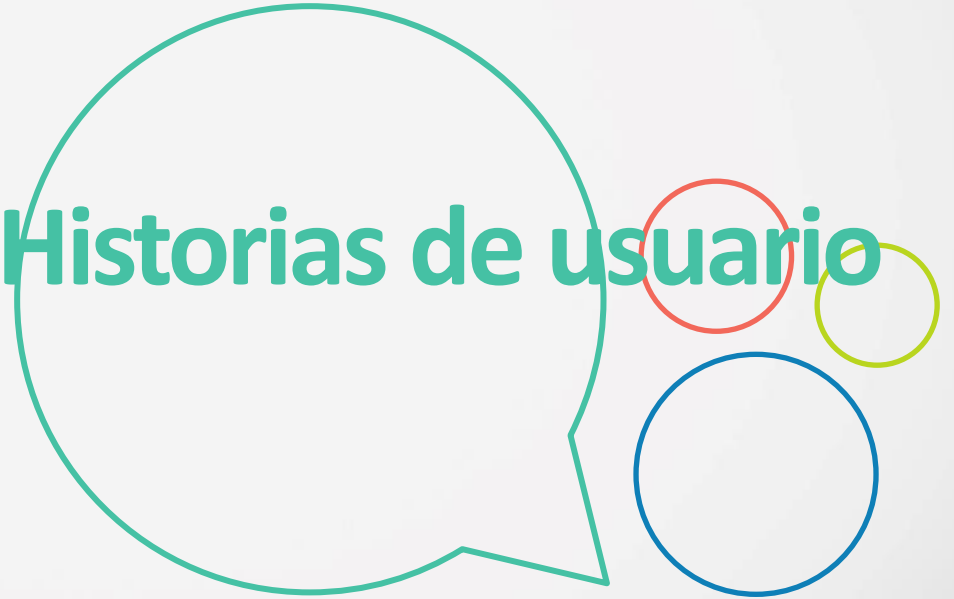







Taller Historias de usuario



Historias de usuario

Historia de Usuario (2) Como <Tipo de Usuario>, Quiero, <Hacer algo>, Para <Propósito>	Personas (1) Personas, Usuarios o consumidores finales, Interesados, Patrocinadores, Equipo de desarrollo, Equipo de soporte	Contexto (4) Necesidad origen, Escenarios de uso de la historia, Dominio del negocio, qué áreas del negocio usan o se impactan por la historia, Reglas del negocio que afectan la historia, Épica origen. No todas las historias provienen de una épica en particular pero si es así, es bueno conocerla, Alcance de la historia	
Criterios de Aceptación (3)		Definición de Preparado (5) Que se debe cumplir para que la historia se pueda construir?	Definición de Terminado (6) Condiciones de Terminado específicas a la historia
Resultado Esperado (7) Solución esperada, Que se quiere cumplir con la Historia?, Valor a ganar	 Métricas (8) Puntos, Esfuerzo, Numero de iteraciones, Valor, Ganado, Calidad, Progreso, Numero de usuarios		Retroalimentación (9) Nivel o Grado de Satisfacción de las personas, Nivel de Felicidad del usuario Impacto generado, Qué hacer para mejorar 
Importancia (10) Desde el punto de vista del negocio. Definido por Dueño de Producto.		Categoría (12) Categorización básica de la historia.	
Estimación (11) Valoración inicial del Equipo del esfuerzo necesario para implementar la historia, comparada con otras historias.			

- El acrónimo INVEST para describir las características de una buena historia:
 - Independiente: las historias pueden completarse en cualquier orden.
 - Negociable: los detalles de la historia son co-creados por los programadores y los clientes durante el desarrollo.
 - Valiosa: la funcionalidad es valiosa para los clientes o los usuarios del software.
 - Estimable: los programadores pueden encontrar una estimación razonable para construir la historia.
 - Pequeña: las historias deberían construirse en poco tiempo, generalmente alrededor de "días/persona". Se tienen que poder construir muchas historias en una iteración.
 - Testeable: se debe poder escribir pruebas que verifiquen que el software de la historia funcione adecuadamente.
- Una forma habitual de escribir las historias es "Como <rol>.... Quiero <característica>... Para <valor>". La parte del "Como..." se refiere a la persona que quiere la historia. La parte "Quiero..." describe la funcionalidad de la historia, y la parte "Para..." describe el motivo por el cual se pide esa funcionalidad.

- Técnicas de priorización de historias de usuario:
 - **MoSCoW** se basa únicamente en dividir las funcionalidades en cuatro grupos en función de su prioridad. Es conveniente que el número de funcionalidades esté equilibrado en cada grupo. MoSCoW es un pseudo-acrónimo formado por las cuatro categorías en las que se tienen que dividir todas las funcionalidades:
 - **Must Have (Imprescindibles):** son funcionalidades que deben ser incluidas antes de que el producto pueda ser puesto en producción. En el contexto de gestión *lean*, se denomina Mínimo Producto Viable (MVP por sus siglas en inglés) a aquel que contiene únicamente este conjunto de características. Sin alguna de ellas, el producto no tendría sentido.
 - **Should Have (Importantes):** son funcionalidades importantes y de gran valor para el usuario pero que no impiden poner el proyecto en marcha si no se tienen. Obviamente, son las siguientes en la lista de prioridad y deberían incluirse justo después de haber finalizado el Mínimo Producto Viable.
 - **Could Have o (Buenas):** son funcionalidades que sería deseable tener y podrían incluirse en caso de que hubiese recursos para ello. No obstante, en caso de que sea necesario, se podrían descartar.
 - **Won't Have o (Excluidas):** son funcionalidades que el cliente ha solicitado inicialmente pero que se descartan por falta de recursos (tiempo, dinero, etc). Como la priorización debe realizarse de forma iterativa a lo largo del proyecto, algunas de estas funcionalidades pueden considerarse en el futuro.

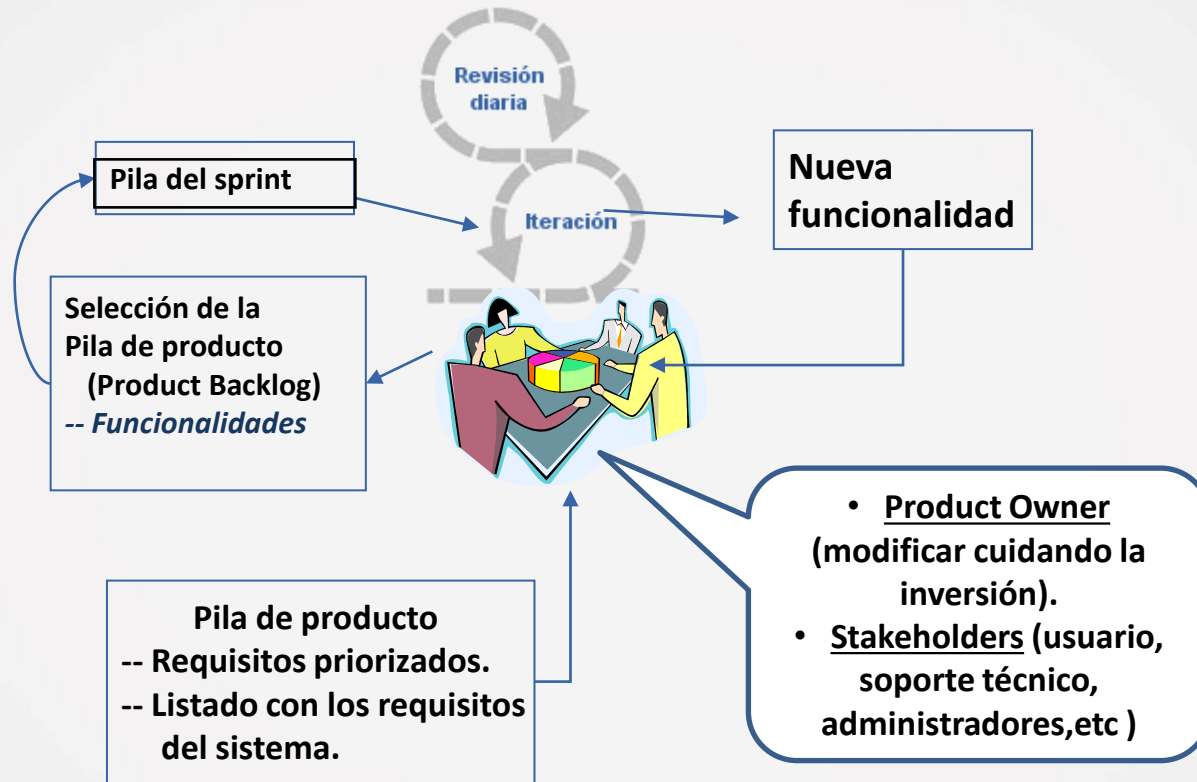
- **Theme Scoring** es una técnica que permite determinar la prioridad de las funcionalidades como una combinación de diferentes criterios, a los que se puede dar diferente importancia. A cada historia de usuario se le asigna un peso de 1 a 5 en cada una de las características. Para ello, por cada característica se elige una historia de usuario de referencia, que tenga un valor medio de 3. A las demás historias de usuario se les asigna el peso en cada característica en comparación con la historia de usuario de referencia. La estimación de una característica por comparación es siempre mucho más sencilla y rápida que la estimación de una medida absoluta.
- **WSJF** es una técnica que prioriza en función del valor, coste y riesgo equitativamente y de forma sencilla. Consiste en sumar el valor al usuario, el valor de tiempo (necesidad de un plan B) y la reducción del riesgo que aporta la historia y dividirlo por el tiempo que llevaría realizarla. Es un algoritmo optimizado económicamente para la priorización y programación del trabajo en un ambiente en el que el coste de la demora y la duración de los elementos de trabajo varían.
 - $$\text{WSJF} = (\text{Valor de Negocio/Usuario} + \text{Valor del Tiempo} + \text{Reducción de Riesgo} \& \text{Valor de Oportunidad}) / \text{Tamaño del trabajo}$$

- Donde los anteriores significan...
 - Valor de Negocio/Usuario: Es el valor que da la empresa o el usuario. Aquí también hay que incluir los costes de penalización si los hubiera así como los ingresos. Para darle un valor podemos escoger una escala de 1,2,3,5,8,13, siendo 1 lo más bajo y 10 lo más alto.
 - Valor de tiempo: En esta medida, lo que intentamos medir es la necesidad de un “plan B” por si las cosas salen mal, ya sea por la pérdida de contratos, clientes o ingresos. Podemos usar la misma escala de antes, en la que 1 sea una necesidad baja de este plan b, y el 13 una necesidad total el tenerlo.
 - Reducción de Riesgo & Valor de Oportunidad: En esta parte damos un valor a la necesidad de la empresa de reducir riesgos o el potencial de las nuevas oportunidades de negocio que se derivan del trabajo actual. Como antes, usaremos la misma escala.
 - Tamaño del trabajo: Estimación del tiempo necesario para poner una nueva característica o función del producto en producción. Podemos usar como antes la misma escala, o tomar una que sea en semanas, por ejemplo.
 - El resultado que nos de la fórmula con los datos anteriores nos prioriza las tareas, pasando a hacer primero las de un valor más alto.

Product Backlog

- Listado con los requisitos del sistema. Una lista de todos los trabajos deseados en el proyecto
 - Priorizadas
 - Documento "vivo"
 - Accesible a todos los roles
 - Todos pueden contribuir y aportar elementos
 - El propietario del producto es su responsable:
 - Representa los intereses del cliente dentro de la empresa.
 - Es el nexo entre el cliente y el equipo.
 - Tiene la visión global del producto buscado.
 - Es el encargado de armar y priorizar el Product Backlog (Lista priorizada de requerimientos).
- Cada tema tiene valor para el usuarios o el cliente (desde el punto de vista del negocio)
- Repriorizada al comienzo de cada Sprint
- Es dinámico.
- Mientras exista un producto, el Product Backlog también existe.





Ejemplos de Historias

***COMO** usuario **QUIERO** poder buscar transacciones **PARA** poder detectar gastos innecesarios en mi cuenta durante cierto periodo de tiempo*

Ejemplo Historia de Usuario 1 – Transacciones

- ROL: Usuario
- OBJETIVO: Poder buscar transacciones
- MOTIVACIÓN: Detectar gastos innecesarios en mi cuenta durante cierto tiempo

COMO usuario **QUIERO** poder acceder a la agenda de enfermería **PARA** utilizar sus funcionalidades

Ejemplo Historia Usuarios 2 – Agenda

- ROL: Usuario
- OBJETIVO: Poder acceder a la agenda
- MOTIVACIÓN: Utilizar funcionalidades

El título de una historia de usuario debe mapear una funcionalidad de nuestro producto o software.

Cuando escribimos historias de usuario, es una buena práctica el usar personajes en vez de roles, ya que dan incluso más contexto

.Por ejemplo:

*COMO **usuario** QUIERO ser capaz de mostrar la lista de visitas
PARA poder gestionar la llegada de pacientes y de salas de espera*

vs:

*COMO **enfermera** QUIERO ser capaz de mostrar la lista de visitas
PARA poder gestionar la llegada de pacientes y de salas de espera*

- Con el simple cambio de “**usuario**” a “**enfermera**” estamos dando mucho más contexto a nuestra historia de usuario y la hacemos mucho más legible para cualquier persona sin background técnico.
- Debemos recordar que el título de una historia de usuario debe mapear una sola funcionalidad de nuestro producto o software.

- La descripción de una historia de usuario ayuda a dar contexto a dicha historia. Ésta puede contener una pequeña explicación de los user flows, algunos casos de uso y casos extremos y, en general, cualquier explicación que ayude a comprender mejor el título.


La descripción no es una biblia, y recordemos que una imagen es mejor que mil palabras: elementos como links a páginas con diseños o capturas de pantalla que ayuden a clarificar lo que se está expresando son siempre bienvenidos

- Una buena descripción contendrá, a ser posible, estos elementos:
- **Contexto:** Proveer contexto de uso y aplicación para el desarrollador y el usuario, de cara a mejorar la comprensión del título.
- **Links a mockups o diseños:** Resultan de gran ayuda de cara a comprender cómo mapear la funcionalidad de la historia de usuario en la aplicación.
- **Limitaciones:** Elementos que todos los actores involucrados en la historia de usuario deben tener en cuenta (limitaciones de navegación y de diseño, requisitos, productos involucrados o reglas de negocio).

Criterios de aceptación en historias de usuarios de la metodología Ágil – Ejemplos


Junto con el título, la parte más importante de una historia de usuario. Los criterios de aceptación son una serie de preceptos que validan la implementación de nuestra historia de usuario. Incluso a día de hoy hay equipos que no utilizan criterios de aceptación – En mi opinión, esto es un grave error. Los criterios de aceptación son fundamentales, pues:

- Ayudan al desarrollador a implementar la tarea de forma correcta
- En TDD los criterios de aceptación se mapean de manera directa sobre tests funcionales
- Siguiendo TDD de forma estricta, la implementación debería empezar con estos criterios
- Ayudan a equipo de QA de cara a dar el OK a considerar finalizada la historia de usuario
- Los criterios de aceptación deben definir juntamente con el título la funcionalidad a implementar



El equipo de QA, idealmente, debería validar que se dan todos los criterios para dar el OK a subir a producción.

En general, los criterios de aceptación deben cumplir las características siguientes:

- **Atomicidad:** Deben tener 2 resultados únicos: ÉXITO o FRACASO (no hay “éxito parcial” – un criterio de aceptación debe dar siempre verde o rojo).
 - **No Ambiguos:** Deben ser interpretados de única manera por N personas (algo como “el formulario debe tener un color alegre” no es válido. . .aunque cosas peores se han visto)
 - **Verificables:** Deben estar escritos de forma que el cliente los pueda verificar rápidamente
 - **Completos:** El grupo de criterios de aceptación debe incluir todos los requisitos funcionales
- 

Ejemplo:

COMO usuario **QUIERO** acceder a la agenda de enfermería de forma segura

PARA usar sus funcionalidades

Criterios de aceptación:

- El **nombre de usuario DEBE tener valor**, en caso contrario se mostrará el mensaje de

error pertinente

- El **nombre de usuario DEBE tener forma de email**, en caso contrario se mostrará el mensaje de error pertinente

- La **contraseña DEBE tener valor**, en caso contrario se mostrará el mensaje de error pertinente

- El **nombre de usuario DEBE existir en la base de datos** en caso contrario se mostrará el mensaje de error pertinente

- La **contraseña DEBE coincidir con la que el nombre de usuario tiene asociada en la base de datos**, en caso contrario se mostrará el mensaje de error pertinente

Si el usuario y la contraseña son correctos, el usuario podrá acceder a la aplicación. Muchas veces estos criterios de aceptación pueden parecer triviales o innecesarios, pero ser riguroso con ellos mientras redactamos las historias de usuario es la mejor manera de desarrollar la funcionalidad

Prácticas para la gestión del producto

- ***Técnica para entender y planear una entrega incremental de MVPs.*** La técnica organiza ideas y recursos en un modelo que busca entender la finalidad principal del producto, considerando las jornadas de los usuarios para realizar las entregas incrementales de productos viables. Como un libro de recetas, con una secuencia de actividades, rápidas y eficaces, la técnica va a permitir que el equipo:
 - Describa la visión del producto. El producto Es - No es - Hace - No hace
 - Priorize los objetivos del producto. Entendiendo los trade-offs.
 - Describa los principales usuarios, sus perfiles y sus necesidades
 - Entienda las principales funcionalidades
 - Comprenda los niveles de incertidumbre, esfuerzo y valor de negocio por funcionalidad
 - Describa las jornadas más importantes de los usuarios
 - Cree un plan de entrega incremental del producto, impulsado por el concepto de MVP
 - Estime el esfuerzo por muestreo
 - Calcule los costos y especifique fechas en el cronograma de entrega



Caso practico

- Un cliente se pone en contacto con una empresa que fabrica robots.
- El cliente les realiza el pedido.

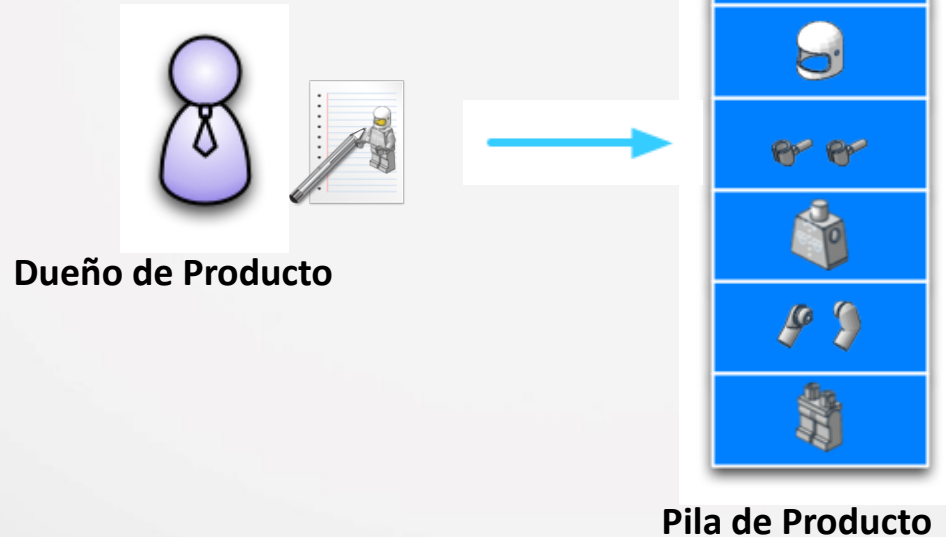
Quiero un robot que me sirva de escolta



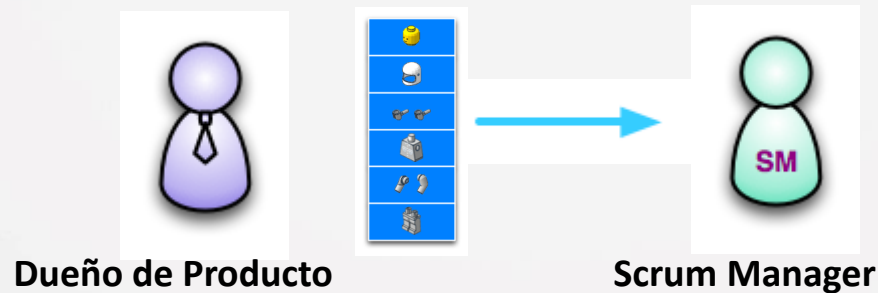
- El Cliente se reúne con el Dueño de producto, que toma nota de lo que tiene en su cabeza.



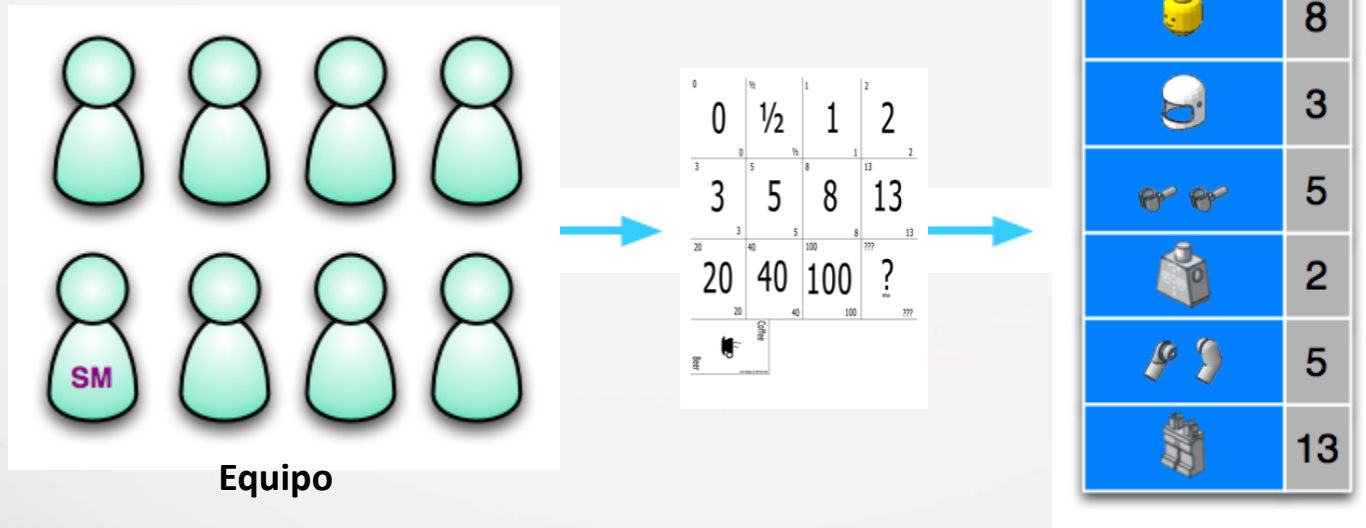
- El Duelo de Producto divide el proyecto en historias que son las que componen la pila de producto.



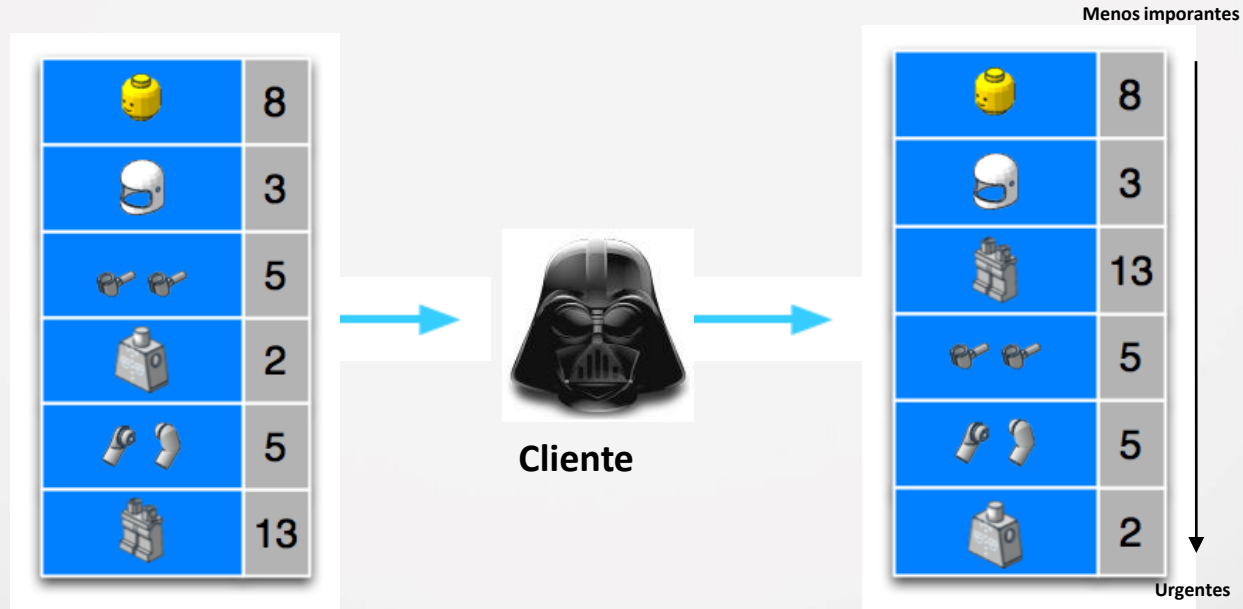
- El Scrum Master es un miembro del equipo que tiene el papel de comunicar y gestionar las necesidades del Dueño de Producto y la pila de Sprint.
- El Dueño de Producto le entrega la pila de producto para que estimen el coste de creación del producto.



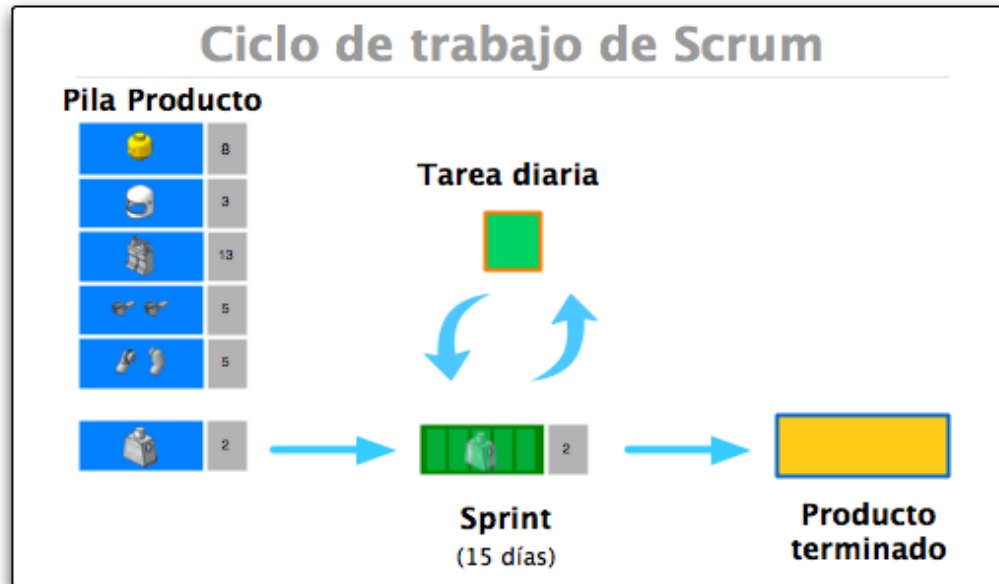
- El equipo se reúne para estimar el coste de cada historia de la pila de producto.
- En este caso utilizan Planning Poker.



- El cliente, una vez aprobado el presupuesto, reordena la pila de producto para que el equipo vaya trabajando según la prioridad del cliente.



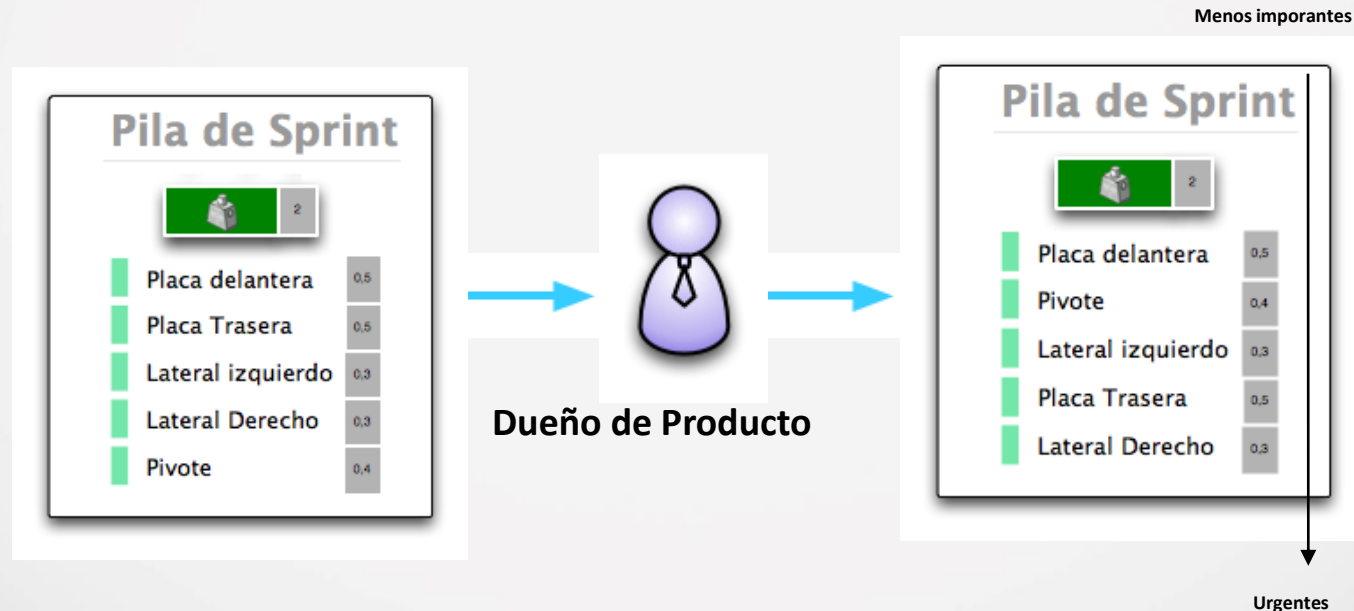
- El equipo comienza su trabajo desglosando la primera historia de la pila de producto, la cual subdividen en tareas menores para crear la pila de sprint.



- La pila de sprint tiene como utilidad fraccionar el trabajo de un periodo de 15 días en tareas mas pequeñas, que tarden como mucho dos días.



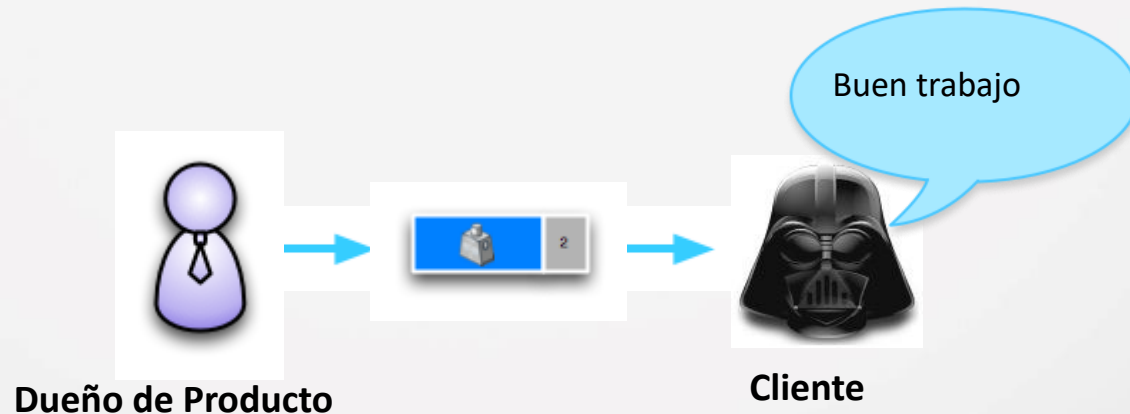
- Estas tareas se colocan en una pila, la cual prioriza el Dueño de Producto, que ha consultado con el cliente, antes de comenzar el sprint.



- El equipo comienza el sprint tomando las tareas priorizadas.
- Una vez concluida una se toma la siguiente de la lista.
- Se convoca todos los días una reunión del equipo donde se cuenta las tareas realizadas el día anterior y cuales se van a realizar ese día.



- Una vez finalizado el sprint, el Dueño de Producto le muestra al cliente el resultado del trabajo realizado.
- El cliente ya tiene el primer contacto con su encargo y además puede volver a priorizar la pila de producto antes de que comience otro sprint.



- El equipo de trabajo celebra su buen hacer con una reunión de retrospectiva, donde se analiza lo ocurrido durante el sprint.

