y con la misma necesidad. El efecto también se observa cuando los sujetos administran castigo en lugar de recompensa.

Ejemplo: *Son varios los casos en el en TV aparecen casos de personas con enfermedades raras y en lo que la gente se vuelca ayudando económicamente, en comparación de ayudar a una fundación en concreto que busque luchar contra la enfermedad.*

FALACIA DE LOS COSTES SUMERGIDOS

Es una falacia que **se produce cuando alguien realiza una inversión que no parece rentable y razona de la siguiente manera**: *«No puedo parar ahora, de otra manera lo que he invertido hasta el momento se perderá».*

Los argumentos para seguir invirtiendo en el proyecto no se deben basar en el miedo a la pérdida de lo invertido sino en las expectativas de funcionamiento del proyecto, ambas cosas totalmente independientes.

Ejemplo: *Este argumento es el que da cualquier jugador de casas de apuestas y que normalmente termina perdiéndolo todo.*

AVERSIÓN A LA PÉRDIDA

Se refiere a la **fuerte tendencia de la gente a preferir evitar pérdidas monetarias antes que conseguir ganancias monetarias equivalentes**: las pérdidas pesan mucho más que las ganancias.

Ejemplo: Considere la siguiente apuesta con dos posibles resultados: 50% de ganar 1000€ o 50% de perder 800€. ¿Aceptaría jugar?

La inmensa mayoría elige no jugar.

SESGO DE RESULTADOS

Tendencia a juzgar una decisión por su resultado final, en lugar de juzgarla por la calidad o acierto de la decisión, cuando fue realizada.

Ejemplo: *Una inversión muy arriesgada y que puede hacerte ganar mucho dinero se juzgará de diferente manera si ha sido acertada o no.*

Sesgos Cognitivos - Selección de Sesgos IV

autentia

SESGO POR EXCESO DE CONFIANZA

Tendemos a sobrevalorar nuestra capacidad intuitiva para razonar y hacer predicciones. Por tanto, pensamos que somos más listos que los demás o que respondemos de forma más eficaz.

Ejemplo: *En una encuesta sobre conducción se preguntaba, ¿cómo de buen conductor se considera Ud. en relación a la media española? Un 39% se consideraba mejor que la media. Si bien en algunos casos era correcto, en otros no y aún así se consideraban por encima de la media.*

COMPENSACIÓN DE RIESGO

Tendencia a hacer elecciones adversas y de riesgo si los resultados esperados son positivos, porque realizando elecciones de riesgo se tiene la preconcepción de evitar resultados negativos o no tan favorables.

Ejemplo: *Se han realizado experimentos en los que se preguntaba a los sujetos si estarían dispuestos a llevar a cabo una acción arriesgada determinada para ganar 10€, y esa misma acción para evitar perderlos.* ***La gente estaba más predispuesta a llevar a cabo la acción arriesgada cuando tenían que pagarlos, en lugar de recibirlos.***

EFECTO HALO

Es la **capacidad de un individuo para modificar la percepción o evaluación que los demás tienen de las demás cualidades personales a través de una cualidad específica.** Un individuo es capaz de trasladar la percepción que tienen los demás de sus cualidades positivas o negativas desde un área de la personalidad a otra a través de una única capacidad que arrastra a las demás.

Ejemplo: *La creencia de que una persona por ser atractiva tiene también características positivas como inteligencia, sensibilidad, clase, etc.*

ILUSIÓN DE ENTENDIMIENTO ASIMÉTRICO

Es un **prejuicio que implica el hecho de que la gente percibe su conocimiento de los demás como superior y más preciso al conocimiento que tienen los demás sobre sí mismos.** De igual manera, las personas tienden a creer que se conocen mejor a sí mismas que lo que sus parejas o iguales se conocen a sí mismos o a ellos.

Ejemplo: *Con relativa frecuencia sentimos que nuestro comportamiento ha sido mal interpretado o nuestro carácter mal percibido.*

Sesgos Cognitivos - Selección de Sesgos V

autentia

SESGO DE AUTOSERVICIO O INTERÉS PERSONAL

Aparece cuando la gente se auto atribuye el crédito personal en sus éxitos pero no de sus fracasos. Se manifiesta también, cuando la gente tiene la tendencia a evaluar algunas situaciones de forma ambigua de tal forma que beneficie a sus propios intereses.

Ejemplo: *Los empresarios encargados de las empresas suelen culpar a sus subordinados cuando la empresa no tiene buenos resultados, mientras que los trabajadores de la empresa tienden a explicar sus razonamientos a las circunstancias.*

EFECTO ILUSORIO DE LA VERDAD

Establece que una persona tiende a considerar como ciertos con más facilidad aquellos enunciados que le resultan familiares.

Ejemplo: *Los medios de comunicación hoy en día están saturados de anuncios publicitarios que afirman diversos "hechos". Aunque estas afirmaciones pudieran ser desacreditadas por personas serias, estos descréditos podrían tener poco éxito.*

SESGO DE SUPERVIVENCIA

Es una **falacia lógica que se concentra en las personas o cosas que superaron un proceso de selección y pasando por alto a aquellas que no lo hicieron, típicamente por su falta de visibilidad.** Esto puede conllevar a conclusiones falsas en diferentes formas de sesgo de selección.

Ejemplo: *Facebook, Microsoft o Google son ejemplos de empresas que han tenido un éxito importante entre muchas. ¿Alguien se acuerda de las que han fracasado?*

SESGO DEL NADADOR OLÍMPICO

Tendencia a pensar que, **si entrenamos tan duro como un nadador olímpico, llegaremos a tener su cuerpo físico.** Cuando en realidad ese nadador olímpico nació con una genética óptima para el deporte, lo que le aporta esa ventaja sobre los demás.

Ejemplo: *La universidad de Harvard, ¿realmente es tan buena universidad que de ella salen mentes privilegiadas o su prestigio se debe a los filtros tan altos que imponen que sólo acceden las mentes más brillantes?*

Sesgos Cognitivos - Selección de Sesgos VI

autentia

SESGO DE NEGATIVIDAD

Es la **noción de que**, incluso cuando tienen la misma intensidad, **las cosas de naturaleza más negativa** (pensamientos desagradables, emociones o interacciones sociales, eventos traumáticos, etc.) **tienen mayor efecto sobre el estado psicológico y los procesos, que las cosas neutrales o positivas**.

Ejemplo: *Los telediarios se centran en dar noticias de aspecto negativo (guerras, subida de impuestos, datos negativos de economía) ante las positivas que se reservan al final (victorias deportivas, éxitos personales, etc.).*

EFECTO ANCLAJE

Es la **tendencia a confiar demasiado en la primera información ofrecida (el ancla) al tomar decisiones.** Durante la toma de decisiones, el anclaje ocurre cuando las personas usan una información inicial para emitir juicios posteriores.

Ejemplo: *En los centros comerciales es muy normal que te pongan un producto muy caro cerca del producto que realmente te quieren vender. De modo que el producto caro es tomado como ancla y frente al cual se toman las decisiones.*

HEURÍSTICA DE LA DISPONIBILIDAD

Es una **predicción sesgada, debido a la tendencia a centrarse en el beneficio o suceso más sobresaliente, más familiar y emocionalmente cargado**. Es decir, al juzgar grosso modo una probabilidad, ésta tiende a estar influenciada por la experimentación emocional del sujeto o a través del suceso más sobresaliente que haya sufrido este.

Ejemplo: *Si estamos buscando apartamento y vemos que algunos miembros de nuestro círculo de amistades no ha tenido éxito en esa tarea, pensaremos que la oferta inmobiliaria es escasa o difícil.*

SESGO DE PUNTO CIEGO

Es la **tendencia a no darse cuenta de los propios prejuicios cognitivos** o a **verse a sí mismo como menos sesgado que los demás.**

Ejemplo: *El cuento de los 6 sabios ciegos y el elefante es un claro ejemplo en el que todos tienen parte de razón y a la vez todos están equivocados.*

Value Stream Mapping - VSM

autentia

¿QUÉ ES UN VALUE STREAM MAPPING?

Es una **herramienta que sirve para ver y entender un proceso e identificar y reconocer las fuentes de desperdicios** para posteriormente una vez identificadas intentar eliminarlos o en su defecto reducirlos.

¿QUÉ SE MUESTRA?

Una foto, **de tu organización**, en la que se revela el valor que está fluyendo a través de ella y en qué áreas operacionales aún tenemos desperdicios. Basándonos en la **filosofía Kaizen** detectamos: la inconsistencia (**mura**), el gasto (**muda**) y la sobrecarga (**muri**).

Normalmente cuando se genera un VSM, realmente se crean dos. Un primero que representa el **AHORA**, tu situación actual y otro en el que se muestra el **FUTURO** o situación deseada después de eliminar los desperdicios.

¿CUÁL ES EL ENFOQUE?

El VSM está enfocado hacia los Managers, para explicarles exactamente **QUÉ** quieres hacer, **DÓNDE** se encuentran actualmente los desperdicios y **CUÁL** es tu plan para eliminarlos.

Según la experiencia, **los Managers reaccionan muy bien a esta herramienta visual** a diferencia que si la misma información se presenta en un documento de texto de N páginas. *Esto es así por el Sesgo de **"Demasiada Información".***

¿CUÁLES SON LOS DESPERDICIOS?

Desaprovechamiento del Talento
No utilizar la experiencia, conocimiento y creatividad del personal

Exceso de Inventario
Exceso de materia prima, productos y procesos sin uso

Movimientos innecesario
Realizados por el personal

Tiempos de Espera
Tiempo perdido mientras se espera por el próximo paso en el proceso

Transporte
Movimientos innecesario de productos y materia prima

Defectos
Información, productos o servicios incorrectos o incompletos

Sobreproducción
Producción no ajustada a la demanda, o bien no interesan a los consumidores

Exceso de Procedimientos
Procesos no optimizados que suponen más trabajo o calidad de la que se quiere

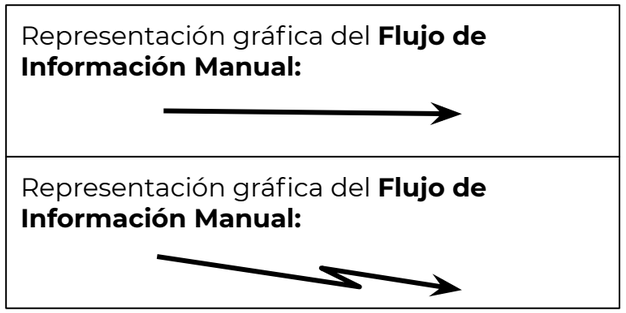
¿CÓMO DIBUJAR UN VSM?

La mejor forma de dibujar el VSM es darte un paseo por toda la organización, desde el origen del proceso, **preguntando a los managers o gestores hasta los empleados**. Basándonos en la información obtenida se construye el VSM. **En el VSM se identifican 3 atributos clave** que son:

1. INFORMATION FLOW

En este atributo clave se va a mapear el flujo de información desde los proveedores hasta los clientes pasando por todas las áreas de la organización. Identificamos dos flujos:

- **El Flujo de Información Manual**, puede tomar variedad de formas como discusiones, conversaciones, informes o memorandos.
- **El Flujo de Información Electrónica**, en este caso toma las forma de interfaces electrónicas como aplicaciones, emails o Internet.



2. MATERIAL FLOW

Es esencialmente un **mapeo del viaje que emprenden los materiales** (en lugar de la información), desde el momento en que se recibe cualquier material dentro del área de fabricación, hasta el punto en que los clientes realmente reciben el producto terminado.
En el caso del **desarrollo de software** sería **desde el momento que el equipo de desarrollo dispone de la Solicitud de Funcionalidad hasta que se entrega un Incremento de la funcionalidad potencialmente publicable.** Identificaremos en cada fase:

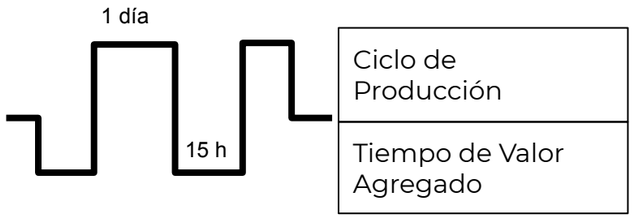
- **Cycle Time** → Tiempo que se tarda en ejecutar esta fase.
- **Set Up Time** → Cantidad de tiempo que se necesita para preparar dicha fase.
- **Uptime (%)** → Es el % de tiempo que la fase estuvo activa.

3. TIME LADDER

Es una escala de tiempo que tiene 2 partes:

UP Portion → Porción superior que muestra la cantidad de tiempo que **no se añade valor.**
DOWN Portion → Porción inferior que muestra la cantidad de tiempo donde **se añade valor.**

Indicando siempre la unidad de tiempo en que se mide (días, horas, minutos, segundos, etc).



EJEMPLO - DEPARTAMENTO DE RRHH

PROVEEDORES (Headhunting, Empresas Externas, etc.)

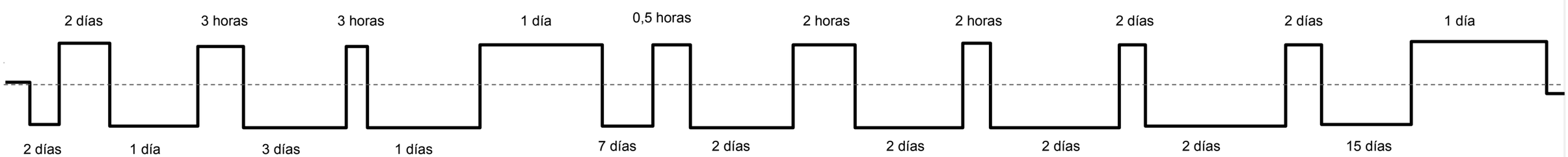
RRHH

DIRECCIÓN

CANDIDATOS

WasteRatio:	**7,5 (días) / 44,5 (días) = 0.16%**
Work Ratio:	**37,5 (días) / 44,5 (días) = 0,84%**

VALORACIÓN INICIAL	CONTACTO	ENTREVISTA PERSONAL Y TÉCNICA	PREPARACIÓN PRUEBA TÉCNICA	ENVÍO DE LA PRUEBA TÉCNICA	DEFENSA DE PRUEBA TÉCNICA	ENTREVISTA CON DIRECCIÓN	ENVÍO DE PROPUESTA ECONÓMICA	DECISIÓN DEL CANDIDATO	INCORPORACIÓN
• Demanda no se puede controlar. • Categorización de las entradas • Priorización de los candidatos • Descarte de contactos	• Envío de acuse de recibo a los candidatos • Notificar de derechos sobre protección de datos (GDPR) • Programación de la Entrevista Personal y Técnica • Descarte de Candidatos	• Priorización de contactos • Entrevista con personal de RRHH • Cuestionario de conocimientos técnicos • Entrevista Técnica • Generación de informe por los evaluadores • Descarte de candidatos	• Elaboración de la Prueba Técnica a defender por el candidato o Selección entre 24 existentes.	• Contacto con el candidato • Programación de la fecha para la defensa de la prueba en Autentia • Envío de la prueba técnica	• Defensa de la prueba (online o en oficinas) • Generación del informe por el evaluador • Programación entrevista con dirección • Descarte de candidatos	• Reunión con alguno de los directores • Obtención de información adicional para generar la oferta económica	• Generación de la propuesta económica • Comunicación de la propuesta económica al candidato	• El candidato tiene un margen de tiempo para aceptar la oferta económica. • Tras un periodo sin respuesta se da por descartado el candidato	• Preparación de la documentación a formar por el candidato (SS.SS, confidencialidad, riesgos laborales, etc.) • Preparación del material • Asignación de tutor • Asignación de tutor para el Onboarding

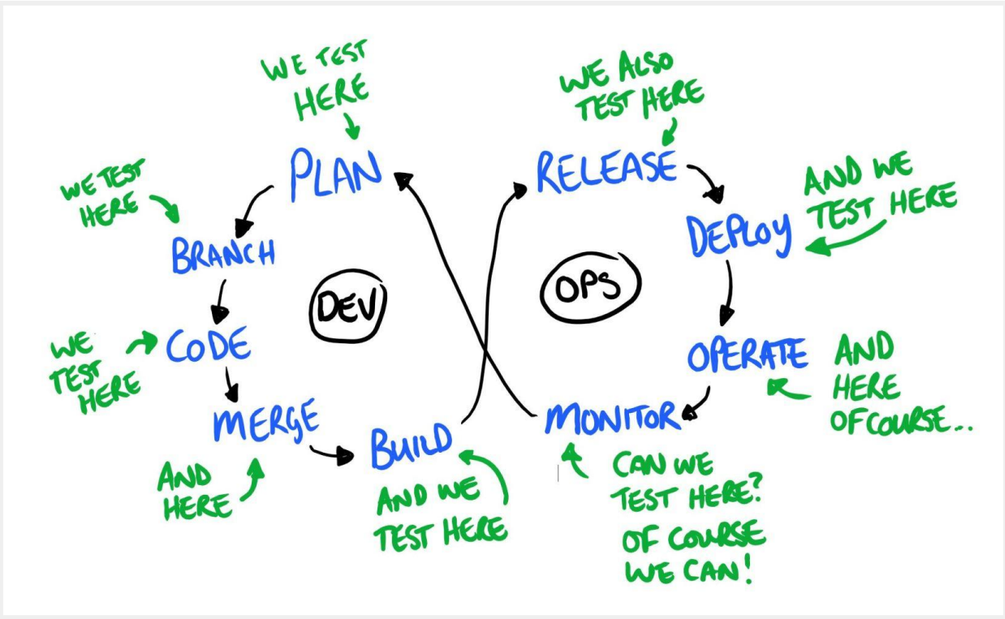


Definición

El **Agile Testing son prácticas de Testing colaborativas que suceden continuamente**, desde el inicio hasta la entrega y más allá, y que **respaldan la entrega frecuente de valor a nuestros clientes**. Las actividades de Testing **se centran en crear calidad en el producto**, usando rápidos ciclos de retroalimentación para validar nuestra comprensión. Las prácticas **fortalecen y respaldan la idea de que todo el equipo es responsable de la calidad.**

TESTING CONTINUO EN EL CICLO DEVOPS

El **Testing es una parte integral del desarrollo de Software**, entendemos las necesidades de nuestros clientes, las construimos, las entregamos y aprendemos como las usan y hacemos el **Testing parte integral del ciclo de DevOps**.



Fuente: Dan Ashby en su blog https://danashby.co.uk/2016/10/19/continuous-testing-in-devops/

LOS 10 PRINCIPIOS DEL AGILE TESTING

En el Agile Testing se introduce la idea de **10 principios** para los Agile Testers y no sólo para ellos sino **para cualquiera dentro del equipo:**

- Proporciona un feedback continuo.
- Entrega valor al cliente.
- Permite la comunicación cara a cara.
- Ten coraje.
- Sim.
- Practica la mejora continua.
- Responde al cambio.
- Auto-organízate.
- Céntrate en la gente.
- ¡Disfruta!

Testing Manifesto

Creado por Karen Greaves y Samantha Laing, el Testing Manifesto (el Manifiesto del Testing) refleja el cambio de mentalidad necesario para un enfoque de Agile Testing exitoso.

THE TESTING MANIFESTO

- **Valoramos el testing durante todo el proceso sobre las pruebas al final.**
  Tradicionalmente el testing ha sido una fase que se ha producido al final del desarrollo. Sin embargo, ***en la agilidad el testing es una actividad*** que necesita darse junto con la codificación, la documentación y el resto de operaciones.

  Si un tablero Kanban posee una columna de Testing, ten por seguro que el testing se está viendo como una fase y la actividad de los testers se está separando de la del resto del equipo. Como alternativa se recomienda una fase de Verificación donde una vez que el trabajo esté a DONE un compañero pueda revisarla.

- **Valoramos la prevención de errores en lugar de encontrar errores.**
  Debe hacerse lo antes posible, normalmente la mayoría de los errores se dan por asunciones sin aclarar en la toma de requisitos.
- **Valoramos la comprensión de las pruebas sobre la verificación de la funcionalidad.**
  Los testers deben ser los representantes del cliente en cada decisión del diseño, asegurándose que la feature satisfaga las verdaderas necesidades de estos y no solo las especificaciones.
- **Valoramos construir el mejor sistema sobre romper el sistema.**
  Tradicionalmente los Testers se han centrado en buscar la forma de romper el sistema, enfocándose en el Testing Negativo. Sin embargo, un tester se debe enfocar en el objetivo principal: crear un producto que aporte valor y cumpla su función como se espera.
- **Valoramos la responsabilidad del equipo por la calidad sobre la responsabilidad del probador.**
  Los testers por si sólos no pueden mejorar la calidad. Su función es la de determinar el nivel de calidad e informar de ella a los interesados. Sólo puedes mejorar la calidad uniendo tus esfuerzos a los del equipo.

Planificación de Pruebas en Contextos Ágiles

Para planificar las pruebas de forma efectiva, un equipo necesita considerar su contexto. Para entender tu contexto piensa en tres aspectos de este:

EL EQUIPO

No todos los equipos que se crean son iguales. Podemos tener 2 casos:

1. **Equipo pequeños**.
   a. Ubicados en la misma sala/edificio
   b. Buenas oportunidad para aprender los valores del resto y compartirlos
   c. La planificación es mucho más fácil
2. **Equipos grandes y distribuidos**.
   a. Trabajan en múltiples proyectos y con múltiples equipos
   b. Surge la necesidad de tener especialistas en el equipo
   c. Se incrementan las dependencias
   d. La planificación se vuelve más compleja

EL PRODUCTO

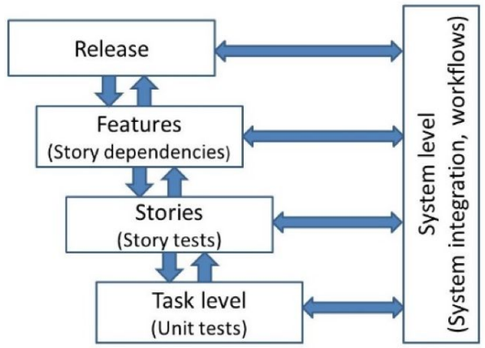
El nivel de calidad que las partes interesadas quieren depende del producto así como del tipo y cantidad de pruebas que pudiera ser requerido. Consideraciones del producto:

- **Tamaño:** ¿Cuánta gente lo va a usar?, ¿Está integrado con Apps Externas?
- **Riesgos:** Para reducir riesgos los equipos necesitan conocer el dominio del negocio.
  - Los expertos de dominio deben ayudar en la planificación de las pruebas.
  - ¿Tiene idea el equipo de como se usa el producto?
  - ¿Tienen todos los miembros del equipo conocimiento del dominio?

NIVELES DE DETALLE

El **Testing diferentes niveles requiere de una planificación extra,** los niveles identificados son:

- **Nivel de Release/Feature**, las actividades deben ser coordinadas entre equipos.
- **Nivel de Historias**, emplear la Specification By Example para incrementar el entendimiento de la historia y convertirla en pruebas.
- **Nivel de Tareas,** usar TDD para escribir pruebas unitarias de bajo nivel. También se puede emplear DDD como alternativa.



Guiar el desarrollo con ejemplos

Ejemplos concretos del comportamiento del sistema de lo deseado y lo no deseado ayuda a construir un entendimiento compartido de cada feature e historia.

BDD

BDD (Behavior Driven Development) captura ejemplos de escenarios en un lenguaje natural y específico del dominio.

Su sintaxis es:

GIVEN: Precondiciones
WHEN: Disparadores
THEN: Post Condiciones

Ejemplo:

GIVEN que he hecho una compra por valor de 9,75€
WHEN entrego al cajero 10 €
THEN espero 0,25€ de vuelta.

ATDD

ATDD (Acceptance Test-Driven Development) es una forma más genérica de guiar el desarrollo con ejemplos sin un lenguaje o reglas estrictas. Se lleva a cabo:

- **ANTES** de Implementar una necesidad todos los miembros del equipo contribuyen para generar escenarios de cómo se comportará dicha necesidad.
- **DESPUÉS** el equipo convierte esos escenarios en pruebas de Aceptación Automatizadas

SBE

SBE (Specification By Example), se empieza identificando los objetivos entorno a la Historia usando un planteamiento como el Impact Mapping.

A partir de aquí **el equipo obtiene ejemplos clave que se convierten en especificaciones**.

A medida que avanza el desarrollo, los ejemplos se perfeccionan **y se convierten en especificaciones ejecutables** que permiten validar el producto con frecuencia.

Estos ejemplos ejecutables junto con BDD y ATDD, **se convierten en documentación viva de la aplicación.**

Habilitando la Colaboración

La colaboración dentro del equipo y entre los equipos es una de las piedras angulares para hacer que los equipos ágiles tengan éxito. Sin embargo, nos encontramos con que muchos equipos no saben como empezar a construir estas relaciones. Te presentamos algunas prácticas que te pueden ayudar en ello.

PRÁCTICAS

Dinámica de los 3 Amigos

Es una herramienta poderosa para compartir un entendimiento compartidos sobre features, historias y como encajan estas en el producto. La dinámica de los 3 Amigos **consiste en una reunión entre el tester, el programador y el experto de negocio para examinar el incremento del producto antes, durante y después del desarrollo.**

Otros roles que pueden participar son: UX, un experto de Big Data, experto en DevOps, etc.

Dinámica del Impact Mapping

Ayuda a decidir qué features debemos construir y quizás incluso determinar qué prioridad deben tener. Empezamos con el objetivo de una característica, el "**why**", después identificamos quién podía ayudarnos a alcanzar ese objetivo "**who**". Por cada "who" preguntamos cómo podrían ayudarnos en alcanzar este objetivo "**how**". Para finalizar pensamos en qué entregables podrían resulta de cada impacto "**what**".

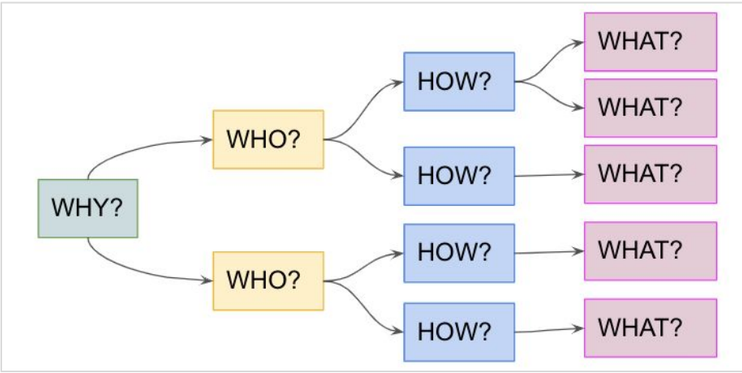


Figura 1: Impact Mapping

Dinámica del Example Mapping

Es una estupenda forma de explorar una feature así como el valor que debería entregar. Mediante esta dinámica se producen conversaciones que concretan ejemplos que se usan para clarificar nuestro entendimiento de las reglas de negocio.

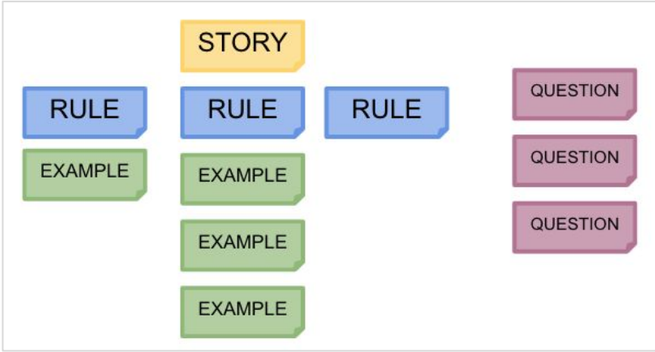


Figura 2: Example Mapping

Pruebas Exploratorias

En las pruebas exploratorias, **una persona interactúa con el sistema y observa el comportamiento real, diseñando pequeños experimentos.** Según lo que aprenden, adaptan el experimento y continúan aprendiendo más sobre el sistema. En el proceso, pueden hacer descubrimientos sorprendentes, incluyendo implicaciones de interacciones que nadie había considerado. **Las pruebas exploratorias exponen malentendidos sobre lo que se supone que debe hacer el software.**

TÉCNICAS

Personas, Jobs y Roles

Asumir el rol de una persona, **permite al equipo probar un producto desde la piel de otra persona y ver las cosas desde una perspectiva diferente,** pudiéndose superar la barrera a diferentes sesgos cognitivos inconscientes tales como el sesgo de confirmación.

Una **persona es un usuario ficticio que el equipo crea** y al que otorga ciertas características como una edad, nivel cultural, educación, experiencia, diferentes rasgos de personalidad, etc. Principalmente basados en el cliente objetivo al que va orientado el producto.

Combinar la persona con oficios o roles da un resultado mucho más favorable en las pruebas exploratorias.

Explorar en Parejas o Grupos

Cuantos más ojos haya sobre el problema mayores oportunidad de que se encuentre algo habrá.

Charters

Los charters te ayudan a organizar la información que necesitas para aprender sobre tu aplicación adecuadamente en sesiones de tiempo fijo. Funcionan mejor cuando se utiliza junto con la técnica Personas. Siguen el esquema:

EXPLORE el carrito de la compra de la web
WITH exploits de scripting
TO DISCOVER cualquier vulnerabilidad

Atributos de Calidad del Testing

Un atributo de calidad, también conocido como requisito no funcional, **describe las propiedades bajo las cuales una característica debe operar.** Podemos pensar en el como una restricción que el equipo debe considerar con cada Feature o Historia.

TIPOS

1. **DESARROLLO,** estos atributos vienen a representar el **CÓMO** desarrollamos nuestro código e incluyen:

    a. Mantenibilidad del Código
    b. Reusabilidad del Código
    c. Testeabilidad del Código

2. **OPERACIONALES,** son atributos que encaran la tecnología e incluyen:

Disponibilidad	**Interoperabilidad**	**Escalabilidad**	**Rendimiento**
Seguridad	**Confiabilidad**	**Robustez**	**Recuperabilidad**
Instalabilidad	**Usabilidad**	**Protectibilidad**	

Los equipos que esperan demasiado en el ciclo para prob
ar estos atributos (normalmente al final antes de lanzar el producto) s**e topan con un bloqueo total.** Este tipo de problemas son normalmente problemas de diseño y no pueden ser reparados tan tarde.

Para mitigar los riesgos el equipo debe colaborar de forma temprana, siendo proactivo. No esperes a que el Product Owner inicie la conversación.

Visualización de una Estrategia de Automatización de Pruebas

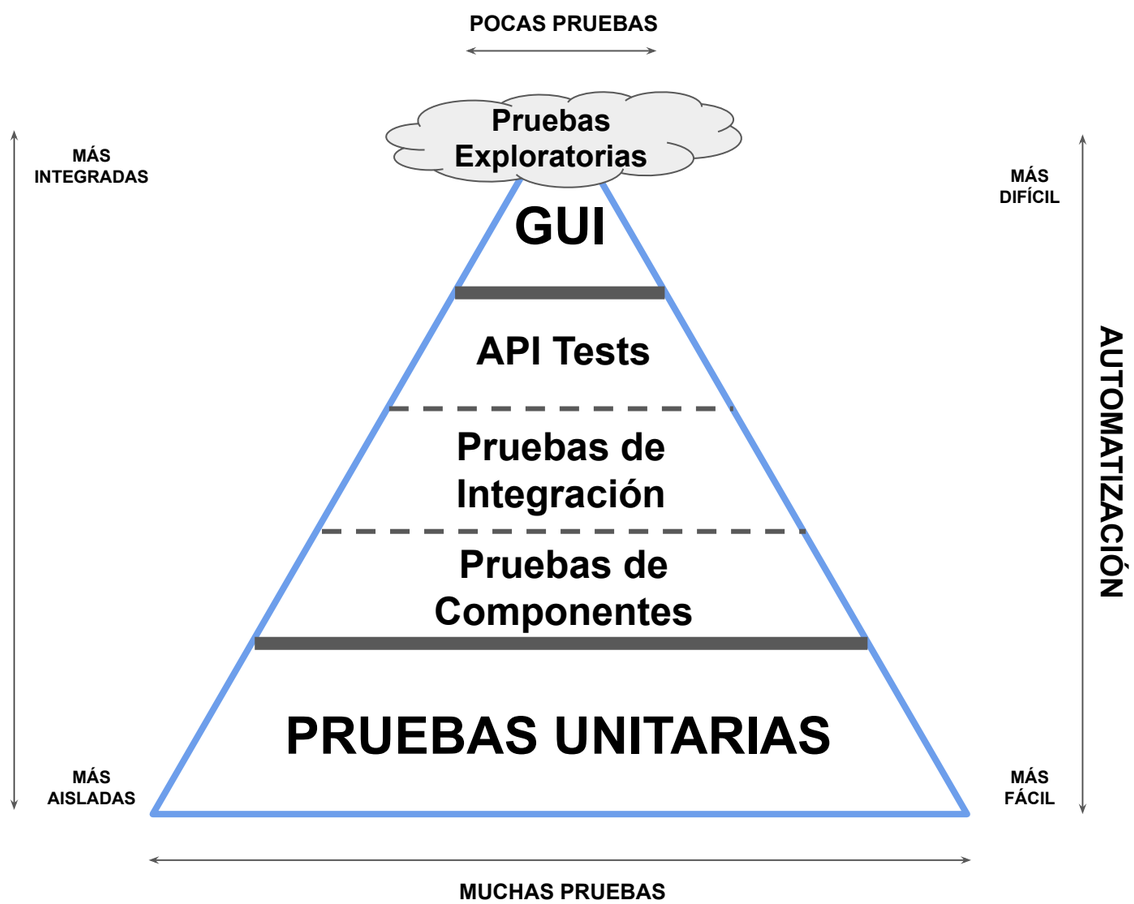
La Automatización de Pruebas es un reto constante para los equipos de desarrollo de software en cualquier parte. **Te mostramos cómo el uso de Modelos Visuales pueden ayudar a tu equipo a diseñar su propia estrategia de automatización.**

PIRÁMIDE DE AUTOMATIZACIÓN DE PRUEBAS DE MIKE COHN

Según la Pirámide de Mike Cohn **el grueso del testing** de cualquier proyecto **deberían ser las Pruebas Unitarias**, luego las **Pruebas de Componentes, Integración y API**. Y donde se debería poner menos esfuerzo será en las de **Interfaz de Usuario (GUI)**. **¿Por qué?**

Se trata de **anticiparse a los errores** que se puedan producir y tener controlados los elementos al más bajo nivel ya que esto nos asegura que no se vayan a propagar hacia arriba.
Además, la amplia cobertura de Pruebas Unitarias **nos documenta nuestro producto**. Hace que sepamos qué está fallando, dónde está el fallo y facilita su lectura *(incluso para las nuevas incorporaciones en el equipo).*

Normalmente **podemos encontrar un planteamiento diferente a la pirámide de Mike Cohn** y este no tiene por qué estar mal necesariamente (*pirámide invertida o reloj de arena*), aunque si la automatización actual no cubre las necesidades del cliente la imagen visual de nuestra estrategia de automatización de pruebas puede ayudarnos a detectar qué cambios son necesarios.



El Agile Tester

A diferencia del tester tradicional encargado de elaborar suites de prueba (a partir de diseños funcionales), y luego ser un mero ejecutor de checklist. El Agile Tester trabaja con el equipo siendo su prioridad principal la calidad y domina las herramientas de automatización, así como las metodologías ágiles.

CARACTERÍSTICAS Y HABILIDADES DEL AGILE TESTER

Las **principales características** de un Agile Tester son:

- *Trabaja codo con codo con los Stakeholders*
- *Proporciona experiencia en las actividades de testing*
- *Se empareja con otros desarrolladores y testers*
- *Dan apoyo al Product Owner*
- *Organiza y adapta la estrategia de QA general*
- *Obtiene buenos datos de las pruebas*
- *Identifica riesgos*
- *Es experto en guiar el desarrollo con ejemplos empleando para ello TDD, ATDD y SBE*
- *Trabajo con el equipo para asegurar la observabilidad de las pruebas para poder probarlas y monitorizarlas en Producción*
- *Tiene pasión por la calidad y las pruebas, abogan por ellas y las defienden*

El Agile Tester tiene una amplia base de habilidades y conocimientos generales que apoyan el conocimiento profundo en el área del Testing.

Las **principales habilidades** de un Agile Testers son:

- **Colaboración,** para participar en prácticas como Mind Mapping o Example Mapping
- **Facilitación,** para ayudar a los miembros del equipo a comunicarse mejor
- **Enseñanza,** para compartir su conocimiento con otros miembros del equipo
- **Coaching,** para ayudar al equipo a identificar problemas y diseñar experimentos para mejorar
- **Comunicación,** para dar y recibir feedback de forma efectiva

¿Qué es el Evidence-Based Management?

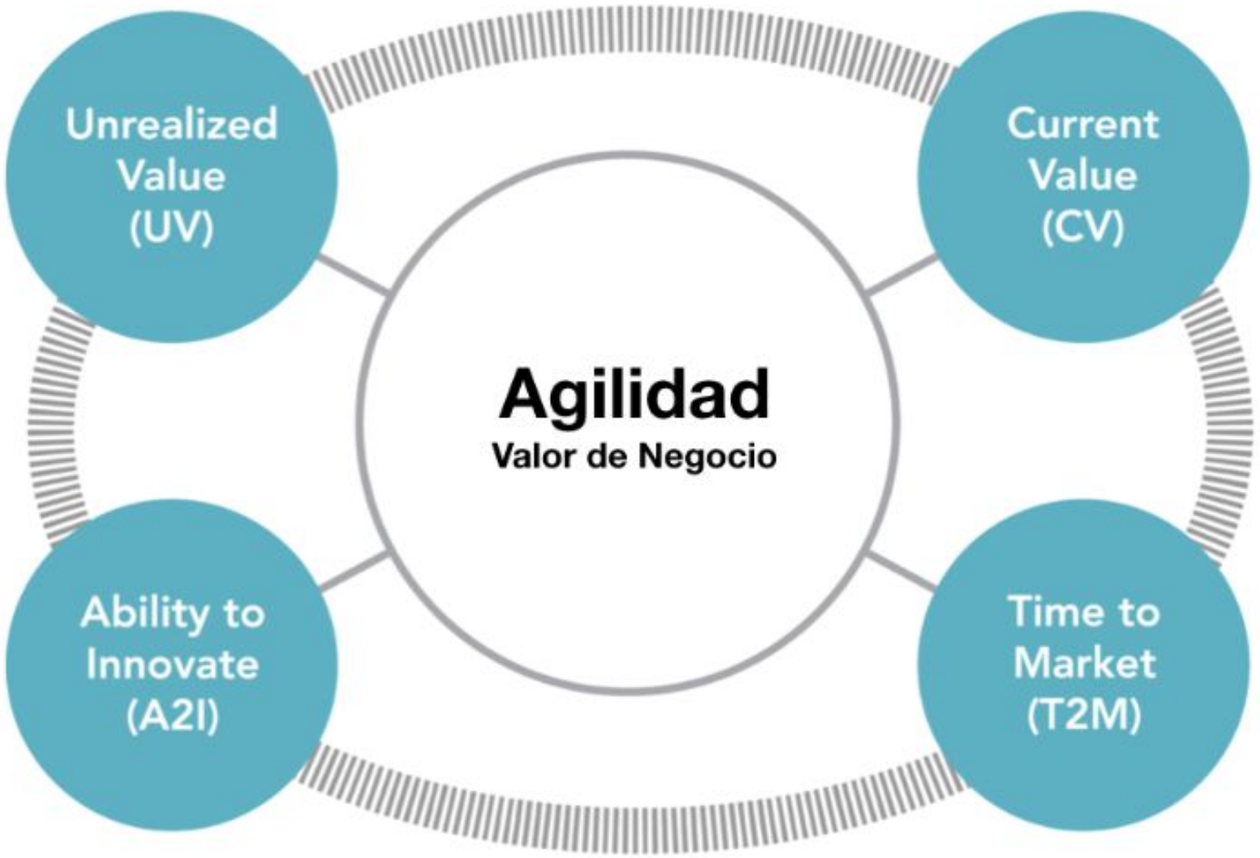
La **Gestión Basada en Evidencias (EBM)** es un enfoque empírico que brinda a las organizaciones la **capacidad de medir el valor que entregan a los clientes así como los medios por los cuales entregan ese valor**, y la capacidad de usar esas medidas para guiar las mejoras en ambos.

¿EN QUÉ CONSISTE?

EBM organiza las métricas de valor **(KVM o Key Value Measures)** en cuatro áreas clave de valor **(KVA o Key Value Areas).**

Cada KVA se enfoca en un aspecto diferente de valor o a la capacidad de la organización para entregar valor. Las organizaciones que no tienen los cuatro KVA fortalecidos pueden ofrecer valor a corto plazo, pero no tendrán garantías de mantenerlo a futuro.

La entrega de valor, tener a los accionistas felices y a los empleados satisfechos **(Current Value)** son importantes, pero las organizaciones también deben demostrar que pueden satisfacer la demanda del mercado con una entrega puntual **(Time to Market)** mientras que pueden mantener la innovación a lo largo del tiempo **(Ability to Innovate)**. La inversión continua en el producto se justifica en función de las medidas a fecha de hoy **(Unrealized Value)** que podrían realizarse si el producto poseyera las capacidades adecuadas.



MÉTRICAS DE ENTREGA Y DE PROPIETARIO

Conviene aclarar que hay dos tipos de métricas de negocio que suelen utilizarse:

- **Métricas de entrega (delivery metrics).** Son importantes para guiar las prácticas operativas, pero peligrosas si se usan como falsas representaciones del valor y se fijan como objetivos. Eso sí, son más fáciles de medir normalmente. Deberían considerarse neutrales en cuanto a la creación de valor (afectan, pero indirectamente).
- **Métricas de propietario (owner metrics).** Reflejan los verdaderos resultados del negocio, siendo verdaderas métricas de VALOR con mayúsculas.

CURRENT VALUE (CV)

Revela el **valor que la empresa (o el producto) entrega a sus clientes, hoy**. El objetivo es maximizar la entrega de valor a clientes, empleados y accionistas considerando lo que existe actualmente. Entre sus métricas clave de valor **(KVM)** más relevantes están:

- **Ingresos por empleado:** Ingresos brutos / número de empleados.
- **Ratio de costes de producto:** total de gastos del producto/sistema medido comparados con los ingresos.
- **Satisfacción del empleado.** Puede reflejarse en un índice global, basado en datos de encuestas o incluso de otras técnicas como las retrospectivas o los calendarios Niko-Niko.
- **Satisfacción del cliente.** Puede reflejarse en un índice global, basado en datos de encuestas, de redes sociales, antigüedad media de clientes (fidelidad), etc.

ABILITY TO INNOVATE (A2I)

La **capacidad de innovar** es lo que permite a la organización y a sus productos o servicios cubrir mejor las necesidades de sus clientes y no quedar desfasados. Es un factor clave para la sostenibilidad a largo plazo del negocio, la captación y fidelización de clientes. Entre sus métricas clave de valor **(KVM)** más relevantes están:

- **Índice de uso de características:** medida de uso de las características del producto. Permite identificar las más usadas y las que rara vez o nunca se usan.
- **Tasa de innovación:** porcentaje de esfuerzo o coste dedicado a desarrollar nuevas capacidades del producto, dividido entre el esfuerzo o coste total del producto.
- **Tendencia de defectos:** aumento o disminución de defectos desde la última medición.
- **Índice de trabajo en el producto:** porcentaje de tiempo que los equipos dedican trabajando en el producto para generar valor.

EBM - Key Value Areas

autentia

CÓMO MEJORAR EMPÍRICAMENTE USANDO EBM

1. **Cuantifica el valor**
2. **Mide los KVMs (indicadores leading y lagging)**
3. **Selecciona los KVA a mejorar**
4. **Dirige experimentos prácticos para mejorar los KVAs seleccionados**
5. **Evalúa los resultados**

- **Indicadores Leading:** tienen como función **informar** sobre tendencias que aún no han sucedido. Es un indicador proactivo.
- **Indicadores Lagging:** se emplea para medir que está pasando actualmente. Es un indicador reactivo.

  Ej: En un equipo de desarrollo un ***indicador Leading*** sería la disponibilidad del equipo y un ***indicador Lagging*** el retraso en las tareas.

TIME TO MARKET (T2M)

Expresa la **capacidad** de la organización **para entregar rápidamente nuevos productos o servicios**, o nuevas funcionalidades y capacidades de los ya existentes. Cuanto más se acorte el T2M, más rápido se podrá adaptar a los cambios del mercado. Entre sus métricas clave de valor **(KVM)** más relevantes están:

- **Frecuencia de Releases:** cada cuánto se entrega una nueva versión de producto.
- **Periodo de estabilización de Release:** tiempo dedicado corrigiendo problemas del producto desde que los desarrolladores indican que se puede lanzar una release y cuando realmente se lanza a los clientes.
- **Cycle Time:** tiempo desde que se inicia un trabajo hasta que se entrega en una release.

UNREALIZED VALUE (UV)

Sugiere el **valor futuro potencial** que podría alcanzarse si la organización pudiese cubrir perfectamente las necesidades de todos sus clientes potenciales.

Entre sus métricas clave de valor **(KVM)** más relevantes están:

- **Cuota de mercado:** el porcentaje del mercado controlado por el producto.
- **Gap de satisfacción del cliente o usuario:** brecha o diferencia entre la experiencia deseada del cliente o usuario y su experiencia real.

¿QUÉ ES UNA RETROSPECTIVA?

Es una oportunidad para que el equipo se inspeccione, aprenda y mejore. En esta sesión el equipo mira hacia atrás a los eventos y comportamientos pasados y se enfoca en responder:

- *¿Qué fue bien?*
- *¿Qué no fue bien y se puede mejorar?*
- *¿Con qué nos vamos a comprometer a hacer para mejorar?*

Se llevan a cabo al final del Sprint y suelen durar entre 60' y 180' dependiendo de la longitud del Sprint.

¿QUÉ SE HACE EN UNA RETROSPECTIVA?

Las retrospectivas pueden ser más o menos sofisticadas, aunque se suelen seguir los 5 pasos sugeridos por Esther Derby y Diana Larsen:

1. **Prepara el escenario**, define un objetivo y dale tiempo a la gente para que llegue y se acomode.
2. **Obtén información**, ayuda al equipo a recordar (cada uno tiene una visión diferente del mundo).
3. **Genera conocimiento**, ¿por qué las cosas pasaron de la forma que se dieron? Busca patrones, visualiza la imagen completa.
4. **Decide qué hacer**, crea un plan de acción sobre problemas concretos y asigna un responsable.
5. **Cierra la retrospectiva**, clarifica el seguimiento, da agradecimientos y deja el final claro.

¿QUIÉN ASISTE A LA RETROSPECTIVA?

El **equipo al completo** sea cual sea el contexto en el que te encuentres. Por ejemplo:

- En **Scrum**, asiste el Equipo de Desarrollo, el Scrum Master, el Product Owner y se invita a todos aquellos externos al equipo que se hayan podido ver afectados por algún tema en concreto.

- En **Kanban**, asiste el equipo al completo; el Service Delivery Manager y el Service Request Manager si están identificados.

¿QUÉ NO ES UNA RETROSPECTIVA?

- **No es una reunión en la que buscar culpables**. De hecho, y sobretodo, en las primeras sesiones hay que dejar claro las directivas de la retrospectiva: «*Independientemente de lo que descubramos, entendemos y realmente creemos que todos hicieron el mejor trabajo que pudieron, dado lo que sabían en ese momento, sus habilidades y capacidades, los recursos disponibles y la situación en cuestión.*»

- **No es una reunión más en la que simplemente se va a hablar.** La retrospectiva debe dar lugar a un plan de acciones concretas y a la asignación de las personas que las llevarán a cabo.

En las siguientes fichas se muestran algunas de las Retrospectivas más utilizadas.

Retrospectivas - El Barco (Sailboat)

autentia

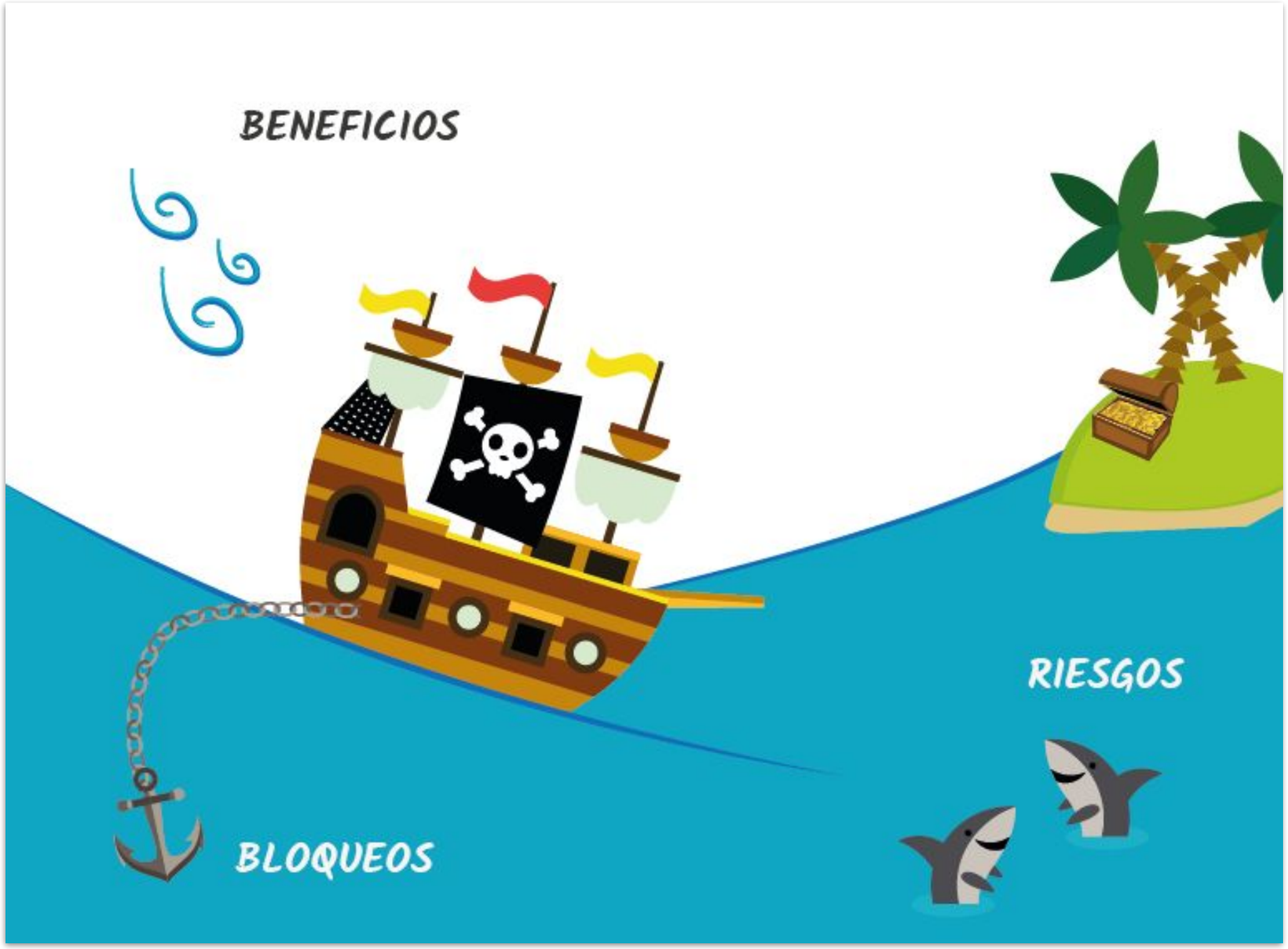
DESCRIPCIÓN

Esta es una técnica muy divertida para reflexionar sobre oportunidades, riesgos y problemas que ha habido en el sprint.

Es muy flexible ya que permite meter nuevos elementos en la dinámica según las necesidades identificadas.

MATERIALES

- 1 Pizarra.
- 1 Rotulador por participante.
- Post-it, a ser posible de colores diferentes.



INSTRUCCIONES

1 - La dinámica del barco conlleva un poco de **habilidades en storytelling** así como **artísticas.** Empezamos la dinámica dibujando un **barco**, que es donde **viaja todo el equipo,** y una **isla** del tesoro a lo lejos, esta representará los **objetivos del equipo**, ¿qué quiere alcanzar? Es **importante** que el equipo *indique en la pizarra cuál es ese tesoro que quiere conseguir.*

2 - Continuamos nuestro StoryTelling dibujando además:
– **un sol:** representa las cosas que nos hacen más fácil la travesía.
– **Un ancla:** hace referencia a todas aquellas cosas que nos retienen o retrasan.
– **Rocas o iceberg:** son los riesgos que nos encontramos en nuestra travesía.
– **Viento:** viene a representar aquellas cosas que nos acercan a nuestro objetivo, el tesoro.

3 - Una vez finalizado el StoryTelling damos un tiempo al equipo para que **individualmente** i**ndique** en post-it, todos aquellos **elementos que hayan identificado en el Sprint**. Se puede imponer un número mínimo de post-it a escribir por miembro.

4 - Posteriormente, **cada miembro** del equipo sale a la pizarra a colocar cada uno de los post-its para **dar su opinión**. Una vez que todos los miembros den sus opiniones, **se agrupan los post-it** por tema en común.

5 - Después de que se hayan comentado todas las opiniones, **se analizan todas las cosas que hayan pasado**, tratando *sobretodo los problemas y riesgos surgidos.* Si existen muchos temas a tratar, se pueden asignar 3 puntos a cada miembro del equipo y hacer una votación.

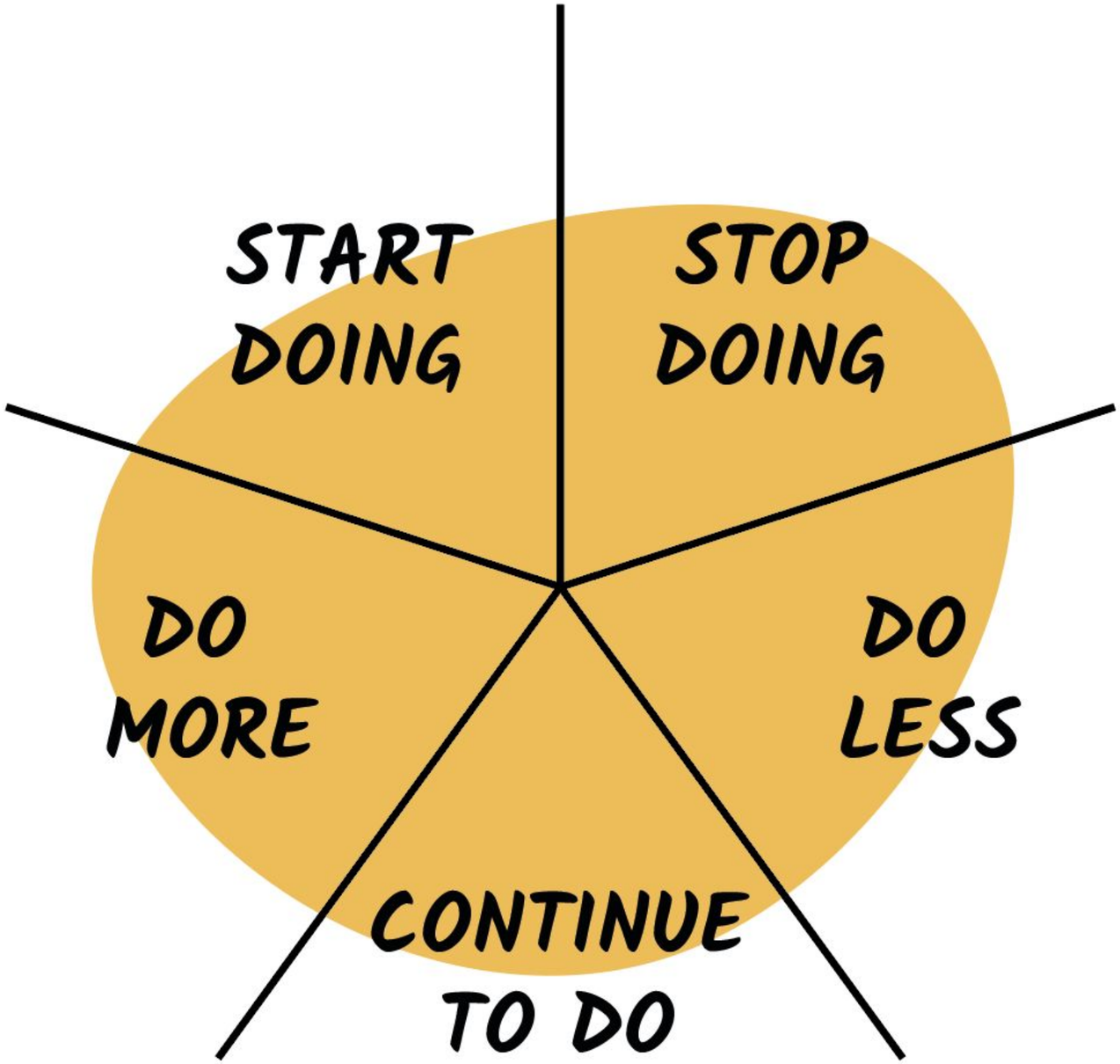
6 - Para finalizar la sesión se **genera una lista de acciones** para tratar todos los asuntos comentados anteriormente o bien priorizarlos. **Se asigna una persona responsable** de que se cumpla, así como una **fecha**. **Importante: estas acciones de mejora se deberán en el Backlog del equipo.**

DESCRIPCIÓN

Patrick Kua creó una útil y popular técnica para estructurar de manera más eficiente tus reuniones de retrospectiva. A la técnica la llamó «estrella de mar» (The Retrospective Starfish), y se basa en usar un diagrama con forma de estrella de mar que permite (ver la figura de de la derecha) crear cinco áreas de cosas a tratar, evitando así centrarse solo en lo bueno y lo malo.

MATERIALES

- 1 Pizarra.
- 1 Rotulador por participante.
- Post-it, a ser posible de colores diferentes.



INSTRUCCIONES

1 - Empezamos dibujando en la pizarra la estrella con los cuadrantes bien diferenciados. Posteriormente nombraremos a cada cuadrante con: **Start Doing, Stop Doing, Do More, Do Less, Continue To Do.**

2 - Posteriormente se explica cada cuadrante, empezamos con **Start Doing:** en este cuadrante incluimos todo **aquello novedoso** para el equipo **con lo que queramos experimentar** o bien se trate de un patrón **o solución** que previamente se haya probado **que funciona y consideramos que debemos usar.** *Por ejemplo, utilizar una foto de los miembros del equipo en el tablero Kanban para identificar a quién está trabajando en qué ítem.*

3 - **Do More:** hace referencia a todas aquellas **prácticas** que se están poniendo **en uso** y que vemos **que son útiles** por lo que consideramos **que debemos continuar haciéndolas y mejorarlas.** Normalmente, serán cosas que un día pusimos en Start Doing y encontramos útiles. *Siguiendo con el ejemplo anterior, hay gente a la que no le gusta poner su foto y prefiere otro mecanismo. En este caso al refinar se deciden usar personajes de Los Simpson para identificar a los miembros del equipo y haciéndolo más divertido.*

4 - **Continue To Do:** indicaremos **todo aquello que** hemos estado haciendo y habiéndose refinado **no se puede mejorar pero nos es útil** y queremos mantenerlo. Se trata de la **evolución lógica** a la madurez **del Do More**, es una actividad que no necesita refinarse más y nos sirve tal cual.

5 - **Do Less:** aquí indicaremos **todo aquello que empezamos a hacer,** puede ser que empezó en un Start Doing, **pero** hemos visto que **no** nos **aporta valor o ha perdido prioridad** con respecto a otras acciones. También se pueden considerar aquí **prácticas** a las **que deberíamos de quitar una parte** porque no nos aporta valor. *Por ejemplo, la hoja de gastos mensual de la empresa que necesita ser simplificada.*

6 - **Stop Doing:** en esta sección indicaremos aquellas prácticas que el equipo considera que no aportan valor y se pueden desechar. Puede ser la evolución de una práctica que estuvo en el Do Less.

7 - **Para finalizar** la sesión **se discutirán los temas con más incidencias y se discutirán las acciones de mejora.** Si existen múltiples, se puede utilizar un mecanismo de votación para que el sólo trate las que consideran más importantes.

Retrospectivas - Las 4 L's

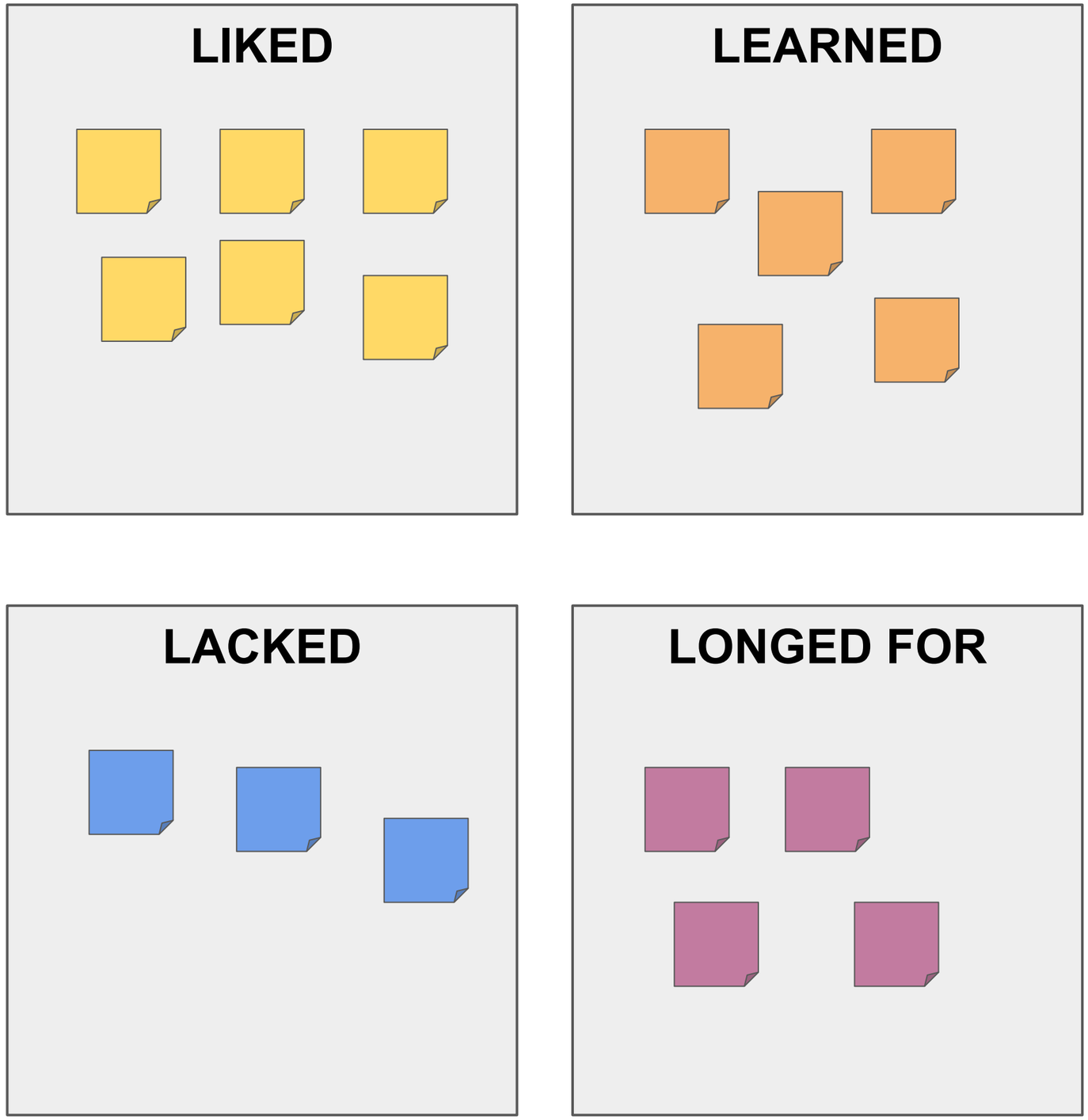
autentia

DESCRIPCIÓN

Las 4L's es una gran dinámica donde se obtiene poderosa información del equipo que enfatiza en su aprendizaje. Es una dinámica, que según nuestra experiencia, se centra en los aspectos positivos que se han dado en el Sprint.

MATERIALES

- 1 Pizarra.
- 1 Rotulador por participante.
- Post-it, a ser posible de colores diferentes.



INSTRUCCIONES

1 - Divide el tablero en las 4 zonas siguientes (pueden ser columnas consecutivas o en forma de cruz):

– **Liked**: cosas que te han gustado.
– **Learned:** cosas que has aprendido.
– **Lacked:** cosas que has echado en falta.
– **Longed For**: cosas que hubieras deseado que ocurriesen.

2 - Pide a **los miembros** del equipo que de forma individual **escriban notas** en post-it **referentes a cada una de las áreas** del tablero. Indica que escriban al menos un post-it en cada columna, lo ideal es que fuesen al menos 3 post it.

3 - Posteriormente solicita que **cada miembro** coloque en la pizarra sus post-its y **dé su opinión al resto del equipo.**

4 - **Mantén una conversación grupal sobre las notas** , incidiendo en aquellas con más frecuencia. Si observas que existen demasiadas siempre puedes utilizar un sistema de votación para **seleccionar las que son más prioritarias** para el grupo.

5 - Por último y una vez mantenida la discusión, **genera una lista de acciones de mejora** para tratar los asuntos más relevantes que hayan surgido, **selecciona un responsable para cada una** de ellas y **asígnales un tiempo estipulado de resolución. Importante: estas acciones de mejora se deberán en el Backlog del equipo.**

NOTA: Existe una **variante** de esta retrospectiva en la que sólo se utilizan 3L's: Liked, Learned y Lacked.

¿Qué son?

La estimación ágil **permite determinar el esfuerzo, la complejidad, el valor así como el tamaño de un proyecto** y nos va a permitir planificar reduciendo los riesgos, la incertidumbre, dando soporte a una mejor toma de decisiones, estableciendo la confianza y transmitiendo la información.

PUNTOS DE HISTORIA

Los equipos ágiles separan la **estimación por tamaño,** de la estimación por duración.

Los Puntos de Historia **son una unidad de medida para expresar el tamaño general de** una Historia de Usuario, Feature o cualquier otra **pieza de trabajo**.

Los Puntos de Historia **son un valor relativo** y cuando se asocian a una Historia de Usuario representan el tamaño general de la Historia.

Cuando se estima utilizando los Puntos de Historia se **utiliza la serie Fibonacci** para dar los valores numéricos, ya que esta nos evita cualquier sesgo cognitivo y al no ser lineal muestra la incertidumbre que existe al estimar grandes unidades de trabajo.

VELOCIDAD

La velocidad **es una medida de la tasa de progreso de un equipo por iteración.** Al final de cada iteración, un equipo puede ver las historias que han completado y calcular su velocidad sumando las estimaciones en Puntos de Historia dadas a cada Historia de Usuario completa.

Conocer la velocidad permite hacer correcciones en la estimación del trabajo que se compromete a realizar.

Ejemplo: *Supón un equipo que estima un proyecto en 300 Puntos de Historia (PdH) de trabajo. Inicialmente creen que podrán completar 25 PdH por iteración, lo cual significa que podrán terminar en 12 iteraciones, sin embargo, la velocidad observada es de 20 PdH. Sin reestimar, habrán identificado que el proyecto les llevará 15 iteraciones y no 12.*

DÍAS IDEALES

El tiempo que te llevaría terminar un proyecto sin tener en cuenta las tareas adicionales (tareas de otros proyectos, reuniones, correos a los que responder, llamadas telefónicas, etc.), **son días ideales.**

No se debe confundir días ideales con días transcurridos.

Ejemplo: *En un partido de baloncesto el día ideal (tiempo ideal) serían 60 minutos (cuatro cuartos de 15 minutos) pero lo normal es que un partido de baloncesto se acerque más a una duración de 2 horas (tiempo transcurrido). El motivo se debe a todas las interrupciones que hay durante el juego.*

Estimación por Analogía

Existen varias técnicas de estimación. Te presentamos las más comunes que se utilizan hoy en día. Cada una de ellas puede ser utilizada de forma individual pero s**i buscas los mejores resultados deberías usarlas de forma combinada.**

ESTIMACIÓN POR ANALOGÍA

Basándonos en el estudio *"Size-Estimation of Familiar Objects under Informative and Reduced Conditions of Viewing"* por H. Richard Schiffman (vol. 80 pp 229-235).

El estudio revela que los humanos **somos mejores estimando el tamaño de un objeto cuando disponemos de otro objeto familiar con el cual compararlo**. En agilidad lo llamamos **"estimación relativa".**

Al estimar por analogía, el estimador **compara la historia que se estima con una o más historias.** Si la historia tiene el doble de tamaño, se le da una estimación dos veces mayor.

Al estimar de esta manera, no se comparan todas las historias con una sola línea referencia universal. En cambio, **estimas cada nueva historia contra una variedad de las que ya se han estimado**. Esto se conoce como **triangulación.**

Ejemplo: *compara la historia que se quiere estimar con un par historias ya estimadas. Para decidir si una historia debe estimarse en cinco Puntos de Historia, observa si parece un poco más grande que una historia que calculó en tres y un poco más pequeña que una historia que calculó en ocho.*

PROS Y CONTRAS

FÁCIL DE HACER SI...

tienes un base de numérica que se pueda usar para las comparaciones.

INDEPENDIENTE DE LA METODOLOGÍA

Y además se puede usar con aplicaciones de cualquier tamaño.

PUEDE USARSE EN...

HdU's, épicas u cualquier otro documento de requisitos ágiles.

SERÁ TAN BUENO...

como los ejemplos que tengas en tu base de datos y sus caracterizaciones.

Affinity Mapping

Una dinámica bastante fluída para poner en práctica la técnica de estimación por analogía es el **Affinity Mapping**. Es una dinámica muy fácil que **pretende organizar la información**. La palabra Affinity hace referencia a **características similares** que sugieren una relación, en nuestro caso el tamaño. Para llevarla a cabo sólo necesitas post-its, rotuladores y una pared libre donde trabajar.

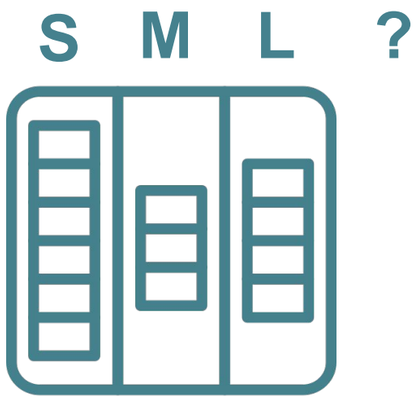
EJECUCIÓN DEL AFFINITY MAPPING

1° PASO: ESTIMACIÓN RELATIVA SILENCIOSA

El equipo distribuye en una mesa todas las Historias de Usuario y cada miembro en silencio coge un montón que estima en su sitio.
Se suele emplear la estimación por tallas de camiseta.

2° PASO: EDICIÓN DEL MURAL

En la pared se colocan los post-it en función de la estimación asignada.



3° PASO: REUBICAR ELEMENTOS

Una vez colocados todos las Historias en el tablero se discuten entre el equipo las estimaciones dadas y se reubican aquellos elementos cuya estimación se haya modificado.

4° PASO: DISCUSIÓN CON EL PRODUCT OWNER

El Product Owner hace sus observaciones referentes a las estimaciones. Aquellas Historias de las que se dudaba su tamaño por falta de información se discuten y detallan.

5° PASO: EXPORTAR A LA HERRAMIENTA DE GESTIÓN

El equipo traslada todas las Historias con sus estimaciones a la Herramienta de Gestión en uso.

Estimación por Consenso en grupo

Existen varias técnicas de estimación, te presentamos las más comunes que se utilizan hoy en día. Cada una de ellas puede ser utilizada de forma individual pero s**i buscas los mejores resultados deberías usarlas de forma combinada.**

ESTIMACIÓN POR CONSENSO EN GRUPO

Es un proceso de **toma de decisiones grupal** en el que **los miembros se desarrollan y acuerdan apoyar una decisión en el mejor interés de todo el grupo u objetivo** común.

Numerosos psicólogos han estudiado el tema y tras decenas de experimentos han determinado que la media de opiniones en consenso eran más acertadas que la media de las opiniones privadas de cada individuo del grupo.

Tradicionalmente en los proyectos se tenía en cuenta la **opinión de un experto,** este **enfoque es menos útil en proyectos ágiles** que en proyectos tradicionales.

En un proyecto ágil, las estimaciones se asignan a historias de usuarios u otra funcionalidad valorada por el usuario. Es probable que el desarrollo de esta funcionalidad requiere una variedad de habilidades lo que normalmente dará lugar a que sea realizada por más de una persona.

Este hecho hace que sea difícil encontrar expertos adecuados que puedan evaluar el esfuerzo en todas las disciplinas.

Referencias: *Francis Galton y sus experimentos en 1906. "El carnicero y los aldeanos". The Wisdom of the Crowds por James Surowiecki.*

PROS Y CONTRAS

PERMITE HACER MEJORES

- Decisiones.
- Implementaciones.
- Relaciones de grupo.

HACE QUE LA GENTE SE SIENTA

- Apoderada.
- Escuchada.
- Valorada.

SUS OBJETIVOS

- Colaborativo.
- Cooperativo.
- Igualitario.
- Inclusivo.
- Participativo.
- Búsqueda de acuerdos.

CONTRAS

- Consume más tiempo.
- Genera presión para que haya conformidad.
- No define claramente la discusión.
- Permite que una persona o grupo domine la discusión.

Estimación por Desagregación

Consiste en dividir una Historia de Usuario o Feature en piezas más pequeñas que sean más fácil de estimar.

ESTIMACIÓN POR DESAGREGACIÓN

Cuando nos topamos con una **característica (feature) grande** en comparación con el resto nos encontramos con que **se hace bastante difícil de estimar**, a esta dificultad se suma la poca existencia de elementos con los que comparar.

La **solución** a este asunto consiste en **desglosar estos elementos grandes en otros más pequeños** más fáciles de comprender y estimar. Sin embargo, hay que tener cuidado de no ir demasiado lejos con este enfoque y atomizarlo todo.

Existen diferentes enfoques para dividir estas características grandes en otras más pequeñas como son:

- **Por el tipo de dato que admite.**
- **Por las operaciones inherentes en la característica.**
- **Segmentando por seguridad, manejo de errores, registro, etc.**

En cualquier caso hay que **evitar dividir las características en las tareas de desarrollo** que sean necesarias para implementar la función. Así como debes **evitar agregar** a una característica aquellas **modificaciones posteriores** que no son necesarias para la entrega del elemento.

PROS Y CONTRAS

FÁCIL Y AMENA

Se puede usar en combinación con otros tipos de técnicas (tallas de camiseta, cubos, etc.)

DESARROLLADA POR UN EQUIPO

Para una aplicación específica, puede usarse para las métricas ágiles (velocidad, etc.)

MEDIDA DE TAMAÑO RELATIVA

Que desarrolla un equipo y casi nunca es válida para otros equipos trabajando en el mismo proyecto.

FALTA DE ESTÁNDARES

Hace que cada uno use el enfoque de forma única y no estandarizada.

El Planning Poker

Existen **combinaciones de varias técnicas de estimación** que permiten potenciar sus ventajas, ser aplicadas en proyectos grandes y compensar sus debilidades. Una de ellas es el **Planning Poker**.

EL PLANNING POKER

El **Planning Poker**, también llamado Scrum Poker, es una técnica de estimación gamificada **basada en consenso**, mayormente usada para **estimar esfuerzo o tamaño relativo** de los objetivos de desarrollo en el desarrollo de software.

Las sesiones de Planning Poker **se inician con la lectura por parte del Product Owner o la parte interesada de aquellas Historias de Usuario** o features que se quieren estimar.

Cada miembro del equipo de desarrollo dispondrá de una **mazo de cartas numeradas con la secuencia Fibonacci** (1, 2, 3, 5, 8, etc.) que representará la unidad de estimación acordada pudiendo ser puntos de historia, días ideales o cualquier otra.

A continuación, **el equipo de desarrollo discute la feature**, haciendo al Product Owner todas las preguntas que sean necesarias para clarificarla. Finalmente, cada miembro en privado selecciona una carta que representa su estimación. Todas las cartas son reveladas al mismo tiempo.

Si hay consenso se asigna la estimación y si no se discuten las estimaciones más altas y bajas y se vuelve a estimar y **se repite el proceso con cada feature**.

Estas fichas son un recurso vivo,
¡anímate a hacernos sugerencias y comentarios!

@autentia

Puedes descargar la última versión en:

https://autentia.com/libros/

Además, encontrarás los libros gratuitos
de Roberto Canales sobre metodologías ágiles.

Agradecimientos: THINKING WITHYOU https://thinkingwithyou.com/