



Historias de Usuario

Ingeniería de requisitos ágil

v. 2.0

Historias de Usuario

Ingeniería de requisitos ágil

Versión. 2.0 – Abril 2018

Imagen y diseño de cubierta: Scrum Manager.

© 2018 Alexander Menzinsky, Gertrudis López y Juan Palacio

© Producción: Scrum Manager®

Información de derechos y licencia de uso:



<http://www.safecreative.org/work/1804036442990>






Scrum Manager® es marca registrada, propiedad de Iubaris Info 4 Media S.L.

Contenido





<i>Contenido</i>	5
<i>Información, recursos y comunidad profesional</i>	6
<i>Mejora continua y control de calidad Scrum Manager</i>	7
<i>Ingeniería de requisitos ágil</i>	9
Historias de Usuario	9
De Temas y Epics hasta Tareas	11
<i>Información en una historia de usuario</i>	13
Informaciones necesarias y opcionales	13
Descripción	16
Valor de negocio	17
Estimación	18
Prioridad	20
Criterios de validación	22
<i>Calidad en las historias de usuario</i>	24
<i>División de historias de usuario</i>	26
<i>Comparativa con otras formas de toma de requisitos</i>	28
Historias de usuario versus casos de uso	28
Historias de usuario versus requisitos funcionales	29
<i>Historias técnicas</i>	30
<i>User Story Mapping</i>	31
<i>Apéndice: Historias de usuario en otros ámbitos</i>	34
<i>Bibliografía</i>	35
<i>Tabla de ilustraciones</i>	37
<i>Índice</i>	38

Información, recursos y comunidad profesional



RECURSOS

	Última versión de este libro. (http://www.scrummanager.net/bok/)
	Open Knowledge Scrum. (http://www.scrummanager.net/oks/)
	Blog (http://www.scrummanager.net/blog/)
	Acerca de la certificación Scrum Manager®. (https://scrummanager.com/index.php/es/certificacion)
	Preguntas frecuentes sobre cursos y exámenes Scrum Manager® (https://scrummanager.com/index.php/es/faq)

REDES SOCIALES

	Twitter. (https://twitter.com/scrummanager)
	Facebook. (https://www.facebook.com/Scrum-Manager-144889095527292/)
	Google+ (https://google.com/+ScrummanagerNet/)
	Pinterest (https://es.pinterest.com/scrummanager/pins/)

REDES PROFESIONALES

	Grupo profesional en LinkedIn (http://www.linkedin.com/e/gis/855957)
	Comunidades Google + (https://plus.google.com/communities/116174698722878580028)

<http://www.scrummanager.com>

Mejora continua y control de calidad Scrum Manager

Gracias por elegir los servicios de formación de Scrum Manager

Su valoración es el criterio del control de calidad de Scrum Manager, y decide la validez o no de los servicios de formación, y en su caso la continuidad de los cursos, centros y profesores.

Si ha participado en una actividad de formación auditada por Scrum Manager, le rogamos y agradecemos que valore la calidad del material, profesor, temario, etc. así como tus comentarios y sugerencias.

Scrum Manager anonimiza la información recibida, de forma que comparte con los profesores, y centros autorizados las valoraciones y aspectos de mejora, pero en ningún caso los nombres de los alumnos que las han realizado.

Puede realizar la valoración en la página accesible a miembros de Scrum Manager:

<https://scrummanager.com/index.php/es/qa>.

Ingeniería de requisitos ágil

Historias de Usuario

Las **historias de usuario** son utilizadas en los métodos ágiles para la especificación de requisitos, **son una descripción breve de una funcionalidad software tal y como la percibe el usuario** (Mike Cohn, 2004).

Describen funcionalidades que dan solución a necesidades o problemas del cliente o del usuario, representan los "**qués**" a construir y se escriben en forma de **historia** con una o dos frases utilizando el **lenguaje común del usuario**.

Estas son una forma ágil de administrar los requisitos de los usuarios sin tener que elaborar gran cantidad de documentos formales y sin requerir de mucho tiempo para administrarlos. Las historias de usuario forman parte de la fórmula de captura de funcionalidades definida en 2001 por Ron Jeffries de las **tres C's**:

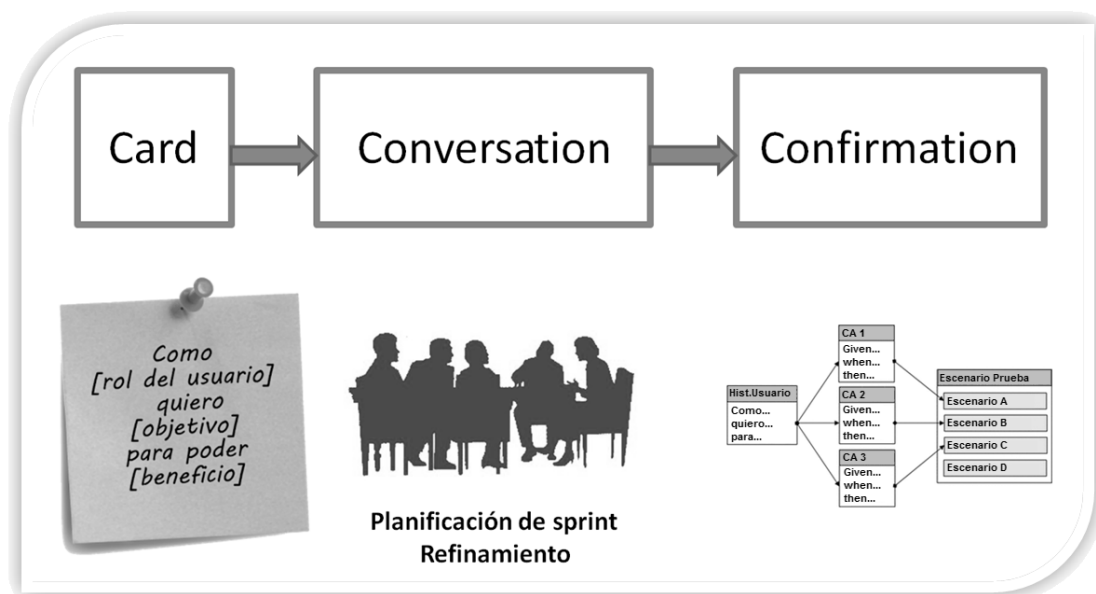


Ilustración 1: Gráfico con las tres fases de la fórmula de las tres C's

Cada historia de usuario debe ser limitada, esta debería poderse memorizar fácilmente y escribir sobre una tarjeta o post-it (**card**), ya que son una promesa de una conversación posterior. Poco antes de ser implementadas, en la reunión de refinamiento o en la de planificación de sprint, estas van acompañadas junto con los criterios de validación asociados de **conversaciones** entre el equipo de desarrollo y el propietario del producto. Como los cambios son bienvenidos en agilidad, no vale la pena profundizar antes, ya que en el momento de la implementación estas pueden haber cambiado desde que fueron escritas. Los criterios de validación, a veces transformados en escenarios de pruebas por el equipo de desarrollo, permiten al propietario del producto o usuario de negocio **confirmar** que el equipo ha entendido y recogido correctamente los requisitos.

Ventajas que aportan las historias de usuario:

- Al ser muy cortas, estas representan requisitos del modelo de negocio que pueden implementarse rápidamente (días o semanas).
- Necesitan poco mantenimiento.
- Mantienen una relación cercana con el cliente.
- Permiten dividir los proyectos en pequeñas entregas.
- Permiten estimar fácilmente el esfuerzo de desarrollo.
- Son ideales para proyectos con requisitos volátiles o no muy claros.

El origen de las historias de usuario viene de **XP “eXtremeProgramming”** o programación extrema, donde las historias de usuario deben ser escritas por los clientes. Esta metodología fue creada por Kent Beck y descrita por primera vez en 1999 en su libro *eXtreme Programming Explained*.

Las **historias de usuario** se aplican en la mayoría de los métodos ágiles, siendo así una herramienta muy importante también en **scrum**.

De Temas y Epics hasta Tareas

Entre los "qués" a construir también se encuentran otro tipo de elementos conocidos como epics y temas.

Se denomina **epic** a una superhistoria **de usuario que se distingue por su gran tamaño**, a diferencia de las historias de usuario, que tienen baja granularidad, los epics tienen una alta granularidad. Es una etiqueta que aplicamos a una historia grande, cuyo esfuerzo es demasiado grande para completarla de una sola vez o en un solo sprint. Los epics suelen tener un flujo asociado por el cual se puede dividir en historias de usuario, en otras palabras, las historias de usuario resultantes de la descomposición de un epic están íntimamente relacionadas entre sí. A medida que aumenta su prioridad y se acerca al momento de su implementación, el equipo la descompone en historias de usuario con un tamaño más adecuado para ser gestionada con los principios y técnicas ágiles: estimación y seguimiento cercano (normalmente diario).

A un nivel por encima de epics e historias de usuario se encuentran los **temas**, estos representan a una **colección de epics y/o historias de usuario relacionados** para describir un sistema o subsistema en su totalidad, más que una funcionalidad describen un elemento que forma parte de la **visión del producto**. Por ejemplo en un sistema de software para gestión contable, el conjunto de epics: "Altas, bajas y mantenimiento de clientes", "Facturaciones puntuales y recurrentes", "Consultas de navegación y acciones de fidelización", "Pedidos", "Devoluciones" se podrían denominar como el tema de la gestión de clientes.

En la jerarquía de los requisitos ágiles y por debajo de las historias de usuario se encuentran los "**cómos**" construir, las **tareas**. Estas son resultado de la **descomposición por parte del equipo de las historias de usuario en unidades de trabajo** adecuadas para gestionar y seguir el avance de su ejecución.

Gráfico con los cuatro niveles de tamaño con que trata los requisitos la gestión ágil:

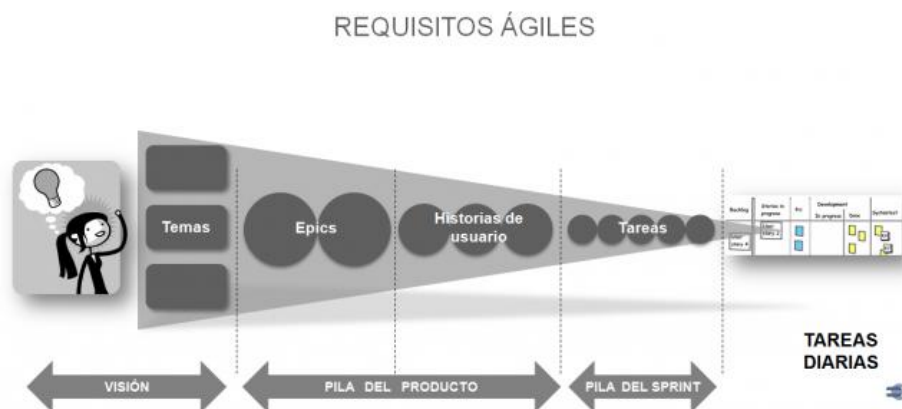


Ilustración 2: Gráfico con los cuatro niveles de tamaño con que trata los requisitos la gestión ágil

En **scrum**, y en los métodos ágiles en general, una **pila de producto** puede contener tanto **historias de usuario** como **temas** y **epics**.

La granularidad es el nivel de detalle de cada elemento de la pila de producto, y este debe de ir en relación a la posición dentro de la misma. Recordemos que la posición va en relación a la prioridad. Al final de la lista, donde está lo menos prioritario, están los epics, historias de usuario de gran tamaño, y los temas que describen grandes cosas, de los que cada uno se descompondrá en elementos más pequeños a medida que avanza por la pila.

Las historias susceptibles de entrar en el próximo sprint son las que más detalle deben de tener para que en la reunión de planificación de sprint el equipo tenga toda la información necesaria para desglosarlas en tareas de la pila de sprint, por tanto las historias de usuario son los elementos que deben de encabezar la lista.

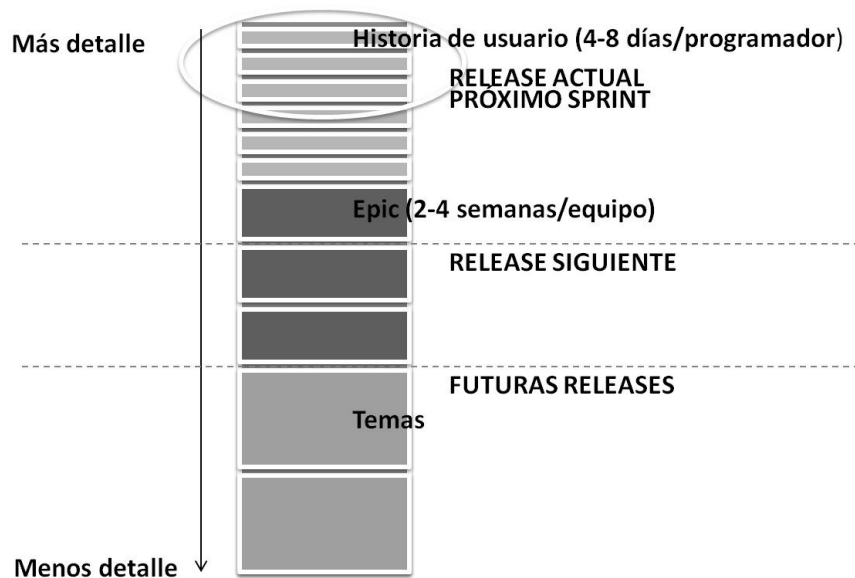


Ilustración 3: Granularidad de la pila de producto

Información en una historia de usuario

Informaciones necesarias y opcionales

Para decidir qué **información incluir en una historia de usuario** es preferible no adoptar formatos rígidos. Los resultados de scrum y agilidad no dependen de las formas, sino de la institucionalización de sus principios y la implementación adecuada a las características de la empresa y del proyecto. Por tanto, aparte de **3 campos que se consideran necesarios**, se puede incluir cualquier campo que proporcione información útil para el proyecto, recordemos que el foco de las historias de usuario y sus informaciones es la construcción de un entendimiento compartido.

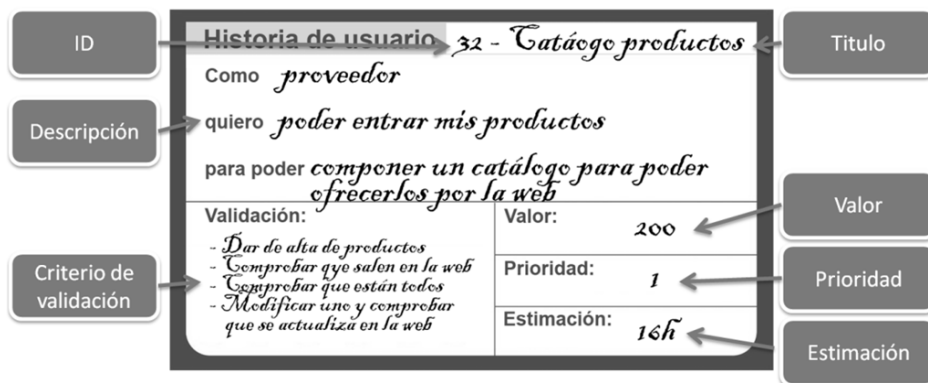


Ilustración 4: Ejemplo de una tarjeta de historia de usuario

Los campos que se consideran **más necesarios** para describir de una manera adecuada las historias de usuario son:

- **Descripción:** descripción sintetizada de la historia de usuario. El estilo puede ser libre, según mejor nos funcione, debe responder a tres preguntas: ¿Quién se beneficia? ¿Qué se quiere? y ¿Cuál es el beneficio? Mike Cohn recomienda seguir el siguiente patrón que garantiza que la funcionalidad está descrita a un alto nivel y de una manera no demasiado extensa:

Como [rol del usuario], quiero [objetivo], para poder [beneficio]

- **Estimación:** estimación del **esfuerzo necesario** en tiempo ideal de implementación de la historia de usuario. Según convenga al equipo también se puede utilizar unidades de desarrollo, conocidas como **puntos de historia** (estas unidades representan el tiempo teórico de desarrollo/persona que se estipule al comienzo del proyecto).
- **Prioridad:** **sistema de priorización** que nos permite determinar el orden en el que las historias de usuario deben de ser implementadas.

Dependiendo del tipo de proyecto, el funcionamiento del equipo y la organización, pueden ser aconsejables otros campos como:

- **ID:** identificador de la historia de usuario, **único** para la funcionalidad o trabajo.

- **Título:** título descriptivo de la historia de usuario.
- **Criterio de validación:** **pruebas de aceptación** consensuadas con el cliente o usuario. Estas son los criterios a veces transformados en pruebas que el código debe superar para dar como finalizada la implementación de la historia de usuario.
- **Valor:** valor (normalmente numérico) que aporta la historia de usuario al cliente o usuario. El objetivo del equipo es **maximizar el valor y la satisfacción percibida por el cliente** en cada iteración. Este campo servirá junto con la estimación para determinar la prioridad con el que las historias de usuario deben de ser implementadas.
- **Dependencias:** una historia de usuario no debería ser dependiente de otra historia, pero en ocasiones es necesario mantener la relación. En este campo se indicarían los identificadores de otras historias de las que depende.
- **Persona asignada:** en casos en que queramos sugerir la persona que pueda implementar la historia de usuario. Recordar que en **scrum** es en último término el equipo autogestionado quién distribuye y por tanto asigna las tareas.
- **Criterio de finalización:** la definición de **finalizada/hecho** incluye los criterios o actividades necesarias para dar por terminada una historia de usuario (desarrollada, probada, documentada...), que son las convenidas por el equipo y el propietario del producto.
- **Sprint:** puede ser útil para organización del propietario del producto incluir el **número de sprint** en el que previsiblemente se vaya a realizar la historia.
- **Riesgo:** riesgo **técnico o funcional** asociado a la implementación de la historia de usuario.
- **Módulo:** módulo del **sistema o producto** al que pertenece.
- **Observaciones:** para **enriquecer o aclarar** la información o cualquier uso que pueda ser útil.

Historia de Usuario	
Número: 1	Usuario: Cliente
Nombre historia: Cambiar dirección de envío	
Prioridad en negocio: Alta	Riesgo en desarrollo: Bajo
Puntos estimados: 2	Iteración asignada: 1
Programador responsable: José Pérez	
Descripción: Como cliente quiero cambiar la dirección de envío de un pedido para que me pueda llegar a casa o a la oficina	
Validación: El cliente puede cambiar la dirección de entrega de cualquiera de los pedidos que tiene pendiente de envío	

Ilustración 5: Ejemplo una historia de usuario

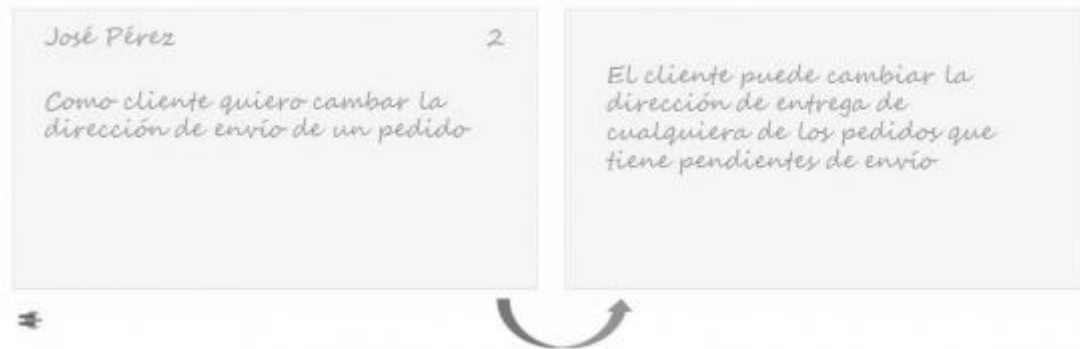


Ilustración 6: Ejemplo una historia de usuario

Mike Cohn comenta que **si bien las historias de usuario son lo suficientemente flexibles como para describir la funcionalidad de la mayoría de los sistemas, no son apropiadas para todo. Si por cualquier razón, se necesita expresar alguna necesidad de una manera diferente a una historia de usuario, recomienda que se haga.** Por ejemplo, la maquetación de pantallas se suele describir con pantallazos, por tanto esta es la mejor manera de transmitir el diseño que queramos darle a una aplicación.

Descripción

El pensamiento de las personas se estructura siguiendo una narrativa, una historia, así es como entendemos el mundo. **Estamos capacitados para comprender personajes, deseos y motivaciones**, por tanto la forma en que más facilidad tenemos para adquirir y retener conocimiento es **a través de las historias**. Nos metemos dentro de los protagonistas de forma que vivimos como tales la historia que nos cuentan, empatizamos, y a todos los niveles que toman decisiones, hasta a nivel de desarrollador, tomamos decisiones más acertadas.

Una técnica que viene de la **Experiencia de Usuario (UX)** y nos acerca a los usuarios, es la **User-Persona**, una técnica que permite ponernos en lugar de los usuarios y vivir su historia. La técnica se basa en la idea de conocer a los usuarios del producto que vamos a construir mediante arquetipos descritos como personajes de ficción, como por ejemplo una representación realista de la audiencia clave de nuestra web o aplicación. Estas User-Personas serán los **roles de usuario** en el patrón para las historias de usuario:

Como [rol del usuario], quiero [objetivo], para poder [beneficio]

- **Como [rol del usuario]:** ¿para quién estamos construyendo esto? No se trata de un cargo o un rol profesional, se trata de la persona que hay detrás de la necesidad descrita, persona que describimos con la User-Persona y que permite al equipo tener un entendimiento compartido de quién es, entender como trabaja, como piensa y siente, en definitiva tener empatía por ella.
- **quiero [objetivo]:** Aquí estamos describiendo su intención, no la funcionalidad que usará. ¿Qué es lo que realmente está tratando de lograr?
- **para poder [beneficio]:** ¿cómo encaja su necesidad inmediata en lo que le rodea? ¿cuál es el beneficio general que está tratando de lograr? ¿cuál es el gran problema que necesita resolver? Se trata de un beneficio que va relacionado con la visión del producto.

Con esta técnica nos centramos en los usuarios y mantenemos nuestras conversaciones centradas en estos. Dándoles un nombre y asignándoles una personalidad **empatizamos** con ellos y con el grupo de usuarios al que representan. Si los usuarios objetivos de nuestra web son un determinado tipo de persona hacer su User-Persona y referirnos a ella con su mote, nos permite ponernos en su lugar e imaginar cómo interactuarán con el sistema, ayudando al equipo a entenderlos, a tomar decisiones más acertadas y ofrecer soluciones más adecuadas a sus necesidades.

La técnica representa a los usuarios en una ficha tipo tríptico, por ejemplo:

- **Nombre y apodo:** una manera rápida de identificar y reconocer al personaje
- **Datos demográficos:** ayudan a clasificar al personaje dentro de un segmento de mercado
- **Descripción:** se trata de un párrafo breve que nos ayude a conocer al personaje
- **Objetivos:** buscamos aquellas necesidades del usuario que distinguen su comportamiento del resto


 <p style="margin-top: 5px;">Roberto García "El Scrumita"</p>	<p>Datos demográficos: padre de familia de 40 años que utiliza la tecnología de forma habitual.</p> <p>Descripción: afable e inquieto, gran lector de libros y blogs sobre Agilidad y Scrum. Es miembro de varios grupos de interés del mundo ágil.</p> <p>Objetivos: como Scrum Master estar al día y formado en prácticas ágiles a través de cursos y comunidades de práctica.</p>
------------------------------------------------------------------------------------------------------------------------------------------------------	-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Ilustración 7: Ejemplo de User-Persona

Valor de negocio

La información "valor" representa el **valor de negocio que aporta una historia de usuario** una vez realizada, es una información más profunda que la razón de la historia misma ya que está íntimamente ligada a la lógica de negocio. ¿Pero qué mide esa información? Podríamos decir, que representa "**cuánto dinero está dispuesto a pagar nuestro cliente por esa funcionalidad**". Cuanto más esté dispuesto a pagar, más valor de negocio tiene. Tal como funcionan scrum y agilidad entregar valor significa focalizarnos en **resolver los problemas de alguien o en dar beneficios a alguien**.

El valor de negocio es muy importante para priorizar correctamente, focaliza al equipo de desarrollo en todo momento en aquello que es prioritario construir y maximiza el valor y la satisfacción percibida por el cliente en cada sprint, aunque no es lo único, hay dependencias y otros factores que pueden determinar la prioridad, es por ello por lo que valor y prioridad son dos informaciones independientes.

El valor **se mide con una escala arbitraria** (normalmente valores numéricos) que aporta la historia de usuario o epic al cliente o usuario. Ha de ser una escala con la que el propietario del producto y los usuarios se sientan cómodos, por ejemplo la serie de Fibonacci, o series de números naturales como del 1 al 10, del 1 al 100 o del 1 a 1.000.000.

Una forma muy estimulante es repartir billetes del Monopolio por el valor total del proyecto a cada uno de los usuarios de negocio involucrados, y que estos repartan el dinero en función del valor que le atribuyen a cada historia de usuario. Es un ejercicio que les acerca a la realidad, no les toca directamente el bolsillo pero lo simula muy bien. Finalmente el valor es la suma de los billetes de todos, opcionalmente se puede dividir el resultado por 100 o 1000 para que sea un número más manejable.

Benoît Pointet y Thomas Botton proponen estimar el valor de la misma manera que el equipo estima el esfuerzo, mediante cartas de **planning poker con la serie de Fibonacci**. El valor, igual que el esfuerzo, es sumativo y relativo entre historias. La idea es que el propietario del producto y los expertos de negocio jueguen a planning poker, y de la misma manera que se crea una base de conocimiento en las reuniones del equipo, las reuniones para la estimación de valor crearán una base de conocimiento compartida para usuarios y gente de negocio. Esta propuesta tiene dos efectos colaterales muy interesantes: trae respeto y comprensión del equipo hacia el propietario del producto, y, este último comprenderá más fácilmente porqué el equipo necesita en ocasiones reestimar ciertas historias de usuario.

Estimación

Como hemos visto la "**estimación**" es una información necesaria en las historias de usuario:

- Ayuda en la priorización de las mismas por parte del propietario del producto
- Permite al equipo hacer el corte en la pila de producto de qué historias caben en el sprint

La estimación ágil se basa en la **estimación relativa**, y una buena serie para la estimación de historias de usuario es la **serie de Fibonacci**. Es un hecho que para historias de usuario pequeñas la incertidumbre en la estimación es mucho menor que para las historias grandes, y mientras **el tamaño de las historias crece linealmente la incertidumbre crece exponencialmente**.

La serie de Fibonacci está compuesta por números que son la suma de los dos anteriores:

1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, ...

La distancia entre los números de la serie refleja el hecho de que la incertidumbre inherente a la estimación crece proporcionalmente con el tamaño de la historia de usuario. Así, las diferencias entre 1, 2 y 3 puntos de historia son probablemente mejor comprendidas que las diferencias entre 13, 21 y 42.

También hay razones que derivan de la naturaleza humana para utilizar esta serie. Para las personas nos es fácil estimar de forma relativa, nos es fácil determinar cuánto menor o mayor es algo respecto a otra cosa. Pero tenemos la tendencia de pensar en múltiplos de dos, por ejemplo, una historia de usuario la comparamos con una de 2 puntos, vemos que es mayor y automáticamente pensamos en 4, 8 o 16. La serie de Fibonacci nos obliga a romper con ello y a buscar la estimación más adecuada.

Las estimaciones que se basan en la serie de Fibonacci usan barajas de cartas que pueden tener ciertas variaciones. Se incluye el 0 para historias que requieren un esfuerzo casi nulo, 1/2 para historias muy pequeñas. Algunas barajas sustituyen el 21 por un 20, ya que decir que una historia tiene un tamaño de 21 da una falsa sensación de precisión. A partir del 21 las barajas suelen pasar al infinito (∞) y, si siguen, incluyen valores redondos como 40 y 100.



Ilustración 8: Cartas planning poker con la serie de Fibonacci

Recordemos que el objetivo del **planning poker** no es obtener estimaciones que puedan ser confirmadas más adelante, sino ofrecer un proceso de estimación o aproximación que sea rápido y ágil para obtener valores útiles para que el equipo pueda estimar la pila de producto o planificar el sprint. Más importante aún es crear un entendimiento compartido del elemento estimado y generar una base de conocimiento común: si

los participantes estiman con valores muy dispares es que alguno sabe algo que otros no saben. Recordemos que en los equipos hay especialistas de diferentes áreas, hay optimistas y pesimistas y eso hace que algunos vean soluciones que otros no ven. La discusión alrededor de las soluciones y la disparidad de la estimación alineará el conocimiento de todo el equipo.

Para la estimación de elementos grandes como pueden ser los **epics** y son los **temas** la serie de Fibonacci no es adecuada, sus números darían una idea de precisión inexistente. Una solución para los elementos grandes y lejanos de la pila de producto es la técnica que está basada en **tallas de camisa**:

XS, S, M, L y XL

Realmente los elementos más grandes de la pila de producto son muy vagos para estimar y el utilizar las tallas de camisa nos da algo como una intuición más que una estimación, con un gran grado de incertidumbre amplio acorde a epics y temas. Esta técnica tiene la ventaja de ser **puramente relativa**, no es posible traducir sus valores a tiempo, a jornadas u horas.

Con ambas técnicas podemos estimar la pila de producto al completo de forma coherente con agilidad. Los elementos de la parte prioritaria de la pila están detallados, incluida la estimación, y los elementos más lejanos poco granulados, con una estimación más vaga y amplia.



Ilustración 9: Cartas de estimación con tallas de camisa

El propietario del producto se basa en el resultado de ambas técnica para priorizar la pila de producto, tiene valores más detallados de lo que tiene frente a sí y necesita estar más perfilado, y tiene valores con detalle suficiente para elementos los lejanos y poco prioritarios.

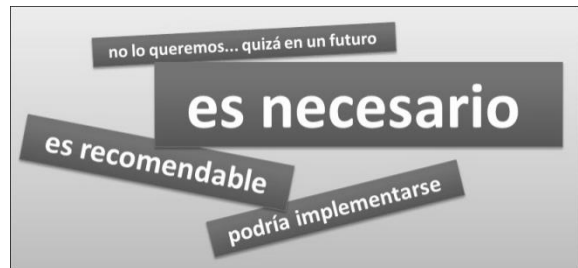
Prioridad

Aunque todas las historias de usuario puedan ser importantes, para poder **focalizarnos en el trabajo de forma eficiente**, es necesario destacar aquellas que den mayor valor al sistema, por tanto, las historias de usuario deben de estar priorizadas. Estas deben de tener **asignadas un valor** que intervenga en el **sistema de priorización**, un valor asignado por el propietario del producto y se basará básicamente en las siguientes variables:

- Beneficios de implementar una funcionalidad.
- Pérdida o coste que demande posponer la implementación de una funcionalidad.
- Riesgos de implementarla.
- Coherencia con los intereses del negocio.
- Valor diferencial con respecto a productos de la competencia.

Uno de los aspectos a tener en cuenta es que la definición de "valor" puede variar para cada uno de nuestros clientes. Más allá de un sistema de clasificación de tipo prioridad alta, media o baja es muy recomendable utilizar algún tipo de escala cualitativa, una que tenga un **significado intrínseco**. Este es el caso de la **técnica MoSCoW**, en la que el usuario responsable de asignar la prioridad es consciente del efecto real que producirá su elección. Esta técnica fue definida por primera vez en el año 2004 por Dai Clegg de Oracle UK Consulting en el libro *Case Method Fast-Track: A RAD Approach*. Su finalidad es obtener un entendimiento común entre cliente y el equipo del proyecto, en concreto sobre la importancia de cada historia de usuario. La clasificación es la siguiente:

- **M - MUST HAVE (es necesario)**: Se debe tener la funcionalidad implementada en la solución, sino esta fallará o la solución no puede ser considerada un éxito.
- **S - SHOULD HAVE (es recomendable)**: Se debería tener la funcionalidad implementada en la solución ya que es una funcionalidad de alta prioridad. La solución es prescindible, no fallará si no existe pero debería de haber causas justificadas para no implementarla.
- **C - COULD HAVE (podría implementarse)**: Es deseable, por tanto sería conveniente tener esta funcionalidad implementada en la solución, dependerá de las posibilidades de los tiempos y el presupuesto del proyecto.
- **W - WON'T HAVE (no lo queremos... quizá en un futuro)**: Se trata de una funcionalidad de muy baja prioridad o descartada en ese momento, pero que en futuro pueda ser relevante. Posteriormente, cuando cobre importancia, puede pasar a alguno de los estados anteriores.



Es importante distinguir entre prioridad y valor para el cliente. Puede ser que una historia de usuario no tenga ningún valor para el cliente o usuario, pero que esta sea absolutamente necesaria, por tanto de alta prioridad. Por ejemplo la infraestructura necesaria para la implementación de un software, no aporta valor al cliente en sí, pero sin ella no se puede desarrollar ni ejecutar la solución desarrollada.

Otra herramienta que ayuda al propietario del producto a priorizar la pila de producto de forma adecuada es el cálculo del **ROI (Return of Investment) o retorno de la inversión**:

$$\text{ROI} = \text{Valor de negocio} / \text{Tamaño}$$

- **Valor de negocio**: el valor relativo del elemento para negocio o el cliente.
- **Tamaño**: estimación en puntos de historia del esfuerzo necesario teniendo en cuenta complejidad y tamaño.

El **ROI aplicado de forma descendente** resulta en una excelente guía para priorizar la pila, cuanto más alto el retorno de inversión tanto más alta es la prioridad del elemento.

Una técnica de priorización lean muy recomendable por su completitud es **WSJF (Weighted Shortest Job First)** de Don Reinersten descrita en 2009 en su libro *The Principles of Product Development Flow: Second Generation Lean Product Development*. Esta técnica está basada en la economía de flujo de desarrollo de productos que calcula dividiendo el coste de demora por la duración.

Lo interesante del **coste de demora** es que incluso las historias técnicas, que intrínsecamente no tienen valor de negocio, lo tienen. El coste de demora representa el valor que no obtenemos por no construir determinada historia, y en caso de una historia técnica se puede no obtener el valor de negocio de las historias de usuario que consumen esta última.

Dado que la duración es imposible de estimar el coste de demora utiliza el tamaño del elemento como sustituto. Al coste de demora se considera que contribuyen los siguientes tres elementos:

$$\text{Coste de demora} = \text{Valor de negocio} + \text{Criticidad en el tiempo} + \text{Reducción del riesgo y valor de oportunidad}$$

- **Valor de negocio**
- **Criticidad en el tiempo:** Esta medida se ocupa de la "necesidad" de la entrega del elemento en una escala de tiempo, viene asociado con la caída de valor con el tiempo. A más "necesidad" de entrega más alto será el valor.
- **Reducción del riesgo y valor de oportunidad:** Esta medida da un valor relativo a la eliminación de uno o varios riesgos o, un valor por el potencial de nuevas oportunidades de negocio que puede aportar el elemento.

$$\text{WSJF} = \text{Coste de demora} / \text{Tamaño}$$

La técnica de priorización WSJF consiste en aplicar los siguientes pasos a la pila de producto:

- La técnica estima las funcionalidades / historias de usuario / epics entre sí de forma relativa
- La escala de unidades es la serie de Fibonacci: 1, 2, 3, 5, 8, 13, 21
- Se rellena la lista columna por columna
- Se pone un 1 en el elemento con el valor más pequeño, tiene que haber un 1 en cada columna
- Se rellena el resto de elementos de la columna con estimación relativa respecto a la del 1
- Hecha la última columna se calcula el WSJF y se ordena de mayor a menor
- El elemento con mayor prioridad es el de WSJF más alto

Funcionalidad / Historia de Usuario / Epic	Valor de negocio	Criticidad en el tiempo	Reducción del Riesgo y Valor de Oportunidad	Tamaño	WSJF
Como miembro del departamento de administración quiero una pantalla para poder gestionar el catálogo de las fichas los productos de la empresa y que esté enlazada con la foto correspondiente en el Google+	1	5	1	1	7
Como comercial quiero un catálogo con un sistema de pedidos online para formar a los clientes y hacer los pedidos directamente para los clientes que se precise	5	13	8	8	3,25
Como miembro del departamento de administración quiero una pantalla para poder gestionar los datos de los clientes	8	2	3	5	2,6
Como gerente quiero tener información mensual de productos vendidos y facturas emitidas para tener reportes con información fiable y a demanda	3	1	1	3	1,67

Ilustración 10: Ejemplo de priorización WSJF

Criterios de validación

Después de 50 años de historia de ingeniería de software hemos llegado a la conclusión de que los **criterios de validación**, que a veces se traducen en pruebas, **son un excelente lenguaje para detallar requerimientos funcionales**, y es por ello que toman una gran importancia en las historias de usuario.

Para medir la calidad de un criterio de aceptación se utiliza el método **SMART** en el que se han de cumplir en lo máximo posible los siguientes criterios:

- **S** - Specific (Específicos)
- **M** - Measurable (Medibles)
- **A** - Achievable (Alcanzables)
- **R** - Relevant (Relevantes)
- **T** - Time-boxed (Limitados en el tiempo)

Se suelen escribir en forma de checklist o descripción de un flujo en cuanto se obtengan historias de usuario susceptibles de entrar en un sprint, y se acaban de refinar en la planificación de sprint. Los criterios de validación ayudan al equipo de desarrollo a entender el funcionamiento del producto, de manera que estimarán mejor el tamaño de la historia subyacente y, cuando el equipo se encuentre en fase de desarrollo servirá de guía cuando el desarrollador tenga que tomar una decisión, tomará la más acertada. Finalmente el propietario de producto comprueba en la revisión de sprint, a través de estos criterios de validación y la definición de hecho, si cada una de las historias de usuarios se puede dar como hecha y finalizada.

Los criterios de validación pueden escribirse con el lenguaje natural, tal como el propietario del producto se expresa, existen diferentes formatos para escribirlos:

Comprobar [Criterios]

Demostrar [Comportamiento esperado]

Verificar que cuando [Rol] **hace** [Acción] **consigue** [Resultado / Comportamiento esperado]

Dado que [Contexto] **y adicionalmente** [Contexto] **cuando** [Evento] **entonces** [Resultado / Comportamiento esperado]

Una opción excelente es escribirlos con la técnica del comportamiento por escenarios propia de BDD (Behavior Driven Development) y con gherkin, un lenguaje específico creado especialmente para las descripciones de comportamiento de software. La sintaxis de gherkin es la siguiente:

(Scenario) Escenario [Número de escenario] [Título del escenario]:

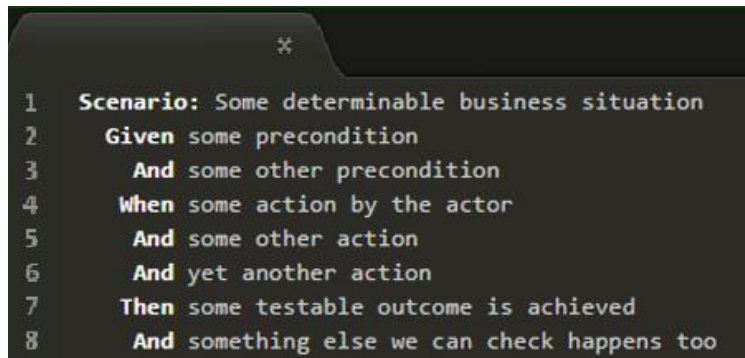
(Given) Dado que [Contexto] **y adicionalmente** [Contexto],

(When) cuando [Evento],

(Then) entonces [Resultado / Comportamiento esperado]

Derivado de esta sintaxis los elementos de los criterios de aceptación son:

- **Número de escenario:** Número (ejemplo 1, 2, 3 ó 4), que identifica al escenario asociado a la historia.
- **Título del escenario:** Describe el contexto del escenario que define un comportamiento.
- **Contexto:** Proporciona mayor descripción sobre las condiciones que desencadenan el escenario.
- **Evento:** Representa la acción que el usuario ejecuta, en el contexto definido para el escenario.
- **Resultado / Comportamiento esperado:** Dado el contexto y la acción ejecutada por el usuario, la consecuencia es el comportamiento del sistema en esa situación.



```
1 Scenario: Some determinable business situation
2   Given some precondition
3     And some other precondition
4   When some action by the actor
5     And some other action
6     And yet another action
7   Then some testable outcome is achieved
8     And something else we can check happens too
```

Ilustración 11: Ejemplo gherkin

Ejemplo de una historia de usuario y sus criterios de validación escritos en gherkin:

Como cliente
Quiero retirar dinero del cajero automático
Para poder evitar ir al banco a hacer una cola

Escenario 1: Cuenta tiene crédito
Dado que la cuenta tiene crédito
y que la tarjeta es válida
y que el cajero tiene dinero disponible
Cuando el cliente pide dinero
Entonces la cuenta es debitada
y el dinero es entregado al cliente
y el cliente recupera su tarjeta

Escenario 2: La cuenta excede el límite negativo acordado con el banco
Dado que la cuenta excede el límite negativo acordado con el banco
y que la tarjeta es válida
Cuando el cliente pide dinero
Entonces el cajero muestra un mensaje negando el pedido
y el dinero no es entregado al cliente
y el cliente recupera su tarjeta

Calidad en las historias de usuario

En 2003 Bill Wake desarrolló un **método llamado INVEST** para asegurar la calidad en la escritura de historias de usuario. El método sirve para comprobar la calidad de una historia de usuario revisando que cumpla una serie de características:

- **I** - Independent (independiente)
- **N** - Negotiable (negociable)
- **V** - Valuable (valiosa)
- **E** - Estimable (estimable)
- **S** - Small (pequeña)
- **T** - Testable (comprobable)

Independent (independiente)

Es ventajoso que cada historia de usuario pueda ser planificada e implementada en cualquier orden. Para ello las historias deberían de ser totalmente independientes (lo cual facilita el trabajo posterior del equipo). Resaltar que las dependencias entre historias de usuario pueden reducirse combinándolas en una o dividiéndolas de manera diferente.

Negotiable (negociable)

Una historia de usuario es una descripción corta de una necesidad que no incluye detalles. Las historias deben ser negociables ya que sus detalles serán acordados con el cliente o el usuario durante la fase de conversación. Por tanto, se debe evitar historias de usuario con demasiados detalles porque limitaría la conversación acerca de las mismas.

Valuable (valiosa)

Una historia de usuario tiene que ser valiosa para el cliente o el usuario. Una manera de hacer una historia valiosa es que la escriba el mismo.

Estimable (estimable)

Una buena historia de usuario debe de poder ser estimada con la precisión suficiente para ayudar al cliente, usuario o propietario del producto a priorizar y planificar su implementación. La estimación generalmente la realizará el equipo de trabajo y está directamente relacionada con el tamaño de la historia de usuario (una historia de usuario de gran tamaño es más difícil de estimar) y con el conocimiento del equipo de la necesidad expresada (en el caso de falta de conocimiento, serán necesarias mas fases de conversación acerca de la misma).

Small (pequeña)

Las historias de usuario deberían englobar como mucho unas pocas semanas/persona de trabajo. Incluso hay equipos que las restringen a días/persona. Una descripción corta ayuda a disminuir el tamaño de una historia de usuario facilitando así su estimación.

Testable (comprobable)

La historia de usuario debería ser capaz de ser probada (fase confirmación de la historia de usuario). Si el cliente o usuario no sabe como probar la historia de usuario significa que no es del todo clara o que no es valiosa. Si el equipo no puede probar una historia de usuario nunca sabrá si la ha terminado o no.

Hay empresas que han diseñado sus propias tarjetas como estas de Braintrust que, aparte de dar un look profesional, son completas en información y coherentes en espacio para los campos.

The image shows two examples of user story cards designed by Braintrust. The left card is titled 'USER STORY' and contains the following fields: 'As a', 'I want', 'So that', 'INVEST' (with six checkboxes), 'Size:', and 'Business Value:'. The right card is titled 'ACCEPTANCE CRITERIA' and contains a large text area, a 'Meets team's definition of ready?' checkbox, and the Braintrust logo.

Ilustración 12 Ejemplo de tarjetas diseñadas por Braintrust

Consejos de buenas prácticas:

- Siempre escribir las historias con el “qué”, evitar el “cómo”.
- No escribir una descripción exhaustiva, solo lo justo.
- Escribir el criterio de validación y ser suficientemente explícito.
- Estimar todas las historias, no hacerlo puede crear falsas expectativas.
- No fiar toda la información en las tarjetas, a veces es buena idea una documentación externa como una wiki por ejemplo.
- Nunca dar una historia por finalizada cuando está “prácticamente hecha”.

División de historias de usuario

Como parte del flujo continuo de toma de requisitos a través de historias de usuario existe la **reunión de refinamiento para mantener actualizada** la pila del producto. Es una reunión propia del propietario del producto, en la que junto con el equipo, trabaja en el refinamiento de la pila del producto. En esta reunión, tal como se ve en la imagen de la derecha, se añaden, repriorizan, eliminan y dividen elementos de la pila. Su objetivo es garantizar que las historias de usuario susceptibles de ser desarrolladas en corto plazo tengan un nivel de detalle y una estimación de esfuerzo suficiente para que el equipo pueda comprometerse.

La experiencia muestra que las historias de usuario más pequeñas mejoran el flujo y que las historias de usuario grandes implican una mayor incertidumbre funcional y una mayor dificultad para ser estimadas.

Dividir historias redundante en mejorar el entendimiento de las mismas, incrementar la exactitud de las estimaciones y hacer que estas sean más fáciles de priorizar.

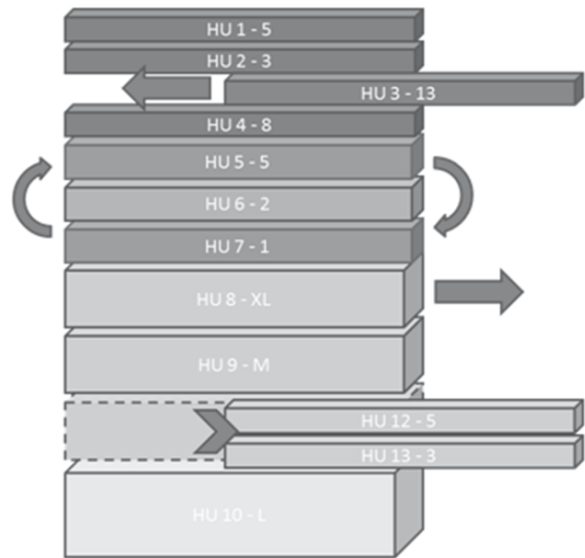


Ilustración 13: Refinamiento de la pila del producto

Podemos dividir las historias de usuario de **forma horizontal y vertical**. Horizontal significa división según el tipo de trabajo, por tecnologías por ejemplo, típico de las metodologías tradicionales. Esta forma de dividir en **horizontal** genera historias que **no tienen valor de negocio de forma individual**, solo el conjunto de todas ellas tiene valor. La división horizontal propicia el "pensar a modo de silos" en que cada miembro del equipo se focaliza solo en las de su especialidad, situación que tiende a producir cuellos de botella. La ingeniería de requisitos ágil evita este tipo de problemas con la multifuncionalidad de sus miembros, todo miembro participa en mayor o menor medida en los diferentes tipos de tarea. Por último las historias provenientes de divisiones horizontales no se pueden priorizar ya que no aportan ningún valor de negocio de forma individual.

A diferencia de la división horizontal, la **vertical** es más útil, una historia dividida en vertical genera historias que a su vez **tienen valor de negocio**, la funcionalidad no está dividida a lo largo de capas técnicas sino de capas funcionales. A semejanza de los incrementos resultantes de un sprint, las historias resultantes son como una porción de una tarta que incluye todas las capas técnicas necesarias.

Christiaan Verwijs describe **10 estrategias para dividir historias de usuario** de forma vertical:

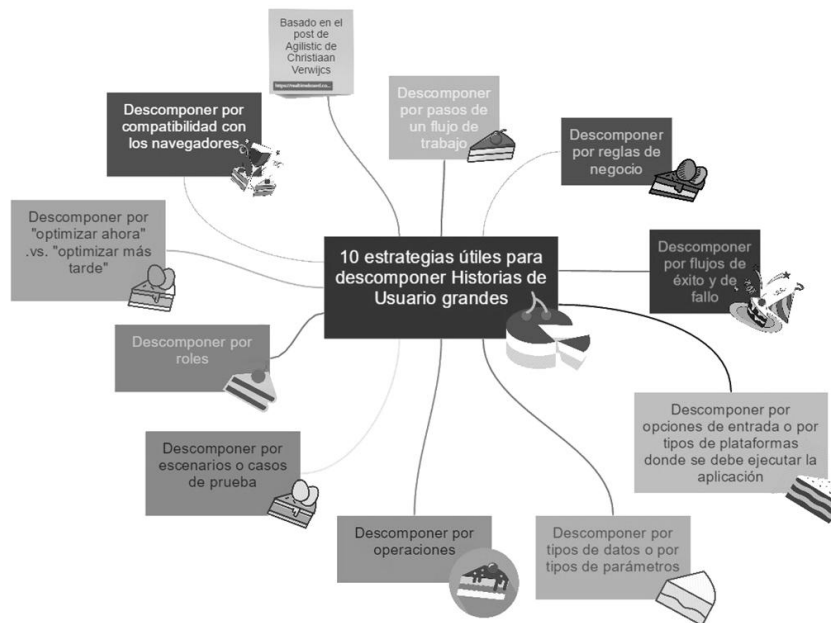


Ilustración 14: Esquema con las 10 estrategias de división de historias de usuario

- **Estrategia 1 - División por pasos de flujo de trabajo:** para historias de usuario que incluyen algún tipo de flujo de trabajo, estas se pueden dividir según los pasos individuales del flujo.
- **Estrategia 2 - División por reglas de negocio:** historias de usuario que conllevan implícita o explícitamente reglas de negocio se pueden dividir por estas reglas. Frecuentemente los casos de test implican importantes reglas de negocio, por tanto para esta división podemos basarnos en las pruebas.
- **Estrategia 3 - División por happy/unhappy flow:** las funcionalidades usualmente describen un flujo en que todo va bien y otros flujos en que se tratan desviaciones, excepciones o problemas, por tanto estos flujos también son una forma de dividir historias grandes.
- **Estrategia 4 - División por opciones/plataformas de entrada:** en caso de productos que han de rodar en diferentes plataformas, como portátiles, tablets, móviles... pueden dividirse las historias de usuario por su plataforma de entrada.
- **Estrategia 5 - División por tipos de datos o parámetros:** algunas historias de usuario se pueden dividir por sus parámetros de entrada o salida, como por ejemplo las diferentes opciones de una búsqueda.
- **Estrategia 6 - División por operaciones:** Hay historias que involucran las típicas operaciones de alta, lectura, modificación y baja (CRUD - create, read, update & delete), operaciones que pueden ser otra forma de división.
- **Estrategia 7 - División por casos/escenarios de test:** a veces hay historias que son difíciles de dividir funcionalmente, en este caso puede ayudar a preguntarse cuáles van a ser los escenarios de test de la historia y dividir por estos. Los escenarios pueden ser combinación de reglas de negocio, flujos que van bien y con excepciones, plataformas de entrada, etc.
- **Estrategia 8 - División por roles:** para historias de usuario que cubren diferentes roles, estas se pueden dividir por las funcionalidades propias de cada rol.
- **Estrategia 9 - División por optimizar ahora o más tarde:** las historias de usuario pueden ser implementadas en diferentes grados de perfección y optimización de la funcionalidad descrita.
- **Estrategia 10 - División por compatibilidad de navegador:** las historias de usuario para aplicaciones web a menudo tienen que trabajar en una amplia variedad de navegadores, los más modernos tienden a ser más compatibles con los estándares y los más antiguos suelen necesitar de personalizaciones para que todo funcione correctamente.

En todas estas estrategias la división reduce la incertidumbre, nos permite desarrollar las historias resultantes importantes y dejar historias menos importantes para desarrollos futuros.

Comparativa con otras formas de toma de requisitos

Historias de usuario versus casos de uso

Siempre que se menciona **casos de uso** e **historias de usuario** se produce algo de confusión. Hay quien ha logrado incluir casos de uso en su proceso ágil pero eso no quiere decir que las historias de usuario sean equivalentes a los casos de uso.

Un caso de uso es una técnica para capturar la funcionalidad deseada desde la perspectiva de como los usuarios (actores) interactúan con el sistema. Se utiliza en UML (Unified Modeling Language) en los diagramas de casos de uso, un lenguaje de modelado que se utiliza para describir un sistema desde sus perspectivas estática y dinámica. Este resulta en un lenguaje descriptivo pensado inicialmente para sencillez en la comunicación pero que no es cercano a la comunicación humana. En cambio una historia de usuario está escrita en un lenguaje coloquial, que funciona a modo de recordatorio de la conversación con el cliente.

Como comenta Alistair Cockburn en 2001 en su libro Agile software development, realmente las historias de usuario están más cerca de la captura de requisitos, la fase que sirve para extraer las necesidades del usuario, que la especificación de requisitos, como son los casos de uso.

Podríamos decir que una **historia de usuario dice el "qué"** quiere el cliente o usuario, y un **caso de uso entra en el "cómo"** lo quiere.

En el criterio de validación también hay diferencias de concepto, a diferencia de los casos de uso que requieren matrices de seguimiento de requisitos con porcentajes de terminado, las historias de usuario que incluyen el criterio de validación incluyen el terminado de forma binaria, o vale o no vale.

La propia agilidad se hace patente en el hecho de que las historias de usuario son vivas. El análisis funcional y técnico se hace poco antes del desarrollo, en la reunión de inicio de sprint en caso de **scrum**, por tanto el desglose en tareas lo hace el equipo, con lo que nivel de detalle y previsión supera en mucho al que pueda hacer un único arquitecto o analista funcional en los casos de uso.

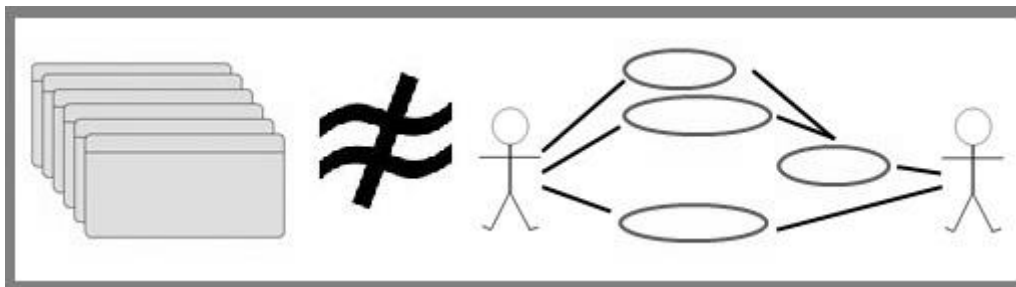


Ilustración 15: Historias de usuario no son parecidas a casos de uso

Cuando un proyecto comienza a seguir un método ágil se deberían de olvidar completamente los casos de uso y el equipo debería de centrarse solo en la realización de historias de usuario.

Historias de usuario versus requisitos funcionales

Por lo general se asocian las **historias de usuario** a **requisitos funcionales**, unas como los artefactos de los métodos ágiles y los otros los de las metodologías tradicionales. Sin embargo detrás de las historias de usuario hay aspectos que las diferencian, diferencias que desconocidas llevan a confusiones muy comunes.

Hemos de ser conscientes de que, aunque las historias de usuario describen funcionalidades que serán útiles para el cliente o usuario, y que se suelen escribir en tarjetas o post-its, son mucho más que eso, ya que implican una conversación posterior en que el equipo detalla junto con el usuario o cliente la funcionalidad a desarrollar.

Igual que en los requisitos funcionales, las historias de usuario dicen el "qué" pero no el "cómo" se desarrollará la funcionalidad. Por tanto, de igual manera, las historias de usuario no deben tener el nivel de detalle que tiene la especificación de un requisito funcional.

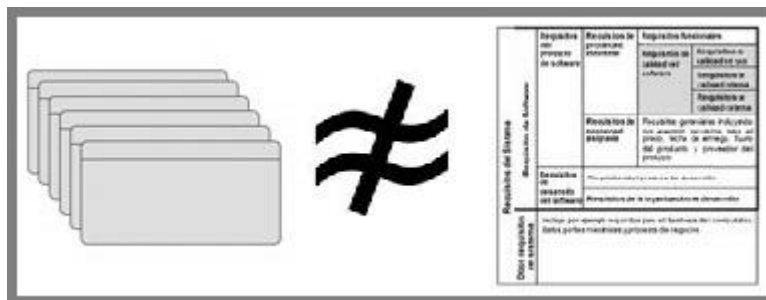


Ilustración 16: Historias de usuario no son parecidas a requisitos funcionales

Historias técnicas

La **pila de producto** contiene todos los "qués" a construir, elementos como historias de usuario, epics y temas. Pero en realidad, para poder realizar planificaciones tanto de sprint como de release, **debe de contener todo el trabajo a hacer**, y algunos de los elementos puede que no sean funcionales y sean **necesidades de carácter técnico** que no aporten valor de negocio, pero son consumidas por los elementos funcionales. También es interesante que negocio y propietario del producto tengan contacto con los aspectos técnicos, ya que así se forja un lenguaje común en ambas direcciones y se da consciencia de su importancia.

A estos elementos se les llama **historias técnicas** y son cosas como preparar un webserver, implementar un conjunto de tablas en una base de datos que va a ser consumida por varias funcionalidades, elementos de seguridad, escalabilidad, rendimiento, etc. Otros tipos de historias técnicas se centran en resolver deuda técnica y en refactorizaciones, otros en historias de exploración como un análisis técnico o uno funcional que sirve para despejar incertidumbre sobre alguna historia de usuario.

Las historias técnicas se escriben directamente en **texto técnico claro y preciso**, sin un patrón como ocurre con las historias de usuario. También tienen **criterios de validación** asociados que se comprueban en la revisión de sprint por la audiencia técnica correspondiente.

Aunque el propietario del producto sea el responsable de la pila y por tanto de todos sus elementos, en el caso de las historias técnicas los verdaderos **"propietarios" son perfiles de carácter técnico** como el equipo o un arquitecto. Estos no solo son responsables de la definición de la misma sino también de responder a dudas y preguntas y aclaraciones en la planificación y en la entrega de las historias.

Se pueden identificar diferentes tipos de historias técnicas:

- **Arquitectura:** construyen elementos como las API's que crean la estructura, funcionamiento e interacción entre distintas las partes del software.
Ejemplo: "Implementar un sistema de login seguro".
- **Infraestructura de producto:** historias que son consumidas directamente por historias de usuario. Esto podría incluir infraestructura nueva y/o modificada, y oportunidades de refactorización originada por alguna necesidad funcional.
Ejemplo: "Preparar los servidores de base de datos y web".
- **Infraestructura del equipo:** historias que respaldan al equipo en su capacidad para entregar software. Suelen ser historias para herramientas y marcos de pruebas, métricas, diseño, planificación... y también pueden implicar que el equipo "desarrolle" o "compre e instale" algo.
Ejemplo: "Preparar un sistema de integración continua"
- **Refactorización:** estas son historias que representan candidatos para refactorizar, como por ejemplo lo es la deuda técnica. Pero no solo el código necesita de refactorización, también puede incluir diseños, automatización, herramientas y cualquier documentación de proceso.
Ejemplo: "Homogeneizar el código de la función de cálculo de préstamos".
- **Spikes:** historias de exploración limitadas en tiempo que dan respuesta a una cuestión, o reúnen información para una toma de decisión posterior o el diseño de una solución.
Ejemplo: "Evaluar Oracle versus SQL/Server".
 - **Spike técnico:** si no estamos seguros de cómo desarrollar algo desde un punto de vista técnico creamos este tipo de spike, una breve actividad que se centra en encontrar un enfoque de desarrollo, en determinar la factibilidad y el impacto de las estrategias de diseño.
 - **Spike funcional:** sirven a los equipos para descubrir los detalles de las funcionalidades y los diseños a través de la creación de prototipos y llegar a entender exactamente lo necesita el cliente.

User Story Mapping

User Story Mapping es una técnica que Jeff Patton describe en 2014 en su libro *User Story Mapping: Discover the Whole Story, Build the Right Product*, y es muy útil para construir una pila de producto que vaya más allá de una lista unidimensional de historias de usuario y epics. Esta técnica nos permite obtener los siguientes beneficios:

- **Visión compartida:** a través de la participación en la elaboración del User Story Map todos los involucrados construirán una visión compartida del producto a obtener, lo que facilitará mantener el foco en la solución a lo largo de su construcción.
- **Alineación:** los miembros del equipo y negocio (propietario del producto e interesados o clientes) podrán estar alineados sobre lo que se va a construir sabiendo el por qué o las razones subyacentes.
- **Mejor entendimiento de lo que quieren los clientes:** a través del modelado de los clientes y de lo que van a hacer con el producto, se puede alcanzar un alto nivel de entendimiento de las verdaderas necesidades de los clientes.
- **Mejor entendimiento de los problemas que enfrentan los clientes:** aplicando esta técnica se puede lograr un alto nivel de empatía con los clientes, tanto por participar en la dinámica, como también por tener la oportunidad de ponerse en el lugar del cliente a la hora de modelar lo que hará con el producto y cómo lo hará, pudiendo pensar en cuáles son sus problemas y cómo solucionarlos mediante el producto que se va a construir.
- **Un mapa de historias de usuario ordenado por versión:** al modelar un backbone del flujo del cliente a través del producto, podemos definir el conjunto de historias que conformarán el Mínimo Producto Viable (MVP) a sacar al mercado, que le permita a los usuarios llevar a cabo todo el flujo de una manera básica (con por lo menos una funcionalidad por cada actividad obligatoria en el flujo), así como una idea inicial de las siguientes versiones que se prevé sacar posteriormente, que puede cambiar en función del feedback y otros elementos del mercado obtenidos cuando salga a producción el MVP y cada una de las versiones siguientes.



Ilustración 17: Pila de producto plana y pila de producto construida con la técnica del User Story Mapping

Se trata de una técnica **colaborativa de construcción de la pila de producto**. Para ello se reúnen los involucrados con la definición, uso y construcción del producto, como interesados, clientes o usuarios finales, el propietario del producto y el equipo con un moderador, que en la mayoría de los casos es el scrum master o el agile coach. El objetivo es que de manera conjunta se definan, descubran, prioricen y estimen las historias de usuarios y/o épicas que se prevén como parte del producto a construir.

Para emplear la técnica del User Story Mapping se van definiendo los siguientes elementos:

1. **Backbone del User Story Map:** el backbone o espina dorsal del User Story Map captura las actividades de alto nivel que un usuario va a realizar cuando use el producto que se quiere construir. Por ejemplo, el backbone de un proceso de compra de un libro en formato digital en una web de venta de libros on-line sería:



Ilustración 18: Ejemplo de Backbone de un User Story Mapping

2. **Historias de usuario asociadas con cada actividad del proceso ordenadas por valor (las más valiosas en la parte superior):** para cada actividad que va a realizar el usuario con el producto se definen las historias de usuario que le van a permitir realizar la actividad. Luego ordenarlas de arriba abajo colocando las más valiosas o prioritarias más arriba. Para el ejemplo de la compra de un libro en formato digital en una web de venta de libros, un conjunto de historias de usuario asociado a cada actividad y priorizadas por valor de arriba hacia abajo sería:



Ilustración 19: Backbone e historias de usuario de un User Story Mapping

3. **Mínimo producto viable (v 1.0) y las siguientes versiones que se prevé liberar del producto:** se determinan cuáles son las historias de usuario que compondrán el mínimo producto viable (v 1.0) y las siguientes versiones reflejando esto en el User Story Map, como por ejemplo:



Ilustración 20: Backbone, historias de usuario y versiones de un User Story Mapping

Una vez que se ha llegado hasta aquí, el equipo puede hacer una **estimación inicial del esfuerzo** necesario para realizar cada historia de usuario. De esta manera el propietario del producto puede tener una estimación del esfuerzo requerido para desarrollar el MVP y cada una de las versiones definidas en el User Story Map. Si conoce la **velocidad** del equipo, es decir, cuántos puntos de historia puede hacer el equipo por sprint y la duración de los sprints, puede hacer un **gráfico de producto** y obtener una estimación inicial de cuándo es probable que se entreguen cada una de las versiones definidas hasta el momento. Esta estimación inicial se irá ajustando y podrá ir cambiando a medida que se tengan estimaciones más precisas y se hagan entregas de software funcionando que proporcionen feedback sobre el producto y sus funcionalidades.

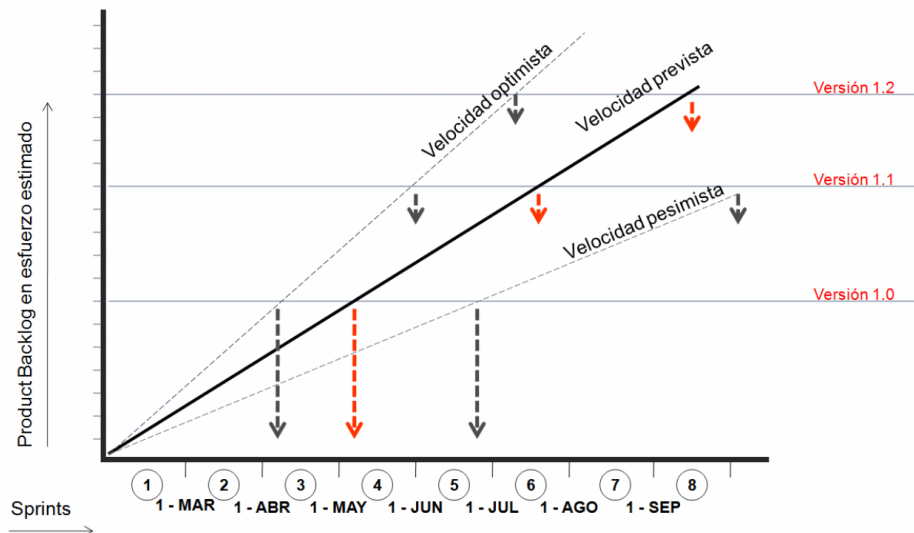


Ilustración 21: Previsión de lanzamiento de versiones sobre gráfico de producto

Como se ha visto, la técnica del User Story Mapping es una forma colaborativa poderosa y efectiva para definir la visión de un producto, su pila de historias de usuario y epics, el MVP y las siguientes versiones en un RoadMap del producto, y permite que de manera flexible se puedan ir reflejando los cambios que surjan a lo largo del proceso de desarrollo y de entregas.

Apéndice: Historias de usuario en otros ámbitos

La agilidad se está propagando a muchos ámbitos de la compañía. Un área especialmente propicia para ello es **marketing**, donde las **microcampañas con feedback rápido en forma de sprints** hacen que scrum sea una forma idónea de trabajo. Jim Ewel, experto en Agile Marketing, compara las historias de usuario de marketing con las de TI poniendo el foco en sus diferencias:

Historias de usuario de TI	Historias de usuario de marketing
Bajo nivel	Alto nivel
Número elevado de estas en la pila	Relativamente pocas en la pila
Hemisferio izquierdo del cerebro	Hemisferio derecho del cerebro
Focalizadas en funcionalidades	Focalizadas en resultados

Las historias de usuario de TI son detalladas y tienen una granularidad muy fina, las de marketing son cosas amplias, por tanto la pila de producto de marketing suele ser corta. Lo interesante es en lo que se focalizan, las historias de TI al hablar de funcionalidades están orientadas a la parte racional de las personas, las de **marketing a la parte emocional**, cuando la gente habla de marketing, por ejemplo de anuncios que les han impresionado, hablan de cómo les han hecho sentir, acerca de como cambió su día o cómo les hizo reír histéricamente.

La solución a la que apunta una historia de marketing nos hace **centrar en lo que los clientes quieren lograr**, ya sea resolver un problema, aliviar un dolor o aspirar a algo mejor a través de nuestro producto. El "para" trata la pregunta que un buen marketing debe responder: ¿Qué hay en él para mí? (What's In It For Me? - WIIFM). Por tanto el **patrón utilizado para historias de usuario de marketing** varía:

Como [cliente ideal], quiero [nuestra solución], para [que sus problemas desaparezcan]

*Como madre quiero grabar y
compartir vídeos de los niños
para poder compartir momentos
importantes con los abuelos,
tíos y amigos*

Ilustración 22: Ejemplo de una historia de usuario de marketing

Bibliografía

Alistair Cockburn (2001). *Agile software development*. Boston: Addison-Wesley Professional.

Bill Wake (2003). *INVEST in good stories, and smart tasks*. <http://xp123.com/articles/invest-in-good-stories-and-smart-tasks/>

Christiaan Verwijs (2015). *10 useful strategies for breaking down large User Stories (and a cheatsheet)*. <http://blog.agilistic.nl/10-useful-strategies-for-breaking-down-large-user-stories-and-a-cheatsheet/>

Dai Clegg (2004). *Case Method Fast-Track: A RAD Approach*. Boston: Addison-Wesley Longman.

Don Reinersten (2009). *The Principles of Product Development Flow: Second Generation Lean Product Development*. Celeritas Pub.

Gertrudis López (2016). *Esquema con las 10 estrategias de división de historias de usuario*. <https://mm.tt/682181724?t=LjoGABkbaF>

Heather Krebsbach (2016). *Know thy customer: agile's essential guide to user story maps*. <https://www.atlassian.com/blog/2016/05/guide-to-agile-user-story-maps>

Javier Garzás (2014). *Agilidad y Lean. Gestionando los proyectos y negocios del s. XXI*. <https://www.miriadax.net/>

Jeff Patton (2014). *User Story Mapping: Discover the Whole Story, Build the Right Product*. O'Reilly Media Inc.

Jim Ewel (2011). *Agile Marketing*. <http://www.agilemarketing.net/>

Kent Beck (1999). *eXtreme Programming Explained*. Boston: Addison-Wesley Professional.

Mike Cohn (2004). *User Stories Applied for Agile Software Development*. Boston: Pearson Education, Inc.

Ron Jeffries, Ann Anderson, Chet Hendrickson (2001). *Extreme Programming Installed*. Boston: Addison-Wesley.

Tabla de ilustraciones

Ilustración 1: Gráfico con las tres fases de la fórmula de las tres C's.....	9
Ilustración 2: Gráfico con los cuatro niveles de tamaño con que trata los requisitos la gestión ágil	11
Ilustración 3: Granularidad de la pila de producto	12
Ilustración 4: Ejemplo de una tarjeta de historia de usuario.....	13
Ilustración 5: Ejemplo una historia de usuario	14
Ilustración 6: Ejemplo una historia de usuario	15
Ilustración 7: Ejemplo de User-Persona	16
Ilustración 8: Cartas planning poker con la serie de Fibonacci	18
Ilustración 9: Cartas de estimación con tallas de camisa.....	19
Ilustración 10: Ejemplo de priorización WSJF	21
Ilustración 11: Ejemplo gherkin	23
Ilustración 12 Ejemplo de tarjetas diseñadas por Braintrust	25
Ilustración 13: Refinamiento de la pila del producto.....	26
Ilustración 14: Esquema con las 10 estrategias de división de historias de usuario	27
Ilustración 15: Historias de usuario no son parecidas a casos de uso	28
Ilustración 16: Historias de usuario no son parecidas a requisitos funcionales	29
Ilustración 17: Pila de producto plana y pila de producto construida con la técnica del User Story Mapping	31
Ilustración 18: Ejemplo de Backbone de un User Story Mapping	32
Ilustración 19: Backbone e historias de usuario de un User Story Mapping	32
Ilustración 20: Backbone, historias de usuario y versiones de un User Story Mapping	32
Ilustración 21: Previsión de lanzamiento de versiones sobre gráfico de producto	33
Ilustración 22: Ejemplo de una historia de usuario de marketing.....	34

Índice

- Asignación, 14
- Buenas prácticas, 27
- Campos, 13
- Caso de uso, 30, 31
- Cómos, 11, 30, 31
- Comprobable, 26
- Coste de la demora, 22
- Criterio de finalización, 14
- Criterio de validación, 14, 23, 27, 30, 32
- Dependencia, 14
- Descripción, 13
- División horizontal, 28
- División vertical, 28
- Epic, 11, 33
- Equipo, 11, 13, 14, 26, 30, 31, 32, 33
- Esfuerzo, 13
- Estimable, 26
- Estimación, 13, 14, 19, 26, 27, 35
- Fórmula tres C's, 9, 26, 31
- Funcionalidad, 9, 13, 21, 31
- Gherkin, 23
- Gráfico del producto, 35
- Granularidad, 11, 12
- Historia de usuario, 9, 12, 13, 14, 16, 18, 21, 23, 26, 28, 30, 31, 32, 33, 34, 35, 36
- Historia técnica, 32
- ID, 14
- Independiente, 26
- Información, 13
- Marketing, 36
- Método INVEST, 26
- Mínimo Producto Viable, 33, 34, 35
- Módulo, 11, 14
- Negociable, 26
- Observación, 14
- Pequeña, 26
- Pila de producto, 12, 32, 33
- Planning poker, 19, 20
- Prioridad, 12, 13, 14, 18, 21
- Propietario del producto, 14, 21, 26, 33
- Pruebas, 14, 23, 26
- Punto de historia, 13
- Qués, 9, 11, 30, 31, 32
- Refinamiento, 28
- Requisito, 9, 31
- Riesgo, 14
- ROI, 21
- Scrum, 10, 12, 13, 14, 30
- Serie de Fibonacci, 19
- SMART, 23
- Sprint, 14
- Tallas de camisa, 20
- Tarea, 11
- Tarjeta, 13, 15, 27
- Técnica MoSCoW, 21
- Tema, 11
- Título, 14
- UML, 30
- User Story Mapping, 33
- User-Persona, 16
- Usuario, 16
- UX, 16
- Valiosa, 26
- Valor, 14, 18, 21, 26, 28
- Visión, 11
- WSJF, 22
- XP, 10