Conceptualización

Marcos Bermejo

PID_00177693

Universitat Oberta de Catalunya

Índice

1. El product owner

1.1. El papel del *product owner*

"The Product Owner is the one and only person responsible for managing the Product Backlog and ensuring the value of the work the team performs. This person maintains the Product Backlog and ensures that it is visible to everyone".

Ken Schwaber (2009). *Scrum Guide* (pág. 5)

El ***product owner*** (PO) es el encargado de guiar los esfuerzos del equipo de desarrollo para lograr crear un producto que genere los beneficios deseados.

Esta definición incluye actividades como las siguientes:

- establecer y comunicar la visión del producto;
- gestionar la pila de producto;
- planificar las entregas;
- involucrar a clientes, usuarios y otros interesados en el proyecto;
- gestionar el presupuesto;
- preparar el lanzamiento de producto;
- colaborar con el equipo.

El papel del PO es crucial, no solo en cuanto al producto, sino en toda la gestión del ciclo de vida del mismo.

Centralizar la gestión y la responsabilidad de las entregas del producto en un único puesto asegura que haya continuidad y promueve el pensamiento a largo plazo.

Pero el PO no tiene un papel solitario sino que debe estar en continua colaboración con el equipo de desarrollo y asegurarse de que el trabajo que ejecuta el equipo sigue la línea marcada por la visión del producto.

1.2. *Product owner* frente a los cargos tradicionales

La función del *product owner* une las competencias y habilidades que tradicionalmente se daban en varios puestos como por ejemplo el *project manager*, el *product manager* o los analistas de negocio, entre otros.

Es una función muy marcada por el contexto: las habilidades requeridas dependerán mucho del tipo de negocio en el que se está trabajando y de la naturaleza y tamaño de los proyectos, lo que da lugar a un puesto muy heterogéneo:

- Para un **producto comercial**, el PO será por ejemplo alguien con experiencia en el segmento de mercado al que apunta el producto.

- Para un **proyecto a medida**, el PO será un analista de negocio de la empresa cliente. Clientes, usuarios, gestores de proyecto, analistas de negocio, arquitectos o CEO[1] (director ejecutivo) adoptar el papel de PO en función del proyecto y de sus características.

(1)Del inglés *Chief Executive Officer*

1.3. Características de un buen product owner

La elección de un buen *product owner* es una decisión crucial para el éxito de un proyecto Scrum. Para ello, y al ser el puesto de *product owner* un cargo relativamente nuevo, es importante elegir la persona adecuada para el lugar en función de sus habilidades y no de su anterior puesto.

Algunas de las habilidades y aptitudes que debe tener un buen *product owner* son las siguientes:

1) Visionario y emprendedor. Tiene que ser capaz de ver el producto final y ser capaz de comunicar esta visión a la vez que fomenta la creatividad y la innovación dentro del equipo.

2) Líder. Debe tener capacidad de guiar al equipo desde esta primera visión hasta un producto final que cumpla con las expectativas de clientes y usuarios, dirigiendo y tomando decisiones en momentos complicados.

3) Jugador de equipo. Tiene que saber colaborar activamente con el resto del equipo y estar disponible para ellos cuando sea necesario para ayudar a alcanzar el objetivo común.

4) Comunicador y negociador. Debe ser capaz de conseguir un entendimiento y comunicación fluidos entre los diferentes interesados en el proyecto y llegar a compromisos (de fecha y alcance, entre otros) cuando sea necesario.

5) Con autoridad. Debe tener el suficiente apoyo de las capas de dirección para poder tomar decisiones y liderar sin interferencias el proceso del equipo.

6) Comprometido. Con el proyecto y con el equipo, seguro, de confianza, enérgico.

7) Calificado. Debe tener el suficiente conocimiento del mercado, ser capaz de gestionar presupuestos, de entender y describir las necesidades y los requisitos del proyecto.

1.4. Trabajo en equipo

Para que el trabajo del *product owner* dentro del equipo sea lo más efectivo posible, este debe formar parte del propio equipo, equipo autoorganizado y multifuncional, de forma que se genere la confianza necesaria entre los demás miembros del equipo y el *product owner*.

> Si quienes forman equipo ven al *product owner* como un miembro ajeno al equipo, la comunicación no será lo suficientemente fluida ni exitosa.

Por eso, es importante que el *product owner* esté disponible el máximo tiempo posible para el equipo Scrum, con el objetivo de ayudar a entender y superar cualquier duda o problemas sobre el producto, que puedan surgir durante el desarrollo. Idealmente, esta disponibilidad tiene que llegar al punto en el que el equipo Scrum y el *product owner* estén ubicados en el mismo espacio físico.

En el supuesto de que no se pueda llegar al punto de compartir el mismo espacio, se tiene que mantener el máximo contacto posible, maximizando los encuentros frente a frente entre el *product owner* y el resto del equipo.

1.5. Trabajando con usuarios y clientes

Para poder crear el mejor producto o solución posible, hay que tener presente que este (el producto o la solución) no es un fin en sí mismo, sino que suele ser una solución o mejora a un problema o inquietud de un grupo de usuarios o clientes, por eso es importante poder contar con el *feedback* de esos usuarios o clientes de la manera más regular y temprana posible.

Hacer entrevistas a usuarios ante prototipos, invitarlos a las demostraciones de final de *sprint* o la entrega frecuente de producto suelen ser formas que tiene el *product owner* para no perder el foco en el usuario o cliente tanto al principio como durante el tiempo de desarrollo del proyecto o solución.

1.5.1. Errores comunes

El hecho de que el puesto de *product owner* sea relativamente nuevo hace que muchas organizaciones no tengan claro cómo llevar a cabo el proceso para incorporar esta figura a sus proyectos. El siguiente listado recoge algunos de los errores comunes a la hora de incorporar la función de *product owner* a una organización:

- **Poca autoridad**. Si el *product owner* no tiene la suficiente autoridad o el suficiente apoyo de la dirección, puede no ser capaz de tomar decisiones necesarias y cruciales para el desarrollo del proyecto durante el mismo, por lo que el proyecto puede tomar un rumbo que lo lleve al fracaso.

- **Sobrecarga de funciones**. Con un *product owner* sobrecargado de trabajo se pueden producir cuellos de botella que eviten que el proyecto avance a la velocidad necesaria además de las consecuencias personales y de salud para la persona (que indirectamente afectan también al proyecto y a la organización). Ser *product owner* de muchos proyectos o no tener suficiente apoyo por parte del equipo pueden ser motivos de este síntoma.

- **División de funciones**. Este es el error contrario al anterior. En este, lo que se hace es dividir las funciones que corresponderían a un *product owner* entre diferentes personas; pérdida de liderazgo, pérdida de implicación en el proyecto y responsabilidad diluida, entre otros, son algunos de los problemas derivados de esta actuación.

- **Trabajo a distancia**. Cuando el *product owner* y el equipo trabajan a distancia (entendiendo por distancia todo aquello que va desde trabajar en diferentes continentes hasta simplemente trabajar en despachos diferentes) la comunicación se ve afectada y da pie a malentendidos, retrasos y falta de confianza, entre otros.

- ***Proxy***. Este error se da cuando una persona actúa como representante del PO real (probablemente porque este está sobrecargado de trabajo, trabaja a distancia o directamente evita alguna de las partes de su trabajo).

- **Comité de *product owner***. En lugar de tener a una persona guiando el desarrollo del proyecto e intentando llegar a la visión de este, se tiene a un grupo de personas, lo que suele desembocar en tensiones debido a políticas, diferentes visiones sobre el producto, intereses, por ejemplo.

2. Visión del producto

La **visión del producto** es una representación del objetivo final: el producto con las características o funcionalidades que tendría el producto idealmente. La visión debe servir de guía durante todo el proceso de desarrollo del producto y ayudar durante la toma de decisiones.

Sin un **objetivo** o **visión** que haga de guía, las decisiones que se tomen serán reactivas y a corto plazo (motivadas únicamente por hechos puntuales y actuales) y pueden comprometer el largo plazo del proyecto.

Una visión efectiva tiene que ayudar a:

- Entender quién es el público objetivo del producto, quién lo utilizará.
- Comprender qué necesidades está intentando cubrir y cuál es el valor añadido del producto.
- Descubrir las características del producto que son críticas para el éxito de este y en cuáles de estas el producto tiene que ser excelente.
- Ver en qué se asemeja y en qué se diferencia el producto del de los competidores u otros productos similares ofrecidos por la misma compañía.
- Establecer un precio y ver cuál será el plan de ventas del producto.
- Decidir si el producto es viable o no para la empresa y en qué condiciones.

2.1. Características de la visión

La visión se tiene que caracterizar por ser:

1) Compartida y unificada. Para que la visión del producto sea realmente efectiva, es necesario que sea única y además que sea compartida por todos los interesados en el proyecto. No sirve de nada hacer un proyecto que cumpla a la perfección con la visión del equipo de desarrollo si esta no coincide con la de los inversores del proyecto. Compartir y unificar esta visión de producto tiene que ser uno de los primeros objetivos a la hora de realizar un proyecto, antes de que sea demasiado tarde y las expectativas de los diferentes implicados en el proyecto se vean afectadas.

2) Corta y atractiva. Una buena visión tiene que ser breve y concisa. Debe intentar reflejar únicamente los tres o cuatro beneficios principales (¡no características!) del producto, los que realmente marcan la diferencia y que sean capaces de atraer a futuros usuarios o compradores.

Elevator's pitch

El ***Elevator's pitch*** (¿podéis explicar el producto en el tiempo que tardáis en subir en el ascensor?) suele ser una buena demostración de una visión corta y atractiva.

2.2. Mínimo producto viable

Para poder crear una visión de producto, es necesario echar un vistazo al futuro e intentar predecir lo que va a pasar. Lamentablemente, nadie puede predecir el futuro con exactitud y, del mismo modo, nadie puede elaborar estudios de mercado que aseguren la viabilidad de un producto de manera 100% fiable.

La clave para superar esta falta de fiabilidad es la **adaptación**: adaptación a cambios de requisitos, a cambios en las condiciones del mercado, a cambios de estrategia, entre otros.

Para conseguir esta adaptación, una buena estrategia consiste en el llamado mínimo producto viable[2].

(2)MPV en adelante.

2.2.1. ¿En qué consiste?

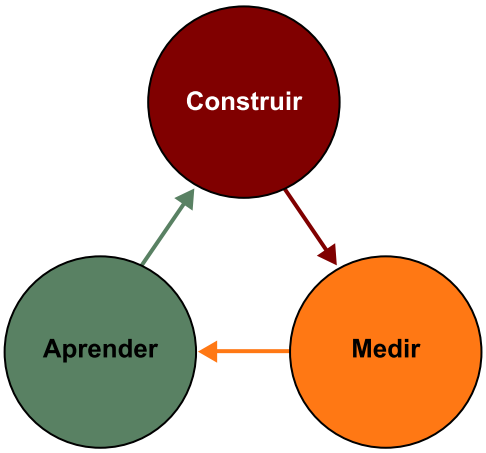
El **MPV** consiste en desarrollar el conjunto mínimo de funcionalidades que nos sirva para confirmar si existe o no un mercado interesante para el producto. Este conjunto mínimo de funcionalidades puede ser lo más pequeño que se quiera y dependerá de la etapa del producto donde nos encontremos.

Por ejemplo, las primeras etapas de un MPV suelen consistir únicamente en una *landing page*, que anuncia el producto y sus beneficios y plan de precios, entre otros, sin tenerlo desarrollado todavía (serviría para medir el grado de interés en el producto). Una vez se ha medido la recepción del producto en el mercado y se ha obtenido el *feedback* de clientes y usuarios (o potenciales clientes y usuarios), es hora de iterar el MPV.

2.2.2. Iteraciones del MPV

El MPV es un medio para conseguir llegar a una versión final del producto (si es que existe una versión final del producto). Para lograr ir avanzando en este objetivo de versión final, el MPV se va desarrollando en iteraciones.

Cada iteración de un MPV tiene la siguiente forma (de acuerdo con la metodología *Lean Startup*):



En cada una de estas iteraciones se intenta construir (***build***) lo que permite medir (***measure***) cómo funciona el producto en el mercado y, a partir de estas medidas, aprender (***learn***) qué necesitamos construir (***build***) en la siguiente iteración.

De este modo, el equipo se va acercando a la versión final del producto pero dando pequeños pasos y adaptando el producto en función del *feedback* recibido (en lugar de construir un producto basado en hipótesis que pueden ser ciertas o no).

2.2.3. ¿Qué beneficios aporta?

Algunos de los beneficios que aporta el MPV son los siguientes:

- minimizar los riesgos,
- minimizar el tiempo que tardan las versiones del producto en llegar al mercado (*time to market*),
- minimizar el tiempo de adopción,
- un menor coste de desarrollo, y
- un retorno de la inversión mayor.

3. Técnicas para crear la visión

3.1. *Inception deck*

La ***inception deck*** es una técnica que consiste en reunir a todas las personas implicadas en el desarrollo de un producto y hacer que contesten de manera colaborativa a una serie de preguntas y hagan una serie de ejercicios sobre el producto.

El objetivo es que, una vez acabado el ejercicio (que puede durar desde un par de días hasta un par de semanas, dependiendo de la complejidad del producto y del número de personas implicadas), todo el mundo comparta la misma visión del producto, así se evita después que las expectativas no se cumplan y que el producto sea diferente del que cada persona esperaba.

Los puntos que se tienen que trabajar en la *inception deck* son los siguientes:

1) ¿Por qué estamos aquí? Intentar definir de forma breve por qué estamos aquí, quiénes son nuestros clientes y por qué hemos decidido sacar adelante este proyecto. Tener claro el porqué del producto ayudará a tomar mejores decisiones y más informadas durante el proyecto.

2) Crear un *elevator pitch*. ¿Cómo describiríamos el proyecto si tuviéramos treinta segundos para describirlo? ¿Qué puntos resaltaríamos y cómo intentaríamos atraer la atención de un posible usuario, comprador o inversor sobre el producto?

3) Diseñar una caja de producto. Si el producto se tuviera que vender en supermercados, ¿cómo sería la caja que lo contendría? ¿Cuál sería el logo? ¿Qué imagen pondríamos en la caja? ¿Qué beneficios se verían reflejados?

El objetivo es crear aquel envoltorio que consiga que el usuario compre el producto en lugar del producto de al lado.

4) Crear una *NOT-List*. Del mismo modo que debemos tener claro qué queremos que forme parte del producto, también debemos tener presente qué es lo que no queremos dentro del producto (al menos de momento).

5) Conoced a vuestros vecinos. La creación de un producto puede abarcar a muchísimas personas, no solo a aquellos directamente implicados en su construcción. El objetivo de este ejercicio es intentar detectar a todos aquellos externos al producto, pero con los que necesitaremos colaborar o trabajar durante el proceso, de forma que podamos empezar a establecer una relación cordial con ellos antes de empezar el trabajo.

6) Mostrar la solución. Esbozar a alto nivel las principales líneas de la arquitectura técnica de la solución para asegurar que todo el mundo tiene la misma idea de cómo estará estructurado en el aspecto técnico el producto.

7) ¿Qué nos quita el sueño? ¿Cuáles son los principales miedos del proyecto? ¿Qué riesgos creemos que pueden hundir el producto? Definir los posibles riesgos, hablar sobre ellos y ver cómo se pueden evitar (dentro de lo posible) y cómo se puede ayudar a afrontar el proyecto con más garantías de éxito.

8) ¿Cuál es el tamaño? ¿Hablamos de un proyecto de un mes? ¿De seis meses? ¿De un año? Tener una idea aproximada nos ayudará a acotar mejor el alcance del proyecto.

9) ¿Cuáles son las prioridades? Dentro de todo proyecto, hay una serie de aspectos que siempre se deben tener en cuenta: **tiempo**, **presupuesto**, **alcance** y **calidad**. ¿Cuáles de estos son los más prioritarios para el proyecto en este momento? ¿Podemos ser flexibles con el tiempo pero con un alcance cerrado? ¿Tenemos un presupuesto estricto o tenemos libertad para gastar? También debemos tener en cuenta aquellos otros aspectos que sean relevantes para el producto: ¿queremos un producto sencillo de utilizar? ¿O priorizamos un producto estéticamente agradable? ¿Es más importante la velocidad? Definir estas prioridades nos facilitará la toma de decisiones durante el desarrollo del producto.

10) ¿Cuánto nos va a costar? ¿En cuánto tiempo podemos tenerlo listo? ¿Cuánto dinero nos costará? ¿Qué equipo necesitamos para desarrollarlo?

Responder a estas preguntas (sin entrar mucho en detalle) nos dará una idea aproximada de lo que nos puede costar el proyecto y puede ser un buen punto para decidir si el proyecto es viable o no en su forma actual.

3.2. Prototipos y *mockups*

El proceso de definir la visión de un producto es un proceso de experimentación e investigación. Los **prototipos** y los ***mockups*** son una muy buena herramienta para trabajar esta experimentación que ayuda a obtener un *feedback* de manera muy rápida e iterativa.

La creación de *mockups* nos permite obtener conocimiento sobre el producto, sobre cómo tendría que ser su apariencia, qué opciones arquitectónicas y tecnológicas son viables o incluso ver si la idea es realizable o no.

Podemos tener prototipos de diferentes tipos:

- En papel, denominados también ***sketches*** o ***mockups*** y que sirven para validar o desestimar una idea referente sobre todo a aspectos visuales del producto.

- Prototipos ejecutables, también llamados ***spikes***, que permiten testear una o varias ideas funcionales sobre el producto.

3.3. Personas y escenarios

La **técnica de personas** sirve para intentar encontrar el perfil objetivo de usuarios del producto. Una persona representa un arquetipo de cliente objetivo. La técnica consiste en crear diferentes perfiles que describan en qué parte del producto están interesados y cómo piensan utilizar el producto.

Una vez se han creado los diferentes perfiles de posibles usuarios del producto, se empieza a trabajar la parte de los escenarios. Cada **escenario** representa una situación en la que cada una (o algunas) de las personas antes referidas utilizan el producto para obtener algún tipo de beneficio en su vida cotidiana.

Lo que se persigue es obtener información sobre cómo el producto puede aportar el máximo valor a diferentes perfiles de usuarios, centrándose en estos usuarios y en cómo tienen que interactuar con el producto para obtener este beneficio.

3.4. Definir un *roadmap*

A medida que el producto va creciendo y madurando, el esfuerzo de crear la visión va disminuyendo, pero de todos modos las diferentes versiones del producto necesitan mantener unos objetivos claros para seguir evolucionando en la dirección correcta.

Para conseguir esto se utiliza el ***roadmap***, una herramienta de planificación que permite ver cómo se quiere que el producto vaya evolucionando en las versiones siguientes y que ayuda a coordinar su desarrollo con productos de otras líneas o productos relacionados (la cartera).

Un *roadmap* tiene que ser lo más sencillo posible, evitando entrar en demasiados detalles y centrándose en los aspectos más importantes de cada una de las versiones que vendrán. Por ejemplo, fecha de lanzamiento, usuarios objetivo y cuatro o cinco funcionalidades principales.

El *roadmap* tiene que ser una herramienta viva, que puede (y seguramente debe) cambiar en el tiempo a medida que se vayan viendo las reacciones del mercado a las diferentes versiones del producto y qué adaptaciones se tienen que hacer respecto a la ruta prevista inicialmente.

El *roadmap* se tiene que empezar a trabajar una vez el producto ha conseguido introducirse de manera satisfactoria en el mercado (de nada sirve elaborar un *roadmap* de cinco años si el producto todavía no está establecido) y por su creación (y adaptación) se tiene que contar con el máximo de personal implicado en el producto.

Horizonte temporal

Un buen horizonte temporal para el *roadmap* suele ser entre seis y doce meses (dependiendo de las características del producto y del mercado al que va dirigido).

3.5. Errores comunes

La visión de un producto es un punto crucial para lograr el éxito del mismo. A continuación, exponemos algunos errores comunes relacionados con esta:

- **Visión inexistente**. Un error sorprendentemente común en muchos proyectos, que se da cuando las funcionalidades o requerimientos de un proyecto se van obteniendo de manera individual sin tener en cuenta la posible o posibles conexiones e implicaciones entre ellas. Este error se denomina muchas veces sopa de funcionalidades.

- **No adaptar**. Tener una visión del producto no asegura en ningún caso el éxito del mismo. Muchas veces es necesario adaptar esta visión a medida que se va obteniendo *feedback* de usuarios y del mercado. Seguir a ciegas la visión inicial sin tener en cuenta esta respuesta del mercado puede conseguir que el producto sea un fracaso.

- **Análisis-parálisis**. Este error se da cuando el producto no es capaz de avanzar puesto que siempre está en una continua fase de estudio de mercado o análisis. Gastar demasiado tiempo en elaborar un análisis inicial tiene el gran peligro de no llegar a tiempo a tener un producto en el mercado y quedar fuera del mismo (aunque el producto tenga el mejor diseño del mundo).

- **Sabemos lo que es bueno para nuestros clientes**. Las empresas que tienen este pensamiento y que no utilizan el *feedback* de sus usuarios para adaptar sus productos corren el riesgo de gastar tiempo y dinero en un producto que después nadie querrá.

- **Cuanto más grande mejor**. Intentar que el producto tenga de inicio un gran número de funcionalidades, que no falte absolutamente de nada y

que sea la solución perfecta desde el primer día es una opción de mucho riesgo, puesto que se combinan algunos de los problemas de los errores anteriores, como la posibilidad de llegar tarde al mercado o construir un producto que nadie quiera habiendo gastado mucho dinero y esfuerzos durante el proceso de construcción del mismo.

4. Las historias de usuario

Una **historia de usuario** es una redacción breve de una funcionalidad que por sí misma da valor al usuario.

Se compone de un título, una descripción breve sobre qué tiene que hacer la funcionalidad y unas condiciones de aceptación. Opcionalmente, se pueden añadir anotaciones y acotaciones a la historia.

Es muy aconsejable usar la siguiente estructura:

Como [papel de usuario] quiero [funcionalidad] para [objetivo]

4.1. Escribiendo historias de usuario

Una buena historia de usuario sigue las siglas **INVEST**, es decir, es:

- **I**ndependiente,
- **N**egociable,
- **V**aliosa para usuarios y clientes,
- **E**stimable (se puede medir),
- ***S**mall* (suficientemente pequeña para poder trabajar con ella), y
- **T**esteable.

4.1.1. Independiente

Las dependencias entre historias a veces derivan en problemas en la planificación y priorización. Si una historia muy poco prioritaria está vinculada con una que lo es mucho, la priorización será complicada.

Cuando dos historias están vinculadas, hacen muy difícil la estimación, puesto que surgen dudas tales como "Si desarrolláramos primero la historia A, la historia B sería más fácil de hacer y supondría menos esfuerzo, pero la historia A no es prioritaria". En la gestión ágil de proyectos siempre se hace antes lo más prioritario, así que entraríamos en un conflicto.

Cuando se presentan estos tipos de dependencias entre historias hay dos caminos para solucionarlo:

1) Combinar las historias dependientes en una más grande, pero independiente del resto.

2) Encontrar un camino diferente en la definición de estas historias.

Sugerencia

Cuando definimos historias, siempre es bueno pensar qué pasaría si en la iteración actual solo pudiéramos desarrollar la funcionalidad que estamos escribiendo.

4.1.2. Negociable

Las historias de usuario son descripciones cortas de funcionalidades, cuyos detalles deben ser negociados en conversaciones posteriores entre el cliente y el equipo de producción. Tal como hemos dicho al comienzo, una historia se compone de una breve descripción acompañada opcionalmente de anotaciones o acotaciones de la historia. Estas acotaciones se tendrán que ir negociando a lo largo del proyecto.

Acotar la funcionalidad

Las anotaciones de una historia deben acotar su funcionalidad. Es decir, si tenemos una funcionalidad de "Como usuario administrador quiero poder generar un documento Word con el listado de nombres de los productos del sistema", una anotación como "la funcionalidad también debe sacar el listado en PDF" no acota ni detalla la funcionalidad, la aumenta. Una anotación posible sería "el documento Word tiene que estar en el formato oficial de la empresa".

En muchos proyectos, se confunde dar más detalles con dar más precisión. No hay que entrar en la precisión excesiva, si la historia se puede estimar y priorizar se pueden dejar los detalles precisos para conversaciones posteriores.

Podemos ver una historia como un recordatorio de tener una conversación entre el equipo de producción y el cliente. Donde la descripción indica sobre qué funcionalidad tenemos que hablar y las anotaciones serían los detalles a los que se han llegado en estas conversaciones.

4.1.3. Valiosa

La mejor manera de asegurarse de que una historia es valiosa para el cliente es dejarlos escribir a ellos las historias. Es habitual que los clientes se muestren incómodos con esta propuesta, ya que están acostumbrados a que todo lo que escriben es un contrato y se puede usar en su contra ("Como no escribiste que querías la validación del DNI..."). En la gestión ágil de proyectos no se busca esta manera de "pasar la pelota". Es muy extraño que detalles importantes no se comenten en las conversaciones entre cliente y equipo de producción.

Si logramos tener a un cliente que confía en nosotros y se involucra, tenemos mucho ganado en la definición de historias.

4.1.4. Estimable

Hay tres razones comunes por las que una historia de usuario no se puede estimar:

1) poco dominio por parte del equipo de producción del lenguaje del cliente;

2) poco dominio técnico por parte del equipo de producción sobre la tecnología que se va a usar; y

3) la historia es demasiado grande.

Si el equipo de producción no entiende la historia tal cual se ha escrito, la solución es celebrar reuniones entre el equipo de producción y el cliente. En la gestión ágil de proyectos no está permitido ahorrarnos conversaciones ni suponer o interpretar conceptos. Si el equipo de producción no está seguro de lo que tiene que hacer, se tiene que hablar para entenderlo.

Si el problema es tecnológico, la solución es reservar uno o dos programadores para un *spike*.

Un ***spike*** es un breve experimento para aprender cómo funciona una tecnología y qué implicaría en nuestro proyecto usarla.

Durante el *spike*, se pide a los programadores que experimenten y aprendan todo aquello necesario solo para estimar la funcionalidad, no que la desarrollen.

Nota

Es recomendable que un *spike* sea limitado por un *time-box*, o porción de tiempo, donde, si se sobrepasa y no se ha conseguido nada, podemos pensar en descartar la tecnología seleccionada.

Si el problema es que la historia de usuario es demasiado grande para estimarla con cierta precisión, la única salida es dividirla, más adelante aprenderemos cómo dividir las historias de usuario.

Historia de usuario pequeña

Puede ser que una historia de usuario como la siguiente:

"Como usuario visitante de la web, tengo que poder registrarme para entrar en el sistema".

sea suficientemente pequeña, ya que la tecnología empleada da herramientas prehechas para afrontar esta tarea.

Aunque una historia sea demasiado grande, a veces son útiles como épicas. Las **historias épicas** son recordatorios de las grandes partes del sistema que tienen que ser desarrolladas.

4.1.5. *Small*

El tamaño ideal de las historias varía en función del equipo, las capacidades y la tecnología empleada.

Si la historia se tiene que dividir, se deben tener en cuenta las consignas siguientes:

1) Dividir historias por su composición

Tenemos que pensar en el flujo de acciones que el usuario va a ejecutar en toda esta funcionalidad, por ejemplo "Como usuario administrador quiero poder administrar los productos del sistema" se podría dividir en "ver una lista de productos", "crear un producto", "eliminar un producto", "editar un producto". Tener en mente las siglas **CRUD** (***create***, ***read***, ***update***, ***delete***) nos puede ayudar en esta tarea.

2) Dividir historias complejas

Esto es menos obvio. Son aquellas historias que son muy complejas técnicamente o que no se han hecho nunca y no sabemos cuánto nos pueden costar, la división recomendada en este caso es **investigar-realizar-refactorizar**. Podemos crear una historia de usuario que sea "Investigar cómo hacer un registro de usuario en la nueva tecnología" y se asignaría un *time-box* limitado, el otro "realizar la funcionalidad" y otra de "refactorizar[3] la funcionalidad".

[3]Refactorizar quiere decir revisar el código programado para poderlo hacer más rápido, mantenible a largo plazo y comprensible para otros programadores, entre otros.

Historia de usuario de investigar

Es importante que si se decide definir una historia de usuario de investigar tenga un fin. Puede ser un documento con conclusiones, una versión tosca de la funcionalidad pero con los detalles primordiales que se han investigado. Si no indicamos una finalidad esta historia se puede convertir en una pérdida de tiempo.

Es posible que nos encontremos con historias demasiado pequeñas, como arreglar algún *bug* o corregir literales. Detectaremos una historia demasiado pequeña enseguida, ya que el equipo de producción dirá que es más rápido hacerla que escribirla.

Hay algunas historias que hacen referencia a todo el proyecto, por ejemplo "La web tiene que ser rápida". Esto no es una historia, es una *constraint* y las ***constraints*** son condiciones que deben cumplir todas las historias que se desarrollen y en general todo el proyecto. "Ninguna página de la web tiene que tardar más de tres segundos en cargarse" sería un buen ejemplo de *constraint*.

Las *constraints* están presentes en todo el proyecto y las deben tener siempre en cuenta el equipo de producción antes de finalizar una historia de usuario.

4.1.6. Testeable

¿Cómo sabemos si una historia está acabada si no se puede probar? Una historia debe tener ciertas condiciones de validación que justifiquen que la historia está acabada. Una historia se tiene que poder probar que funciona.

4.2. Los beneficios de las historias de usuario

Las historias de usuario comportan, entre otros, estos beneficios principales:

1) Promueven la comunicación verbal

Los seres humanos tienen una gran tradición oral. Cuentos y leyendas han sido transmitidos oralmente entre generaciones, pero en algún momento, en torno al año 1970, se empezó a sentir la necesidad de registrarlo todo por escrito. Hemos cambiado el compartir conocimiento por compartir documentos. Parece que dejar algo por escrito evita desacuerdos y malentendidos, pero no es así.

Comunicación verbal

Si en un restaurante vemos que en el primer plato sirven bacalao o lomo con guarnición, ¿significa que tanto la carne como el pescado vienen acompañados de guarnición? Si ya nos cuesta leer el menú en el restaurante, imaginad la definición de una funcionalidad de un proyecto de producción multimedia.

La palabra *debería* no se tiene que usar nunca. ¿Qué quiere decir "el sistema debería avisar si el usuario ha introducido un DNI incorrecto"? ¿Significa que el sistema puede no avisar?

Cuando cliente y proveedor usan la comunicación oral, se produce un *feedback* constante, del que se extrae mucha más información que por escrito. Promueve el aprendizaje y el entendimiento mutuo.

2) Son comprensibles por todo el mundo

Como una historia de usuario está escrita en lenguaje natural, es comprensible tanto por usuarios como por el equipo de producción y por todo el que participe en el proyecto. Promueve la participación y la opinión de todo tipo de perfiles, tanto diseñadores como programadores o comerciales, entre otros. No se restringe la opinión a ninguna persona que lea la funcionalidad.

3) Tienen el tamaño justo para planificar

Ni demasiado grandes ni demasiado pequeñas, una historia de usuario tiene el tamaño justo para que tanto cliente como desarrolladores se sientan cómodos con ella, tanto para definirla como para desarrollarla, probarla o demostrarla al cliente en una demo.

4) Funcionan muy bien en el proceso de producción iterativo

En la gestión ágil de proyectos, donde se trabaja de manera empírica, iterativa e incremental, las historias de usuario son la mejor unidad mínima de funcionalidad para desarrollar. Iteración tras iteración se pueden usar para ir construyendo el proyecto multimedia final.

5. Trabajando con la pila de producto

La **pila de producto** es uno de los principales artefactos de las metodologías ágiles, contiene la lista priorizada de todo aquello (requerimientos y funcionalidades o ambos) necesario para finalizar un proyecto. Esto hace que la pila de producto desempeñe un papel importantísimo en la gestión y conceptualización del producto, puesto que refleja en un momento dado cómo se prevé que será el producto.

Es importante destacar que la pila de producto es un elemento vivo, que irá cambiando a medida que el proyecto vaya avanzando y la gente involucrada tenga cada vez más información y conocimiento sobre el mismo.

5.1. Organizar la pila de producto

Tal como ya se ha comentado, la pila de producto es un elemento vivo, no es una lista de requerimientos hecha al inicio de proyecto y de la que uno se puede olvidar durante el desarrollo del mismo. Esto quiere decir que la pila de producto se tiene que ir disponiendo a medida que el proyecto va avanzando. Normalmente, esta disposición tiene lugar por los motivos siguientes:

- se descubren nuevos elementos y se añaden a la pila de producto;
- elementos existentes en la pila de producto cambian o se tienen que eliminar;
- se reprioriza la pila de producto y elementos que antes no eran prioritarios pasan a serlo y al revés, y
- se detallan con más profundidad elementos que estaban descritos en alto nivel.

5.1.1. ¿Quién ordena la pila de producto?

El hecho de que la pila de producto esté en un estado correcto (correctamente priorizada, con los elementos más prioritarios estimados) es responsabilidad del jefe de producto (*product owner*), pero hay que resaltar que es un proceso colaborativo y que requiere la participación de las diferentes personas involucradas en el proyecto (como clientes, usuarios o equipo).

Haciéndolo de este modo se estimula el diálogo entre el ***product owner*** y los diferentes **equipos** que participan en el proyecto al lograr que haya más unidad y colaboración entre ellos y se ayuda a eliminar ambigüedades sobre el producto.

5.1.2. ¿Cuándo se ordena la pila de producto?

La pila de producto se puede ordenar en diferentes momentos y a diferentes intervalos. Es responsabilidad de cada equipo o grupo de proyecto decidir cuándo es el mejor momento para llevarlo a cabo.

Algunos de los momentos más utilizados para hacerlo son los siguientes:

- al finalizar una iteración;
- en las demostraciones de producto;
- de forma semanal, o
- después del *daily meeting*.

Lo más importante es establecer una rutina regular para hacerlo, puesto que así se puede asegurar en todo momento que la pila de producto refleja realmente las necesidades del producto/proyecto en el que se está trabajando, requisito indispensable para que el proyecto llegue a buen puerto con éxito.

5.2. Planificar una *release* (o liberación)

Muchos de los proyectos de producción multimedia piden liberar versiones cada cierto tiempo. Estas versiones suelen ser un conjunto de errores arreglados más un conjunto de nuevas funcionalidades implementadas.

¿Cuándo se entrega una *release*?

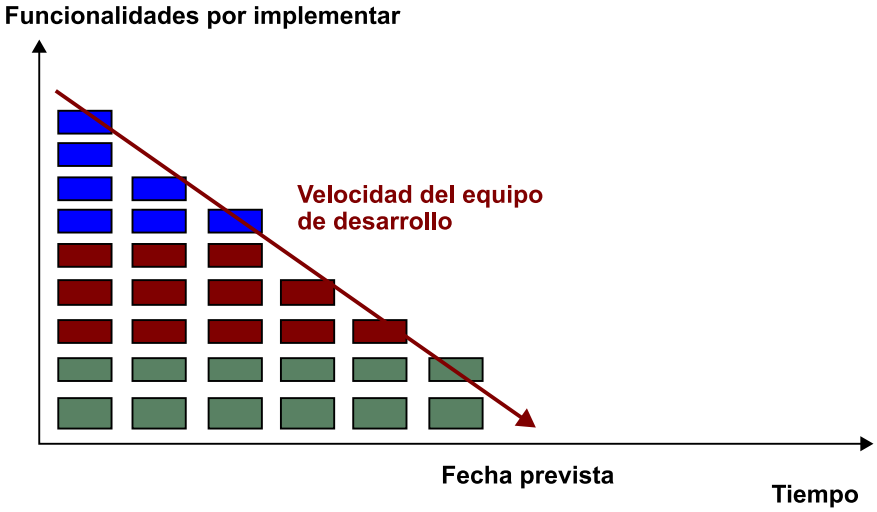
Dependiendo del proyecto, una ***release*** se libera cuando el equipo de desarrollo acaba un conjunto de funcionalidades comunes.

En la gestión ágil de proyectos, partiendo de un *backlog* estimado y priorizado y partiendo de una velocidad de producción (cantidad de historias producidas por iteración), podemos definir plazos en función de las historias épicas; por ejemplo: "Al ritmo actual de producción, en cuatro o cinco iteraciones el equipo de producción habrá acabado las funcionalidades correspondientes a la gestión de productos, y en dos o tres iteraciones más, el registro de usuarios".

De todos modos, lo más probable que pase es que las fechas sean fijas, en este caso es fácil definir cuándo tiene que salir una *release* (en esa fecha), y lo complicado es decidir cuántas funcionalidades estarán acabadas en esa fecha. A partir del *backlog* estimado y priorizado y la velocidad de desarrollo, podemos saber si llegaremos a la fecha mencionada con las funcionalidades requeridas.

5.3. *Product burndown*

El ***product burndown*** es la gráfica que presenta funcionalidades pendientes frente al tiempo que queda para entregar una *release*.



En el gráfico de ejemplo vemos que, para la historia épica azul, hay que hacer cuatro historias de usuario, para la roja tres y para la verde dos. Gracias a la velocidad del equipo de desarrollo, sabremos para qué fecha tendremos las épicas hechas y, por lo tanto, para cuándo podremos tener la *release*.

La velocidad es una herramienta para el *product owner* que sirve para planificar *releases*. Es un dato muy difícil de conseguir puesto que necesitamos un equipo estable y con un buen historial de velocidades anteriores.

5.4. Priorización de historias de usuario

Hay muchas maneras de priorizar funcionalidades y todas se basan en dos conceptos:

- hacer primero lo que más riesgo tenga;
- hacer primero lo que más impacto tenga en el resto de funcionalidades;
- según los deseos del cliente, y
- según los deseos tecnológicos de los desarrolladores.

En la gestión ágil de proyectos de producción multimedia se priorizará siempre según los deseos del cliente. Siempre.

Ahora bien, los deseos del cliente pueden variar según las estimaciones de las funcionalidades, es decir, si el cliente prioriza mucho una funcionalidad sobre otra, pero el equipo de producción le dice que esta funcionalidad puede tardar semanas en tenerla, probablemente repriorice sus peticiones.

Priorizar funcionalidades

Un ejemplo reciente que se ha dado en la empresa donde trabaja el autor de este módulo es una funcionalidad que pedía hacer: que en un producto multimedia, el formulario de registro cambiara de campo cada vez que el usuario hiciera clic en la tecla **Intro** y no en la tecla de **tabulación**.

Para el cliente era algo casi imperceptible, pero cuando le dijimos que era un cambio que requería un retraso en otras funcionalidades también importantes, decidió que no era tan prioritario como él pensaba.

Para poder priorizar, antes, es necesario estimar en puntos de historia.

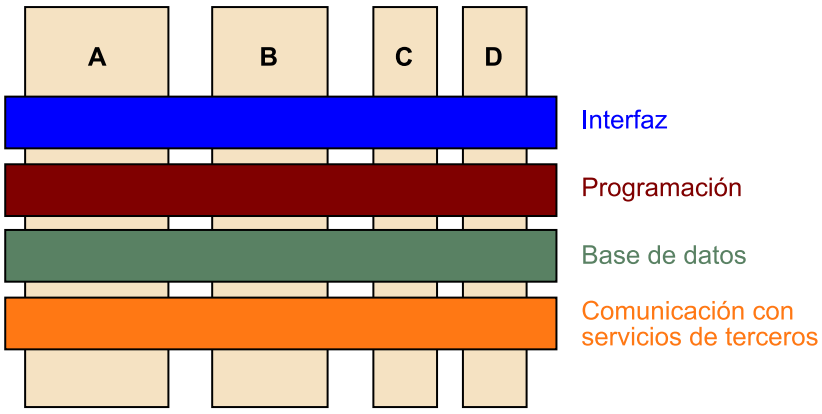
5.5. Priorización de las historias con más riesgo

En general, en la gestión ágil de proyectos de producción multimedia se priorizan las funcionalidades con más valor para el cliente que las que tengan más riesgo, puesto que prestar atención a funcionalidades con riesgo tecnológico pero poco prioritarias puede acabar en una pérdida de tiempo si el cliente decide a medio proyecto que ya no necesita la funcionalidad con riesgo.

A veces, es necesario comunicarle la necesidad de priorizar funcionalidades que hagan un corte transversal en toda la arquitectura de la aplicación. En esos casos, hay que definir conversaciones con el cliente para hacerle entender el esfuerzo que comportan estas funcionalidades.

Si estamos trabajando para un cliente que necesita una web para registrar reservas de hotel, tendríamos que comunicarle la necesidad de priorizar la historia épica A (reserva de hoteles) antes que la B (registro de usuarios).

El siguiente diagrama muestra las capas por las que atraviesan las funcionalidades.



6. Crear la pila de producto: *user role modeling*

Como ya se ha indicado en puntos anteriores, la creación de la pila de producto no es una actividad que tenga lugar únicamente al inicio del proyecto, sino que una vez creada se tiene que ir ordenando, añadiendo nuevos elementos, modificando los ya existentes, eliminando aquellos que ya no tienen cabida, entre otros.

Para empezar a crear la pila de producto, se tiene que partir de la visión del producto. A partir de esta, hay que intentar definir el mínimo número posible de elementos para obtener un producto funcional. Hay que evitar intentar definir absolutamente todos los elementos de la pila de producto en el inicio del proyecto, puesto que sería una pérdida de tiempo y dinero, y solo se tiene que intentar tener claro lo que es más prioritario en el momento actual (sin perder de vista la visión y lo que se quiere conseguir a medio/largo plazo).

Una buena técnica para ayudar a crear el *product backlog* es la que se denomina ***user role modeling***.

En las metodologías ágiles, es común utilizar el formato de historias de usuario como formato para describir los elementos de la pila de producto. Esta técnica, aunque se pueda utilizar con cualquier otro formato, está muy orientada a pilas de producto que trabajan con historias de usuario.

La idea principal de esta técnica es intentar reunir los requerimientos del producto o proyecto utilizando como base las diferentes tipologías de usuarios que lo utilizarán o lo consumirán.

Por ejemplo, si el producto es una aplicación web para reservar libros de una biblioteca, podríamos pensar en los siguientes tipos de usuario:

- Aquel usuario que solo utiliza la página de manera muy esporádica, por ejemplo, cuando necesita un libro para hacer un trabajo para la escuela o universidad.
- Aquel usuario que utiliza la página de manera recurrente y que no tiene ninguna preferencia de género o tipo de libros. Podríamos pensar que es una persona a la que le gusta la lectura en general.
- Aquel usuario que solo está interesado en un determinado tipo de libros. Por ejemplo, alguien aficionado a la historia o alguna persona que por trabajo necesita acceso a determinados tipos de libros de manera regular.
- Y por último un usuario administrador, aquel que accede a la aplicación para desempeñar tareas de gestión de la misma.

Tener presente estos cuatro tipos de usuarios del ejemplo (o todos los que sean necesarios) ayudará al equipo a descubrir nuevos requerimientos o a pensar en diferentes situaciones que no se tendrían presentes si se empieza a trabajar en el producto sin tener hecho este ejercicio.

6.1. Etapas del *user role modeling*

Para lograr tener un buen conjunto de tipologías de usuarios, se deben seguir los pasos siguientes:

1) *Brainstorming* inicial

De manera colaborativa, se lleva a cabo una sesión de *brainstorming* para intentar obtener el máximo número posible de tipologías de usuario. Cada participante debe coger un conjunto de tarjetas (o *post-its* o cualquier elemento similar) y tiene que ir escribiendo todos los posibles tipos de usuarios que le vengan a la cabeza. Esta parte tiene que durar unos quince minutos aproximadamente.

Es importante intentar que cada papel o tipología de usuario represente a un único usuario. Es decir, se tienen que intentar evitar roles del tipo "la compañera", que incluyen en su interior a más de un posible usuario o consumidor del producto.

2) Organizar el conjunto de roles

Cuando el *brainstorming* ya ha acabado, se colocan todas las tarjetas en un panel y se sobreponen aquellas cuyos roles compartan perfil. Puede ser que algunos roles se solapen por completo o que algunos estén completamente aislados del resto.

3) Consolidación de roles

Todos los roles que estén muy solapados se pueden consolidar en uno solo. Es posible que en este punto haya discusiones en el equipo, de estas discusiones tienen que surgir decisiones de las diferencias entre los roles tratados.

4) Refinación de los roles

Una vez se han consolidado los roles, debemos tener una comprensión básica de cómo se relacionan unos roles con los otros y con nuestro sistema. Para llevar a cabo esta tarea es útil definir **atributos**. Cualquier información que distingue a un usuario de otro puede ser entendida como atributo. Para definir atributos tenemos que pensar en:

- la frecuencia con la que usará el producto;
- el nivel de experiencia en multimedia;

- el objetivo básico de este usuario; y
- el lugar desde donde el usuario accederá a la aplicación (móvil o fijo).

Definir *personas*

Crear una *persona* para un rol es imaginarnos a un individuo completo para este rol. Todos los miembros del equipo se tienen que imaginar a la *persona* del mismo modo, le tienen que poner cara. Si es necesario, usar técnicas de búsqueda demográfica y estadísticas para definir exactamente ese *target*. Buscad una fotografía, ponedle nombre y tendréis a una *persona*. Cuando a un rol le ponemos nombre, por ejemplo, María, a partir de ese momento vuestras historias de usuario se pueden referir a María como ejecutora de la acción y todo el equipo sabrá perfectamente qué experiencia de usuario esperará María de vuestra aplicación multimedia.

Persona

En inglés se utiliza la palabra *persona* para referirse a este personaje ficticio pero con unas características y una imagen perfectamente definidas. Por eso, usaremos la palabra *persona* y el plural *personas*, para diferenciarlo de la palabra castellana.

Personajes extremos

Cuando se está definiendo el producto, es muy útil imaginar usuarios extremos, que pongan a prueba el producto multimedia. Podemos imaginar a una abuela usando la aplicación: ¿necesitará ampliación del texto mostrado? O al presidente del país, ¿necesitará capacidad extra de almacenamiento en la agenda? Puede ser que estos personajes extremos generen funcionalidades poco prioritarias, pero puede ayudarnos a encontrar funcionalidades en las que ningún miembro del equipo habrá pensado.

7. Estimación de historias de usuario

En todos los proyectos de producción multimedia queremos saber cuánto nos va a costar desarrollar lo que se ha definido y tenemos que estimar y medir el esfuerzo necesario para llevar a cabo esta tarea. Una buena medida sería aquella que cumpliera ciertos requisitos, por ejemplo:

- que nos permita cambiar la estimación cuando nos llegue información nueva de la historia;
- que funcione tanto para historias épicas como para historias de usuario;
- cuya estimación no comporte demasiado tiempo invertido; y
- tolerante a imprecisiones.

7.1. Los puntos de historia

En la gestión ágil de proyectos multimedia, la opción más interesante es estimar en puntos de historia.

Un **punto de historia** es una unidad de esfuerzo relativa.

Se puede entender de muchas formas, como días ideales (días sin ningún tipo de interrupciones), unidades abstractas de esfuerzo o cualquier tipo de relación que encaje con el equipo de producción.

Los puntos de historia dan respuesta a la pregunta de cuánto esfuerzo se requiere para llevar a cabo una historia épica o de usuario.

Los puntos de historia son relativos a otros, es decir, sabemos que una historia son cinco puntos de historia porque es más del doble que una de dos y sabemos que una es de dos porque es el doble que una de un punto. Por lo tanto, el punto de partida al empezar a usar puntos de historia es definir qué representa un punto, ¿un día ideal? ¿Una semana ideal? Depende del equipo. Una vez está definido qué significa un punto, el resto de estimaciones se hacen por ***planning poker*** o por **triangulación**.

La **estimación** siempre la define el equipo de producción, puesto que son ellos los que implementarán la funcionalidad y saben cuánto cuesta hacer las cosas. En la gestión ágil de proyectos multimedia, siempre estima el equipo en equipo.

7.1.1. *Planning poker*

El ***planning poker*** es una técnica de estimación de historias donde el equipo se reúne y tiene a su disposición las historias de usuario. El equipo formula preguntas al cliente para aislar dudas. Una vez las historias están claras, cada miembro del equipo dispone de ocho cartas con diferentes números: 1, 2, 3, 5, 8, 13, 20 y 40.

Cartas de *planning poker*

Cada miembro del equipo escoge una carta que representa los puntos de historia que él cree que cuesta la historia y la deja boca abajo en la mesa, todo el equipo muestra a la vez su estimación.

En ese momento surgirán divergencias entre el equipo, la persona que ha puesto la carta más baja explica sus motivos y el miembro del equipo que ha puesto la carta más alta también. De esta conversación, salen formas alternativas de ejecutar la tarea y alineación entre los miembros del equipo. No se trata de burlarse de quien ha puesto más o menos puntos de historia, sino de aprender del motivo por el que lo ha hecho.

Una vez ha finalizado el debate, se vuelve a estimar. En esta segunda vuelta, lo más probable es que la mayoría coincida.

El *planning poker* fomenta la estimación en equipo, la conversación y la propuesta de formas alternativas de llevar a cabo las tareas.

7.1.2. Triangulación

Una manera alternativa de asignar puntos de historia a las historias de usuario de un proyecto de producción multimedia es distribuir las historias de la que requiere menos esfuerzo a la que más, estableciendo agrupaciones de poco esfuerzo, esfuerzo medio y esfuerzo alto. Durante esta fase de agrupación, el equipo discute sobre si una historia es más, menos o igual de compleja que las demás.

Una vez ordenadas, podemos asignar un punto de historia a la más sencilla e ir escalando en orden desde la uno, y asignando los puntos de historia según la sucesión de Fibonacci (1, 2, 3, 5, 8, 13).

La sucesión de Fibonacci es una sucesión exponencial de números donde cada número es el resultado de la suma de los anteriores. Estimar las historias en puntos de historia que sigan Fibonacci es positivo por dos motivos:

1) Representa a la perfección la filosofía por la que, estimando cosas muy grandes, el margen de error es muy amplio. Una historia grande se puede evaluar en 8 o 13, pero nunca en 12 o 13, puesto que cuanto mayor sea la historia más complicada es de estimar.

2) Ayuda a estimar historias grandes guiándose por las anteriores: "¿formar esta historia de 8 puntos es lo mismo que hacer esta de 5 más esta de 3?"

8. Conclusiones

Ser *product owner* de un producto multimedia en la gestión ágil de proyectos requiere más que el conocimiento de cómo escribir una historia de usuario o gestionar un *backlog* de funcionalidades. Los propietarios de productos profesionales necesitan tener un conocimiento concreto de todo lo que impulsa el valor de sus productos.

Hoy en día, una organización de producción multimedia debe ser ágil. La gestión ágil de proyectos ya no es una opción. Tienen que ser capaces de ofrecer productos nuevos o mejorados, y sistemas como el dictado por los clientes, la competencia y las presiones comerciales, a medida que operan dentro del entorno en el que se producen los cambios, tienen que ser flexibles sin perder de vista su propósito.

En las empresas multimedia, los clientes requieren una mejora continua en tiempo real. A medida que se van cambiando prioridades y requisitos durante el proceso de producción, se tiene que poder ser a la vez predecible y eficiente, con un riesgo controlado.

El objetivo tiene que ser la liberación de producto tan rápido como sea necesario y no más rápido de lo que los clientes pueden absorber. Obtener el *feedback* continuo del cliente en cada fin de iteración para así cerrar el círculo de mejora continua que toda gestión ágil de proyectos multimedia tiene que tener.

Bibliografía

Beck, Kent; Cohn, Mike (2004). *User Stories Applied: For Agile Software Development*. Addison-Wesley. Disponible en: http://www.amazon.es/user-stories-applied-development-signature/dp/0321205685/ref=sr_1_fkmr2_1?s=foreign-books&ie=UTF8&qid=1336041612&sr=1-1-fkmr2.

Pichler, Permanece (2010). *Agile Product Management with Scrum: Creating Products That Customers Love*. Addison-Wesley. Disponible en: http://www.amazon.es/agile-product-management-scrum-addison-wesley/dp/0321605780.

Rasmusson, Jonathan (2010). *The Agile Samurai: How Agile Masters Deliver Great Software*. Disponible en: http://www.amazon.es/the-agile-samurai-pragmatic-programmers/dp/1934356581