



Seminario 2 – Historias de Usuario





¿Qué son las HU?

“The story is the unit of functionality in an XP project. We demonstrate progress by delivering tested, integrated code that implements a story. A story must be understandable to customers, developer-testable, valuable to the customer, and small enough that programmers can build half a dozen in an iteration.”

Beck, K.; Fowler, M.: “*Planning Extreme Programming*”.



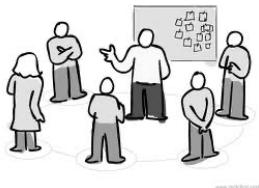
¿Qué son las HU?

- ✖ Es una descripción simple de una tarea concreta que aporta **valor al usuario** o al **negocio**.
- ✖ Se pueden ver como una corta declaración de intención que describe algo que el sistema necesita hacer para el usuario.
- ✖ Suelen obtenerse de las reuniones con los “**stakeholders**”.



¿Qué son las HU?

- ✖ Las historias tienen que ser simples, cortas, fáciles de entender, de aceptar y recordar, sin que tengan que escribirse todos sus detalles.
- ✖ Se describen usando tarjetas (descripción escrita en lenguaje del negocio que identifica y recuerda el requisito y ayuda a la priorización).



¿Qué son las HU?



Cortesía: Jacopo Romei (Vía Flickr)



© dreamstime.com

ID 155290187 © Comzeal



Objetivo de las HU

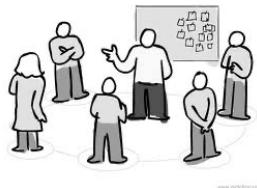
- ✖ El objetivo de las HU es principalmente, lograr la interacción con el equipo y con el usuario, por encima de documentar los requisitos que hay en el sistema.
- ✖ Hay que justificar el beneficio de todo lo que aparece en una HU.



¿Qué no son las HU?

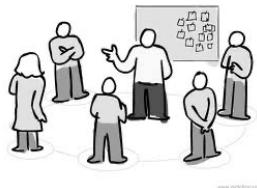
No es un requisito del sistema (**algo que el sistema deba hacer**),

hay que verlo más como algo negociable (**necesitaríamos hacer algo similar a esto**)



Historias de usuario / Casos de Uso

CONCEPTO	CASOS DE USO	HISTORIAS DE USUARIO
Objetivo	Modelar la interacción entre un 'actor' y el Sistema	Redactar una descripción breve de una funcionalidad tal y como la percibe el usuario
Estructura	Texto detallado donde se sigue una plantilla predefinida a completar con conceptos técnicos (objetivo, resumen, actor, evento disparador, extensiones, etc.)	Corta y consistente en una o dos frases escritas en el lenguaje del usuario.
Planificación	No se utilizan para planificar	Se utilizan para planificar
Agilidad	Requieren tiempo para análisis y la redacción de plantillas predefinidas	Se pueden escribir en pocos minutos
Comprensión	Suelen ser de difícil comprensión, incluso para personal técnico	Fáciles de leer y comprender
Mantenimiento	Suelen pertenecer a documentos con cientos de páginas. Difíciles de mantener.	Muy fáciles de mantener
Comunicación	Modelo textual asociado con diagramas: todo tiene que estar escrito	Basada en la comunicación verbal y orientada a la colaboración y discusión para clarificar detalles



Historias de usuario / Casos de Uso

CONCEPTO	CASOS DE USO	HISTORIAS DE USUARIO
Soporte	Escritos en documentos con el objetivo de que estos sean archivados como documentos de referencia.	Escritas en tarjetas (teóricas o reales) con el objetivo de que sean usadas directamente
Duración	Puede ser implementada en varias iteraciones	Debe ser implementada y probada en una única iteración
Autores	Definidos por 'intérpretes' (analistas, consultores, etc.)	Posibilidad de ser definidas por usuarios y clientes
Pruebas	La definición de pruebas se redacta en documentación separada	Contienen 'Pruebas de Aceptación' en el reverso de la tarjeta
Contexto	Proporcionan una visión más general del Sistema y la integración en él.	Proporciona una visión menos obvia, por eso las pruebas y el <i>feedback</i> de los usuarios son tan importante en las metodologías agiles.
Metodología	Asociado con RUP	Asociado con Programación Extrema (aunque pueden usarse en RUP)

<http://www.agile-ux.com/2009/01/23/use-cases-user-stories-so-precious-but-not-the-same/>



¿Quién escribe las HU?

- ✗ **XP**: El cliente.
- ✗ **SCRUM**: El Product Owner (con ayuda de clientes, stakeholders, equipo de desarrollo).
- ✗ **En la realidad**: Cualquier miembro del equipo de desarrollo con conocimiento suficiente en el dominio del problema.



¿Quién escribe las HU?

- ✖ Los que pueden expresar las **necesidades reales** del sistema.
- ✖ Participar en las **reuniones** para detallarlas y priorizarlas.

(Customer Team) ---- (Stakeholders)

- Clientes reales (usuarios).
- Product Manager.
- Probadores.
- Diseñadores de interacción.



¿Quién escribe las HU? (2)

- ✗ Hay que identificar y organizar a los usuarios del sistema. (Roles o personas)

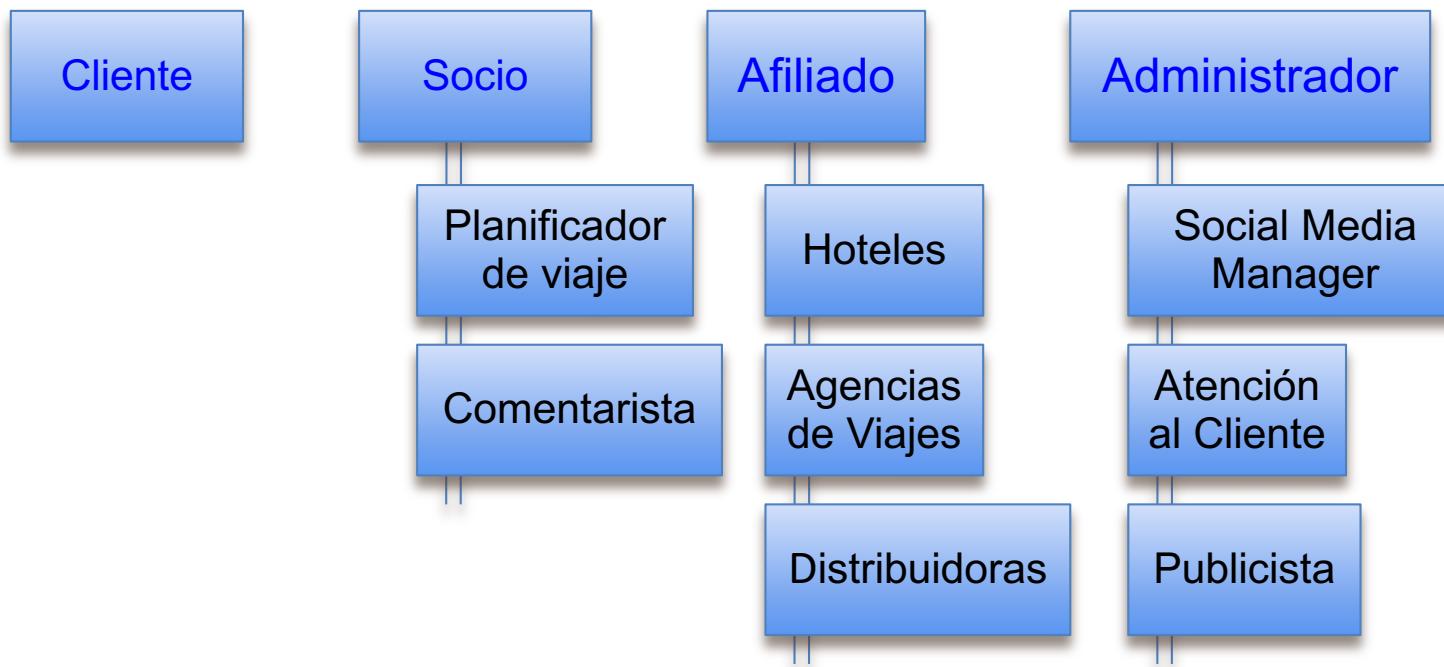
Roles del sistema:

- ✗ Podemos usar una sesión de “Brainstorming” para definirlos.
- ✗ Definición de los roles. (Frecuencia de uso del software, objetivos que se plantean, forma habitual de acceso, etc.)



¿Quién escribe las HU? (3)

- ✗ Tabla de roles del sistema (Agencia de Viajes)





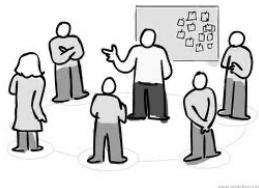
Formato de una HU

**Cómo [rol], Quiero [característica/objetivo] ,
Para [valor de negocio/beneficio]**

[Cómo:] Quién es el usuario o grupo de usuarios que participan en la historia, normalmente es un “rol de usuario”, el que usará la funcionalidad” descrita por esta historia.

[Quiero:] Es la actividad, el eje de la historia, qué hace el usuario en la historia.

[Para:] Es el propósito de la historia, la meta que quiere alcanzar el usuario al ejecutar la historia.



Ejemplos: Página Web de viajes

Como cliente, quiero reservar una habitación de un hotel

Como un cliente, quiero cancelar una reserva previa

Como un planificador de viaje, quiero ver fotos de un hotel

Como un cliente, quiero comparar las características y precios de dos hoteles

Como un planificador de viaje, quiero conocer opiniones de otros viajeros

Como cliente, quiero buscar un hotel cerca del mar

Como un cliente, quiero planificar un viaje para estas vacaciones de Navidad

Como un cliente, quiero filtrar los hoteles en los que admiten perros



Ejemplos de HU

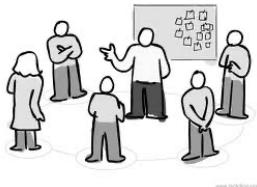
- ✖ Un usuario puede buscar un libro insertando cualquier combinación de título o autor.
- ✖ Un usuario puede poner los libros en el carrito y sucesivamente comprarlos.
- ✖ Un usuario puede escoger acompañar el envío de un libro con su propia dedicatoria.
- ✖ Un supervisor puede ver un informe de las ventas realizadas en el día.
- ✖ Un usuario puede enviar información acerca de un trabajo a una amiga vía mail.



Elementos de una HU. “CCC”

- × **Tarjeta (Card)**: Representa las 2-3 sentencias usadas para describir la intención de la HU.
- × **Conversación**: Detalles sobre la HU obtenidos de la conversación con el usuario o cliente.
- × **Confirmación**: Cómo el equipo confirmará que la HU ha sido implementada (Test de aceptación).

Ron Jeffries

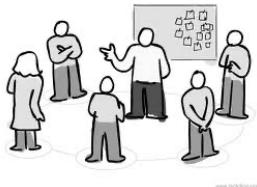


Información en una HU

- ✗ Es una descripción corta que no incluye detalles.
- ✗ Los detalles se trabajan durante la etapa de "Conversación".
- ✗ Una tarjeta con demasiados detalles limita la conversación con el cliente.

Un cliente puede pagar un viaje usando una tarjeta de crédito.

Nota: Se pueden aceptar Visa, Master Card y American Express, discutir si se aceptan otras más.



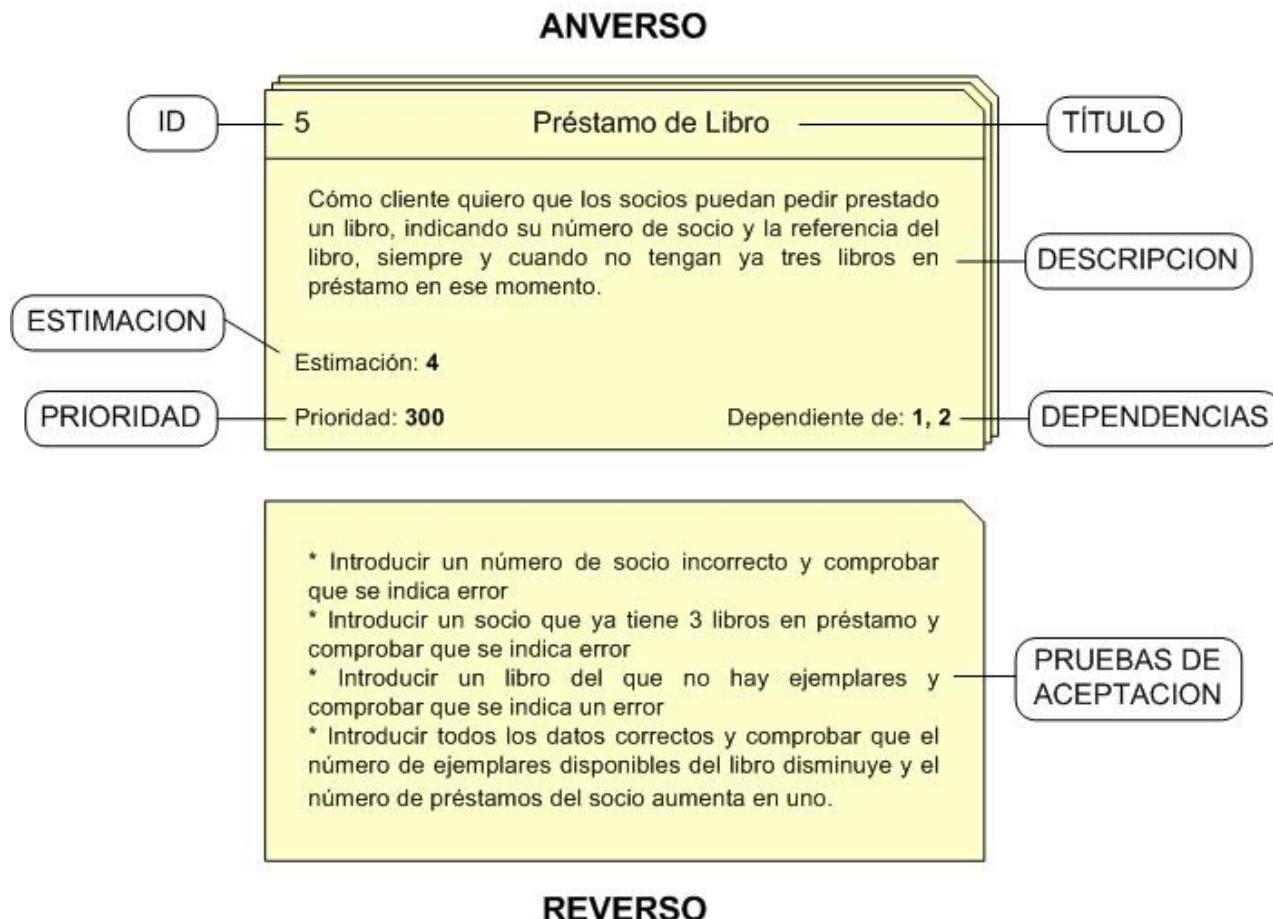
Narración de la HU

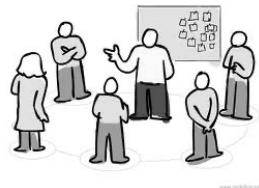
¿Dónde están los detalles de las HU?

- ✖ Se describen en la **conversaciones** entre el equipo y los usuarios.
[Conversación]
- ✖ Se delimita la parte a abordar por la HU usando las “**pruebas de aceptación**”
[Confirmación]
 - Flujos alternativos en la actividad.
 - Límites de aceptación.
 - Pruebas de aceptación por parte de los clientes.



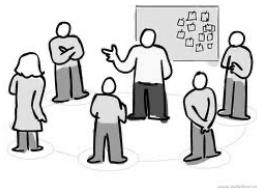
Tarjeta - Historias de Usuario





Ejemplo de tarjeta con una HU

Historia de Usuario	
Número: 3	Nombre: Elaboración de Presupuesto
Usuario: Desarrollo Empresarial	
Modificación de Historia N°: 0	Iteración Asignada: 2
Prioridad en Negocio: Media (Alta/Media/Baja)	Puntos estimados:
Desarrollador Encargado: Luis Ariel Castillo Estupiñán	
Descripción: Elaborar el presupuesto para la ejecución del evento. Depende de la disponibilidad presupuestal estipulada para la dependencia para el año vigente.	
Observaciones: El presupuesto está estipulado para la dependencia para el año, lo cual implica que para cada evento se tiene un máximo de recursos dependiendo de la naturaleza del mismo.	



Ejemplo de tarjeta con una HU

Historias de usuario

Como [cliente habitual], **quiero** [ver productos relacionados] **para** [ver si hay otros productos que me puedan interesar]

Condiciones de completitud

- *Los productos estarán ordenados por valoración y margen de beneficio.*
- *Cuando el usuario haga clic en un producto, se desplegará el detalle.*
- *Etc.*

Prioridad

70

Coste

5

http://farm1.static.flickr.com/55/147874576_8a453079f3.jpg



Buscar detalles de la HU (Conversación)

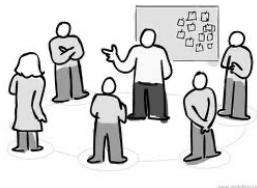
Como cliente. Quiero cancelar
una reserva previa

- ✗ ¿Qué coste le va a provocar al cliente la cancelación?
- ✗ ¿Todos los clientes van a tener el mismo coste?
- ✗ ¿Se puede cancelar en cualquier momento?
- ✗ ¿Se puede realizar una cancelación total o parcial?
- ✗ ¿A quién se notifica la cancelación?
- ✗ ¿Es lo mismo para todos los hoteles?



Pruebas de aceptación (Confirmación)

- ✖ Se usan para expresar muchos de los detalles que se obtienen de la conversación entre clientes y desarrolladores.
- ✖ Se usan a dos niveles:
 - Características que se añaden a la tarjeta de la HU.
 - Pruebas que se usan para demostrar que la HU ha sido correctamente desarrollada.
- ✖ Los programadores pueden usarlos para mejorar el desarrollo ya que se describen antes de codificar nada.



Pruebas de aceptación (Confirmación)

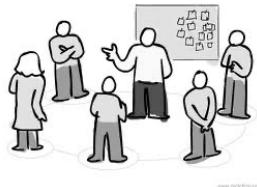
- ✖ Se escriben en un lenguaje del ámbito de negocios.
- ✖ Aunque cualquiera puede escribir una prueba, el Product owner es el principal propietario de la prueba.
- ✖ Son pruebas de caja negra en el sentido de que sólo se verifican que las salidas del sistema cumplen las condiciones de satisfacción indicadas, sin preocuparse de cómo se consiguen los resultados.
- ✖ Se implementan en el transcurso de la iteración en la que se implementa la propia historia de usuario.



Pruebas de aceptación (Confirmación)

- ✖ Podemos añadir pruebas de aceptación mientras añadan valor y claridad a la HU.

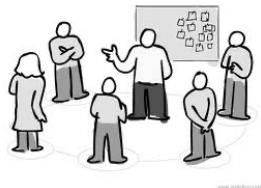
- ✖ No hay que olvidar que el equipo de desarrollo realiza pruebas de unidad sobre los módulos que se desarrollen, independientemente de las pruebas de aceptación definidas.



Pruebas de aceptación (Confirmación)

Se podrían describir usando el formato (Behavior-Driven Development):

- × Dado un sistema donde existe un usuario con nombre “idprueba”.
- × Cuando alguien intenta registrarse con nombre “idprueba”.
- × Entonces el sistema muestra un mensaje indicando que el nombre “idprueba” ya existe.



Pruebas de aceptación (Confirmación)

Como cliente. Quiero cancelar
una reserva previa

- ✗ Comprobar que un Cliente Premium puede cancelar en el mismo día sin coste adicional.
- ✗ Comprobar que a un cliente No-Premium se le carga un 10% al cancelar el mismo día.
- ✗ Confirmar que se le envía un mail de confirmación al cliente.
- ✗ Confirmar que el hotel recibe notificación de la cancelación.



Pruebas de aceptación (Confirmación)

Un cliente puede pagar un viaje usando una tarjeta de crédito.

- ✗ Comprobar con una Visa, Mastercard y American Express (funciona).
- ✗ Comprobar que con una tarjeta Diner's Club (no funciona).
- ✗ Comprobar con un número de tarjeta bueno, malo y sin número.
- ✗ Comprobar con una tarjeta expirada.
- ✗ Comprobar con diferentes cargos.



Características de las HU

- ✗ Son **cortas y fáciles de leer**, entendibles por los desarrolladores, interesados y usuarios.
- ✗ Representan **incrementos pequeños de funcionalidad valorada**, que puede ser desarrollada en un período de días o semanas.
- ✗ Son relativamente **fáciles de estimar** porque el esfuerzo de implementar la funcionalidad puede determinarse rápidamente.
- ✗ No se gestionan con documentos grandes o pesados, sino más bien en **listas organizadas** que pueden ordenarse más fácilmente y reordenarse a medida que se descubre nueva información.
- ✗ Necesitan **poco o ningún mantenimiento** y se pueden desechar con seguridad después de la implementación.
- ✗ Las historias de usuario, y el código que se crea a continuación, sirven como documentación, la cual también es elaborada de manera incremental.



HU terminada (Done)

- × Definir qué significa que una HU está “terminada”.
- × La definición de “terminado” es un acuerdo de calidad entre los desarrolladores y los clientes.
- × Se puede establecer una definición de “terminada” para todas las HU o un “terminado” específico para algunas HU.



HU terminada (Done)

- ✗ Aspectos a considerar en la definición de “terminada”:
 - Tipos de prueba y el grado de prueba.
 - Nivel de calidad del producto software (métricas, etc.)
 - Nivel de documentación,
 - Entorno específico en que concluye el trabajo: en prepro, testing, producción.
 - etc...



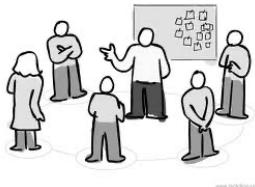
¿Por qué HU?

- ✖ Mantenemos una **relación cercana** con el **cliente**.
- ✖ Nos centramos en los **aspectos relevantes actuales** aplazando los detalles para cuando sean necesarios.
- ✖ **Desarrollo oportunista** (Requisitos emergentes).
- ✖ Facilita el **trabajo iterativo**.
- ✖ Buen tamaño para planificar y estimar.

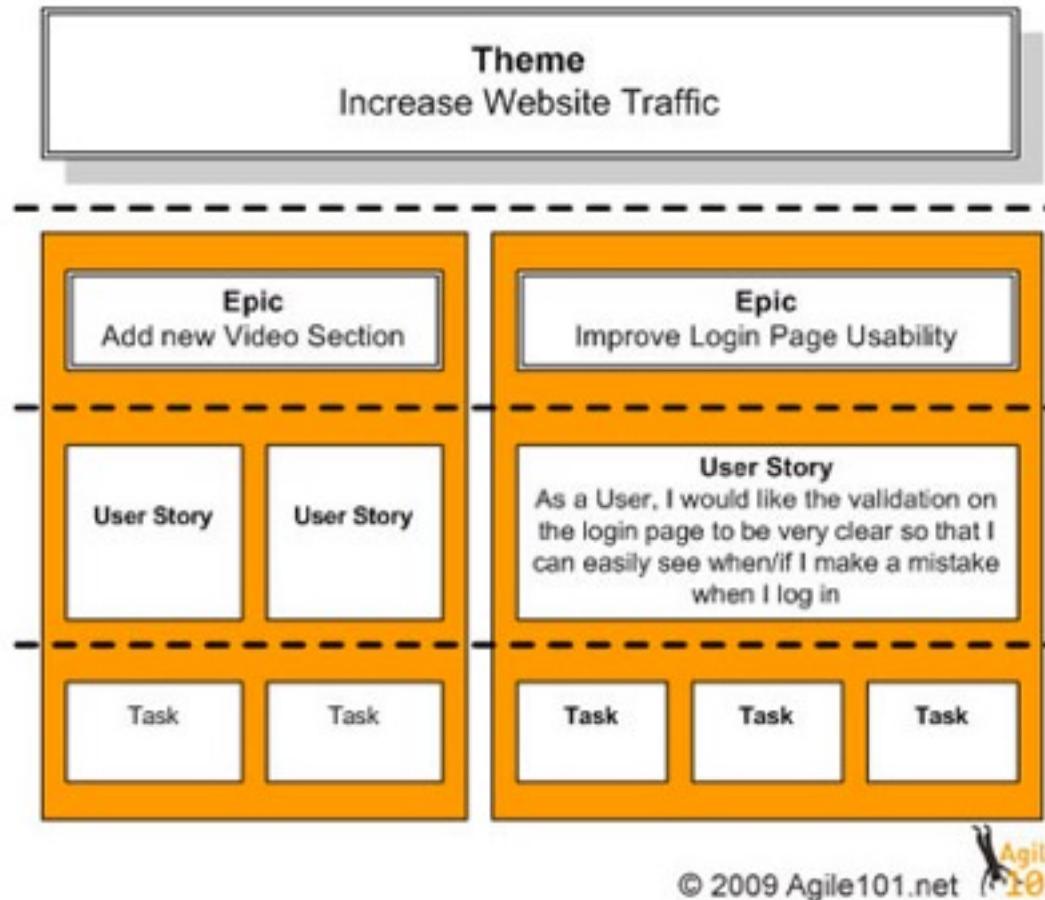


Granularidad de las HU

- ✗ **Temas.** Grandes proyectos, peticiones globales sin más análisis ni detalles. Subproductos.
Planificador de viajes, gestor de ofertas de hoteles, sitio Web para socios.
- ✗ **Épicas.** Súper historias de usuario, más concretas que los Temas. Conjunto de HU relacionadas entre sí.
Sistema de búsqueda para hoteles, anotador de satisfacción de instalaciones y viajes, filtros aplicados a las búsquedas, comparador avanzado de ofertas.
- ✗ **Historias de Usuario.** Una manera simple de describir una tarea concisa que aporta valor.
Como cliente quiero buscar un hotel en una localidad determinada, como cliente quiero obtener un viaje una zona determinada y bajo un rango de precios, como socio quiero pagar la reserva de mi viaje usando una tarjeta de crédito.
- ✗ **Tareas.** Las HU pueden ser divididas en diversas tareas por necesidades técnicas o de desarrollo.
Crear la pantalla de presentación de resultados de la búsqueda de hoteles, crear los métodos de consulta a BBDD para que retornen resultados, mostrar mensaje si no se encuentran resultados con los criterios de búsqueda, probar integración del modulo de socios con el sitio Web.



Granularidad de las HU



The Hierarchy of Agile Requirement Formats - Themes, Epics, User Stories, Tasks



Requisitos no funcionales

- ✗ Describen **cualidades o restricciones** del sistema que no se relacionan de forma directa con el comportamiento funcional del mismo.

Se pueden aplicar:

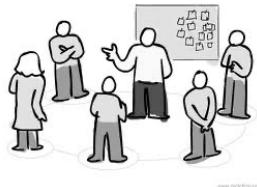
- ✗ Al **sistema o producto en su conjunto**.
- ✗ A **determinadas funciones o características del sistema**.



Requisitos no funcionales

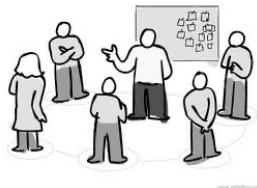
Los RNF pueden tratarse como:

- ✗ **Restricciones** asignadas a cualquiera de los elementos que **tratamos** en **las pilas de desarrollo** (Temas, Épicas, HU y Tareas).
- ✗ Criterios de aceptación.
- ✗ Como **parte de la definición de terminada (done)**.
- ✗ **Documento de especificación específico** para los requisitos no funcionales globales al sistema.



Requisitos no funcionales: Ejemplos

- × Como comprador, quiero que el sitio web esté **disponible el 98% de las veces que intento acceder a él** para poder realizar mi compra y no tener que buscar otro lugar para adquirir el producto.
- × Como cliente de banca online, quiero **que el saldo de mi cuenta corriente se muestre en 5 segundos** cuando lo solicite para no frustrarme y poder ver si tengo fondos para pagar mis facturas.



Requisitos no funcionales: Ejemplos

- × Como analista financiero, quiero ver las transacciones mensuales de mis clientes para poder aconsejarles sobre su salud financiera.

Criterios de aceptación:

- × El sistema muestra todas las transacciones que cumplen los parámetros de búsqueda en los 10 segundos siguientes a la recepción de la solicitud.
- × Las transacciones de los clientes etiquetados como confidenciales sólo se muestran a los usuarios con nivel de seguridad 2.

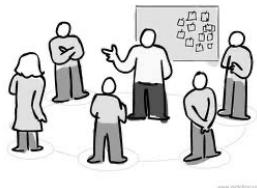


Modelo I.N.V.E.S.T.

¿Cuándo consideramos que una historia de usuario es buena?

- ✗ **Independiente**: Una historia debería ser independiente de otras (lo más posible).
- ✗ **Negociable**: Las HU no son un contrato o un requerimiento que el software debe implementar.
- ✗ **Valiosa**: Tiene que tener valor para el cliente (para el usuario o para el comprador). El cliente las escribe, no son un contrato y son negociables.
- ✗ **Estimable**: Para permitir que se pueda priorizar y planificar la historia.
- ✗ **Suscinta**: Una buena historia debe ser pequeña en esfuerzo (2-3 personas/semana de trabajo).
- ✗ **Testeable**: Una historia necesita poder probarse para que ocurra la etapa de "Confirmación".

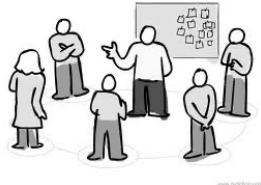
[Bill Wake]



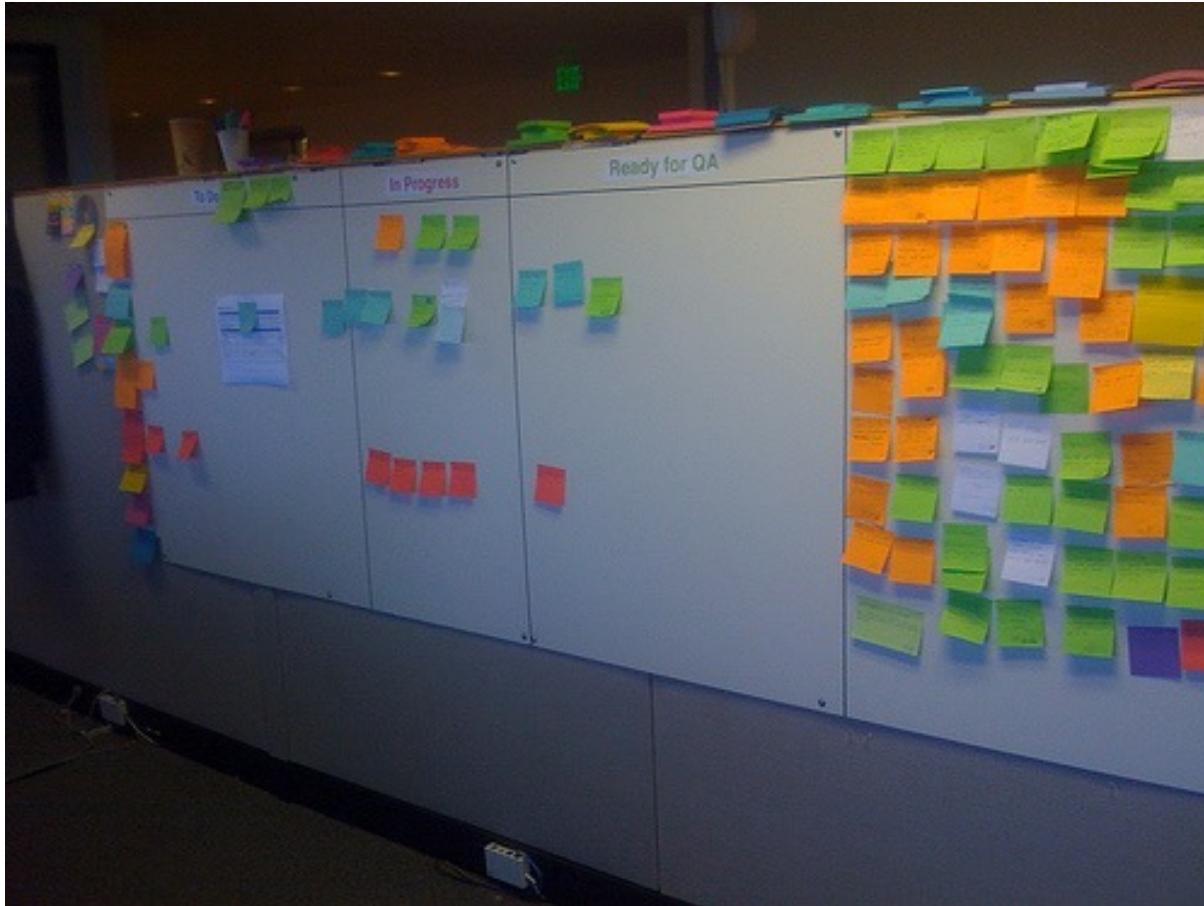
Pila de producto (Product backlog)

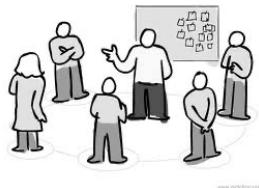
“Lista priorizada de HU que contiene toda la funcionalidad deseable del producto”.

- ✗ Prioridad. Valor que le da el cliente a la HU (Medida en la que permite satisfacer sus necesidades).
- ✗ Cubre todas las funcionalidades necesarias.
- ✗ Se revisa/actualiza frecuentemente.
- ✗ Es una herramienta de planificación y gestión.



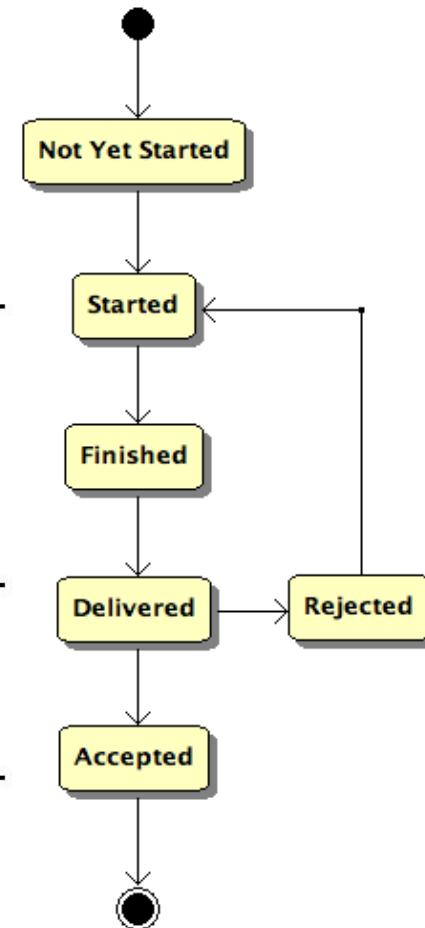
Mapa de historias de usuario

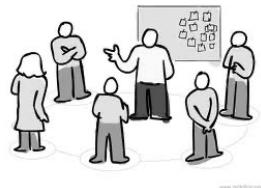




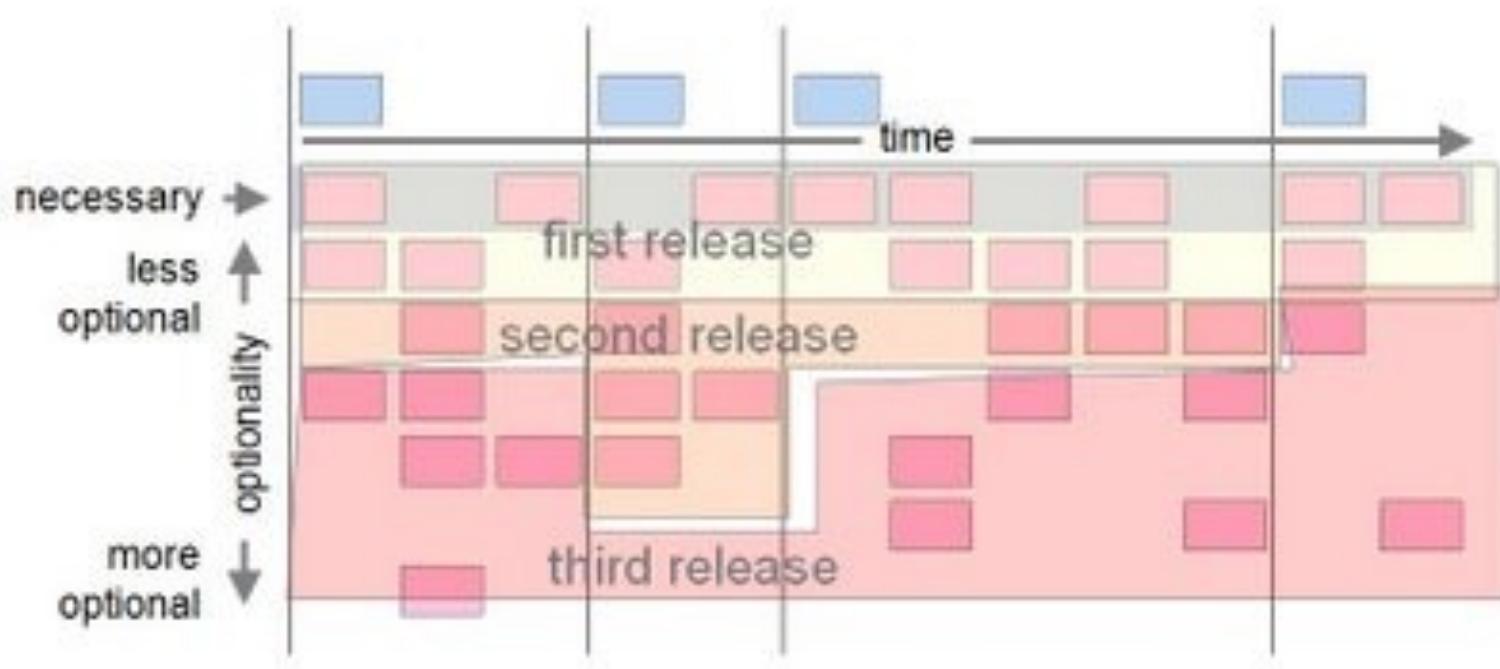
Mapa de historias de usuario (2)

Story	To Do		In Process	To Verify	Done
As a user, I... 8 points	Code the... 9	Test the... 8	Code the... DC 4	Test the... SC 6	Code the... D Test the... SC 8
	Code the... 2	Code the... 8	Test the... SC 8	Test the... SC	Test the... SC 6
	Test the... 8	Test the... 4		Test the... SC	
As a user, I... 5 points	Code the... 8	Test the... 8	Code the... DC 8		Test the... SC
	Code the... 4	Code the... 6		Test the... SC	Test the... SC 6





Mapa de historias de usuario (3)





Mapa de historias de usuario (4)

Id.	Descripción	Prioridad	Estimación	Criterio de validación	Sprint
1	Actualización de datos generales de los empleados Cálculo automático de antigüedad de los empleados según fecha de ingreso a la empresa	1 5	3 2	Se dispondrá de los kardex actuales de empleados Los datos iniciales deberán ser similares a los kardex de empleados	1 1
3	Permitir la acumulación máxima de dos gestiones de vacaciones	6	2	Las vacaciones más antiguas caducan	2
4	Registro de las vacaciones solicitadas de los empleados	2	3	El registro deberá permitir tomar días enteros o medios días de vacación	1
5	Cálculo automático de la fecha de conclusión de la vacación de acuerdo al número de días solicitados	7	2	Considerar días laborales para el cálculo. Identificar automáticamente sábados y domingos y considerar los feriados	2
6	Registro de días feriados	3	2	De acuerdo a los feriados establecidos en el país y la empresa.	1
7	Permitir la autorización previa de la solicitud de vacación por parte del jefe de área.	8	2	Deberá aprobarse previo a su procesamiento en el área de Personal	3
8	Procesamiento de la solicitud de vacación en el área de Personal, actualizando el saldo de vacaciones del empleado	9	2	Considerar los días disponibles de vacación del empleado	2
9	Emisión de comprobantes de vacaciones	10	2	Usar formulario establecido	
10	Actualización de personal autorizado para la aprobación de solicitudes	4	3	Listo provista por la gerencia	1
11	Recordatorio de vacaciones acumuladas en riesgo de vencimiento.	11	2	Se deberá emitir un listado de las vacaciones acumuladas de los empleados que están a punto de caducar por tener una acumulación mayor a dos años.	3
12	Emisión de informes de personal en vacación para un periodo.	12	5		3

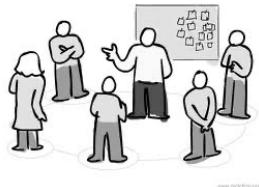
Product Backlog

Lista de HU a tratar en el proyecto



Proceso de definición de las HU

- ✗ **Toma de requisitos** en reuniones con usuarios/clientes (Entrevistas, cuestionarios, observación).
- ✗ Informar de la **visión general** transmitida por el cliente.
- ✗ Crear HU (reuniones de escritura de HU).
- ✗ Actuar sobre cada HU mediante un **estudio funcional y técnico** realizado en equipo (Conversación).
- ✗ Definir criterios de **aceptación**.
- ✗ Estimar [1] el tamaño de las HU para priorizarlas [2].
- ✗ Descomponer las HU en tareas [3].



Estimar el tamaño de las HU [1]

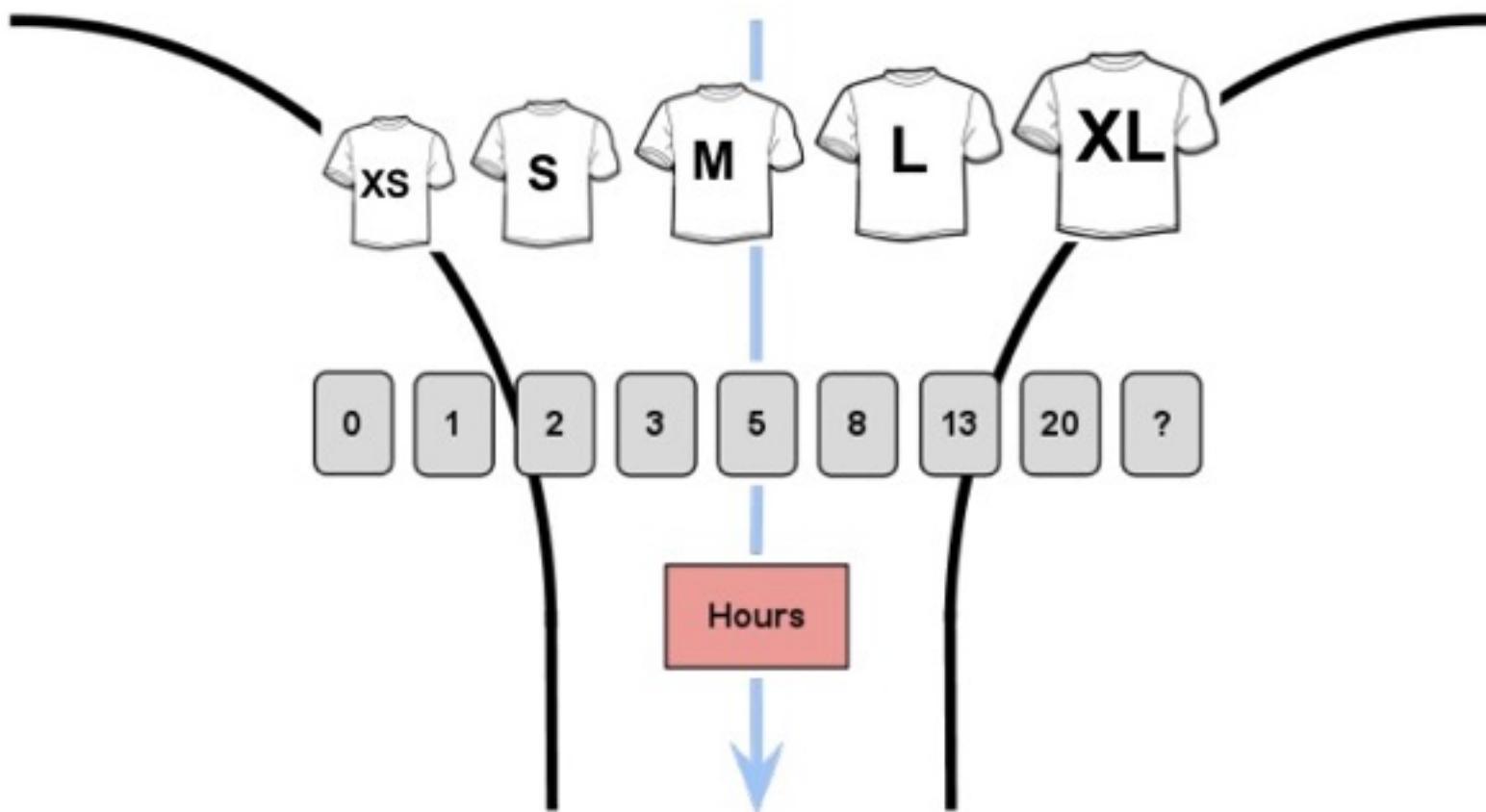
Estimación. “Puntos de historia” o “Días ideales”

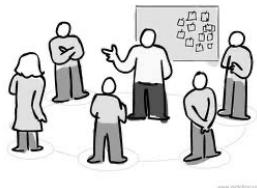
- ✗ **Puntos de Historia:** Tamaño relativo de una historia comparado con otras historias estimadas por el mismo equipo.(esfuerzo necesario + complejidad + riesgo + ...)
- ✗ **Días ideales:** Un día de trabajo sin interrupciones.
- ✗ **Estimado por el equipo de desarrollo.**



Estimar el tamaño de las HU [1]

Cono de Incertidumbre

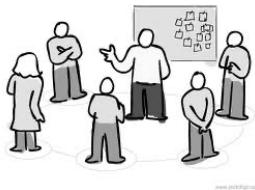




“Planning poker” (El juego de la planificación)

- × Técnica para realizar una **estimación** en base al **consenso entre los estimadores**.
- × Un mazo típico contiene tarjetas mostrando la Secuencia de Fibonacci (0, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89) o (0, $\frac{1}{2}$, 1, 2, 3, 5, 8, 13, 20, 40, 100)





“Planning poker” (El juego de la planificación)

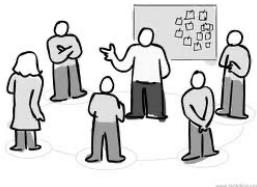
- ✖ Existe un **moderador** que no estima y gestiona la reunión.
- ✖ El **desarrollador con más conocimiento** de una determinada característica **proporciona una breve introducción** sobre la misma. El equipo tiene la oportunidad de **hacer preguntas y discutir** para aclarar los supuestos y riesgos.
- ✖ Cada **persona coloca una carta del mazo boca abajo representando su estimación**. Todo el mundo muestra sus cartas a la vez.
- ✖ A las personas con **estimaciones fuera del promedio** (altas y bajas) se les da un tiempo para **ofrecer su justificación para la estimación y la discusión continúa**.
- ✖ Se **repite** el proceso de cálculo hasta que se **alcance un consenso**.
- ✖ Se asegura de respetar los tiempos asignados a cada estimación, permitiendo hacer de esta reunión un ejercicio productivo y eficiente.



Priorización de las HU [2]

- **Requeridas o críticas**. Sin las cuales la solución no puede vivir (entregas primeras del proyecto).
- **Importantes**. Sin las cuales el sistema puede vivir durante algún tiempo (entregas intermedias del proyecto).
- **Opcionales**. Si hay tiempo y presupuesto se construyen. (entregas finales del proyecto).
- **No se construirán**. Por restricciones de presupuesto o de tiempo o no tienen ningún valor para el negocio.

Las HU pueden ser **negociadas** con el cliente y van a evolucionar a lo largo del proyecto.



Priorización de las HU [2]

- ✗ Priorizar las HU en base al valor para el cliente (Valor de Negocio).
- ✗ Tener en cuenta el tamaño y los riesgos de cada objetivo.
- ✗ La prioridad puede cambiar el tamaño no.
 - ✗ HU importante para un número alto de usuarios.
 - ✗ HU importante para un número pequeño de usuarios importantes.
 - ✗ HU muy relacionadas con otras HU priorizadas previamente.



Priorización de las HU [2]

✗ Método MoSCoW

- *M* – **Must**, se debe completar este requerimiento para finalizar el proyecto.
- *S* – **Should**, se debe completar este proyecto por todos los medios, pero el éxito del proyecto no depende de él.
- *C* – **Could**, se debería completar este requerimiento si su implementación no afecta a la consecución de los objetivos principales del proyecto.
- *W* – **Would**, se puede completar este requerimiento si sobra tiempo de desarrollo (o en futuras versiones del mismo).



Velocidad del equipo

- ✖ Es la cantidad de **puntos de historia** que un equipo puede completar en una **iteración**.
- ✖ Se calcula por observación en varias iteraciones del proyecto.
- ✖ Permite **planificar** las **HU** que se abordaran en una iteración.



Velocidad del equipo

× Calcular la velocidad:

- Utilizar datos históricos (“el tiempo que hizo ayer”).
- Ejecutar una iteración inicial y utilizar la velocidad de esa iteración.
- Hacer un pronóstico (“el cálculo de recursos”)



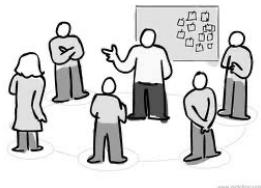
Cálculo de la Velocidad: Primer Sprint

- × Partimos de un equipo de desarrollo de 5 miembros.
- × Las iteraciones las vamos a realizar de 2 semanas.
 1 iteración = 10 días de trabajo
- × En nuestro entorno de trabajo estimamos que:
 1 PH = 1 día de trabajo ideal
 Todos los miembros del equipo trabajan las mismas horas.

La **velocidad** del equipo de trabajo:

5 programadores * 10 días de trabajo = 50 días de trabajo por iteración
Vamos a considerar un factor de dedicación del 80 %

$$\text{Velocidad} = 50 * 80\% = 40 \text{ PH}$$



Planificación de Iteraciones

Asignar HU a las iteraciones dentro de cada una de las entregas de software

Story	Story Points
Story A	3
Story B	5
Story C	5
Story D	3
Story E	1
Story F	8
Story G	5
Story H	5
Story I	5
Story J	2

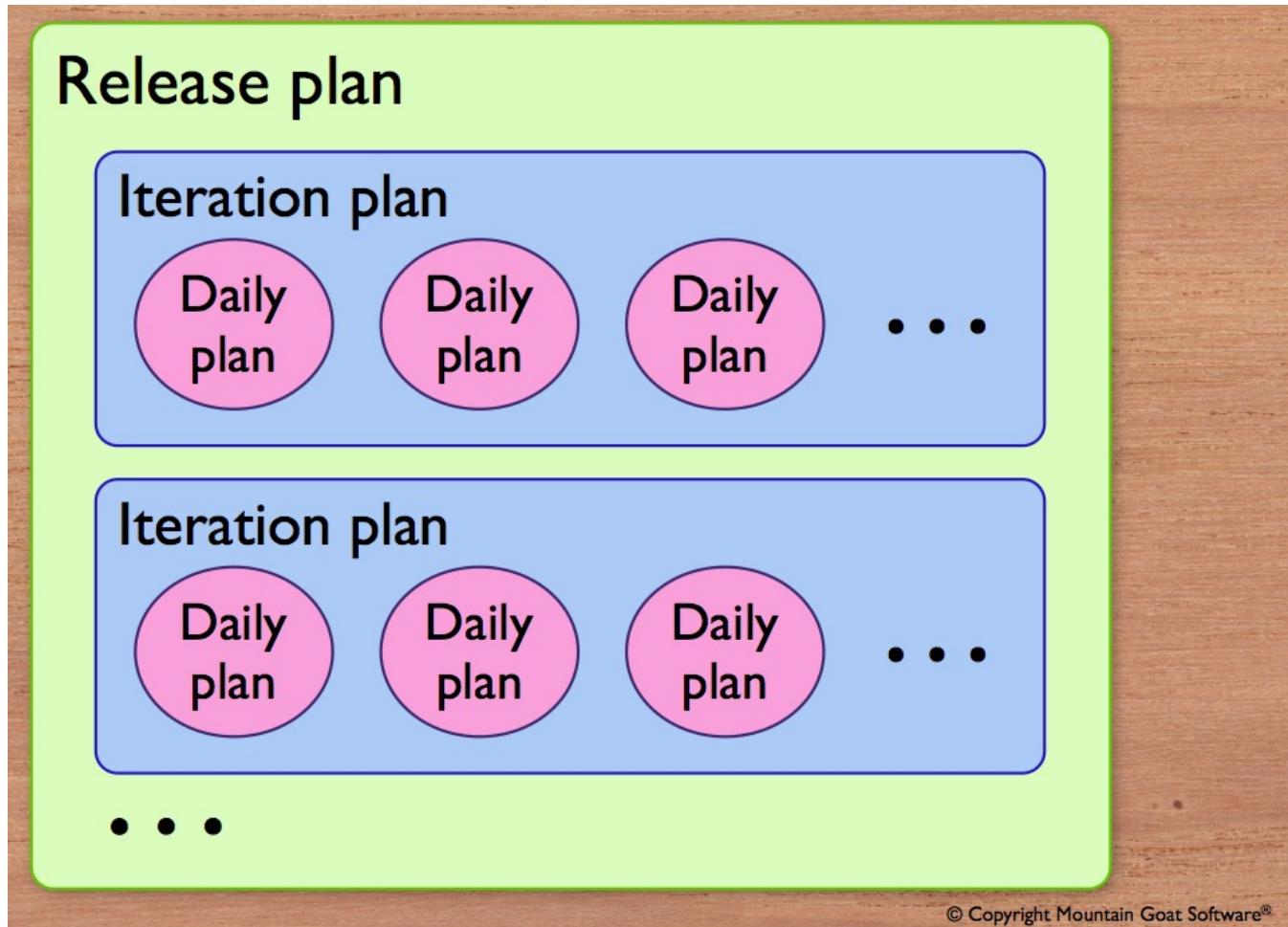
Velocidad del equipo = 13

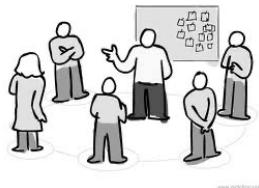
Iteration	Stories	Story Points
Iteration 1	A, B, C	13
Iteration 2	D, E, F	12
Iteration 3	G, H, J	12
Iteration 4	I	5

Iteration	Stories	Story Points
Iteration 1	A, B, C	13
Iteration 2	D, E, F	12
Iteration 3	G, H, Y	13
Iteration 4	J, Z	4



Planificación de Iteraciones





Descomposición de una HU en tareas [3]

- ✖ Las HU hay que descomponerlas en tareas que puedan ser tratadas de forma individual por el equipo de desarrollo.
- ✖ La duración de una tareas debe estar entre medio día a 3 ó 4 días de trabajo.
- ✖ Una vez terminada la tarea se debe generar un producto entregable.
- ✖ No dedicar mucho tiempo a estudiar detalles de las tareas.
- ✖ Incluir tareas de prueba y automatización.



Descomposición de una HU en tareas [3]

HU: Como administrador quiero poder realizar un mantenimiento de los clientes de la empresa.

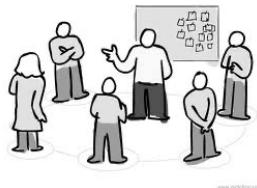
- Construir la IU.
 - Construir la lógica de negocio.
 - Construir la capa de datos.
-
- Implementar añadir usuario.
 - Implementar modificar usuario.
 - Implementar eliminar usuario.
 - Realizar pruebas de añadir/modificar/eliminar usuario.



Descomposición de una HU en tareas [3]

HU: Como cliente, quiero buscar un hotel

- ✗ Codificar la pantalla de búsqueda básica.
- ✗ Codificar la pantalla de búsqueda avanzada.
- ✗ Codificar las pantallas de resultados.
- ✗ Escribir las consultas SQL para las búsquedas.
- ✗ Documentar la nueva funcionalidad en manuales y ayudas.



Patrones para partir HU grandes

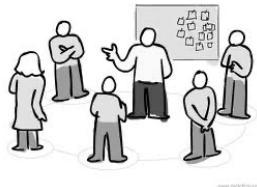
1. Pasos del flujo de trabajo.

Identificamos los pasos que realiza el usuario para hacer una actividad e implementamos esos pasos de una forma incremental.

Como un vendedor, quiero actualizar y publicar los nuevos programas de precios para los clientes.

- ✗ ... quiero publicar los programas de precios en la página principal.
- ✗ ... quiero enviar un mensaje a cada cliente con los nuevos precios, usando el portal web.
- ✗ ... quiero enviar la lista de precios actualizados con descuentos personalizados a cada uno de los clientes.

<http://www.richardlawrence.info/2009/10/28/patterns-for-splitting-user-stories/>



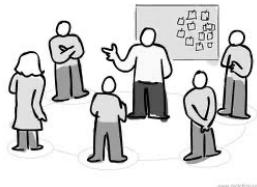
Patrones para partir HU grandes

2. Variaciones de las reglas de negocio

Las HU pueden resultar sencillas sino tenemos en cuenta todas las posibles reglas de negocio que hay que aplicar.

Como un vendedor, quiero calcular el descuento que le vamos a aplicar a un cliente en todas sus compras.

-
- ✗ ... teniendo en cuenta las últimas compras que ha realizado.
 - ✗ ... teniendo en cuenta el potencial de venta que tiene en función de la demografía.



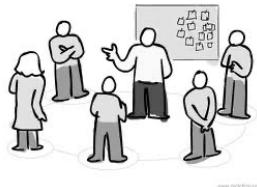
Patrones para partir HU grandes

3. El mayor esfuerzo de desarrollo

Algunas veces podemos partir una HU en varias partes de forma que las primeras tendrán un mayor esfuerzo de desarrollo y las siguientes van a consistir en añadir mayor funcionalidad.

Como un usuario, quiero poder cambiar la tarifa de electricidad que se me aplica usando un selector en la página de cliente.

- ✗ ... quiero usar la tarifa de medida de uso de electricidad.
- ✗ ... quiero usar la tarifa de prepago.
- ✗ ... quiero usar las tarifas de descuento.



Patrones para partir HU grandes

4. Sencillo / complejo

Si durante la descripción de la HU cada vez se va complicando y parece ser más larga, hay que parar y preguntarse, ¿qué es lo más simple que podría funcionar?. Luego podremos añadir variaciones y complejidad sobre nuevas HU.

Como un usuario, quiero tener siempre el precio más bajo de un producto.

- ✗ ... teniendo en cuenta el proveedor que de el precio más bajo.
- ✗ ... avisando de posibles ofertas que van a aparecer en el futuro próximo.



Patrones para partir HU grandes

5. Variaciones en datos

Las variaciones sobre los datos a usar pueden ser un origen de aumento de la complejidad.

Como un administrador, quiero enviar mensajes a los clientes.

-
- ✗ ... en inglés.
 - ✗ ... en español.



Patrones para partir HU grandes

6. Métodos de entrada de datos (Vista del IU)

Muchas veces la complejidad está en el IU más que en la funcionalidad. Podemos empezar con una funcionalidad básica y añadir posteriormente una interacción más rica.

Como un usuario, quiero ver la evolución de mis compras usando gráficos estadísticos.

- ✗ ... usando una gráfica de barras que compara las ventas mensuales.
- ✗ ... usando una gráfica de barras con el criterio seleccionable de un conjunto definido.



Patrones para partir HU grandes

7. Niveles de calidad del sistema

Muchas veces las primeras implementaciones no son lo suficientemente eficientes como podrían ser. El equipo puede aprender mucho del desarrollo y posteriormente darle un mayor valor al cliente.

Como un usuario, quiero ver resultados en tiempo real de mis medidas.

- ✗ ... usando medidas interpoladas de las últimas medidas tomadas.
- ✗ ... usando medidas obtenidas en tiempo real del sistema.



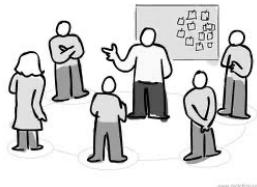
Patrones para partir HU grandes

8. Operaciones múltiples

Operaciones como gestionar, administrar, controlar pueden ser divididas en múltiples operaciones.

Como un usuario, quiero poder gestionar mi cuenta.

- ✗ ... darme de alta como cliente.
- ✗ ... editar los parámetros de la cuenta.
- ✗ ... cancelar la cuenta.
- ✗ ... añadir más dispositivos a la cuenta que tengo asociada.



Patrones para partir HU grandes

9. Escenarios de un caso de uso

Si partimos de un estudio de casos de uso, podemos partir cada uno de ellos en escenarios individuales.

Como un usuario, quiero poder gestionar mi cuenta.

- ✗ ... darme de alta como cliente.
- ✗ ... editar los parámetros de la cuenta.
- ✗ ... cancelar la cuenta.
- ✗ ... añadir más dispositivos a la cuenta que tengo asociada.



SPIKE

Tipo de HU usada para gestionar una dificultad técnica. Tiempo necesario para reducir un riesgo o una incertidumbre de un problema técnico o para aumentar la seguridad sobre una estimación de una HU.

SPIKE Técnico: Analizar varias aproximaciones técnicas en el dominio de la solución del problema.

- Decidir si comprar o desarrollar algo para la solución.
- Evaluar el rendimiento de una solución propuesta.
- Evaluar el impacto de carga de una nueva HU.

SPIKE Funcional: Si hay una incertidumbre importante en cómo el usuario debe interactuar con el sistema.



SPIKE

Puede darse por:

- El equipo no tiene suficiente conocimiento sobre un dominio y usamos la HU para familiarizarnos con el dominio o una tecnología nueva.
- Tenemos una HU grande y la estimación necesita ser partida en trozos y analizar lo que implica.
- La HU tiene riesgos técnicos y el equipo necesita realizar un prototipo o algo de desarrollo para ganar confianza en el desarrollo o el diseño a realizar.
- La HU tiene riesgo funcional y no está claro como tiene que actuar el sistema con usuarios para alcanzar los objetivos planteados.



Ejemplos de SPIKE

Como un cliente, quiero ver el gasto que he tenido de electricidad en un histograma, de forma que pueda comprender el gasto realizado y el que se prevé que gaste.

- ✗ SPIKE Técnico. Investigar cuánto tiempo se tarda en actualizar una pantalla personalizada con el uso actual y los requisitos de comunicación y de gestión que tendrá.
- ✗ SPIKE Funcional. Realizar un prototipo de histograma en el portal web y obtener realimentación de varios usuarios sobre la forma de presentar la información.



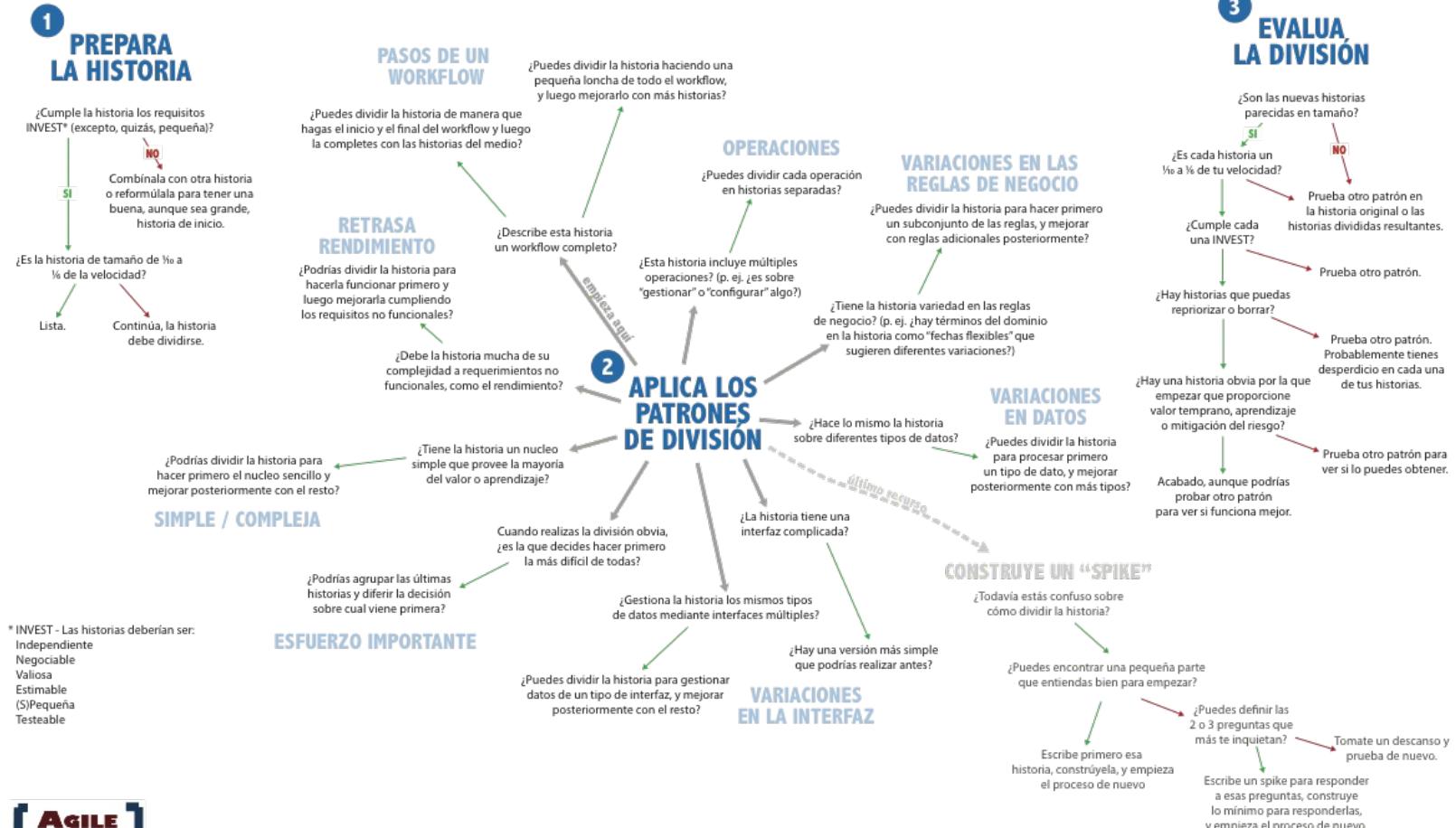
Reglas sobre SPIKE

- ✖ Se tratan como otras HU (las ponemos en el backlog, las priorizamos, las estimamos, etc.)
- ✖ Su resultado no es código, es información útil para otras HU. Debe generar un decisión, un prototipo, una demostración de un diseño, o una solución parcial de un problema.
- ✖ La solución de un SPIKE debe ser demostrable al equipo de desarrollo.
- ✖ Un SPIKE, como cualquier HU, debe ser aceptada usando criterios de aceptación definidos en la HU.
- ✖ Puede ser interesante dejar la resolución del SPIKE para una interacción diferente a la de la HU que lo creó.



Patrones para partir HU grandes

COMO DIVIDIR UNA HISTORIA DE USUARIO



Traducción por Joserra Díaz con Martin Alaimo y Alan Cyment.
Visita <http://www.richardlawrence.info/splitting-user-stories/> para más información sobre los patrones.
Copyright © 2011-2013 Agile For All. All rights reserved.

Última modificación 6/2/2013



Partir HU Grandes

Gestionar Usuarios

13

Registrar un nuevo Usuario

3

Generar factura en HTML, PDF o Excel

Editar un Usuario existente

5

Como operador quiero saber que usuarios están conectado en el sistema

Buscar un Usuario

3

Como administrador quiero poder tener 100 usuarios simultáneos

Borrar un Usuario

5

Administración manual del sistema de usuarios

Generar factura en HTML, PDF o Excel

Registrar un nuevo Usuario

3

Editar un Usuario existente

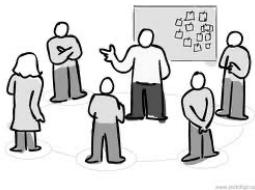
5

Generar factura en HTML, PDF o Excel

Como operador quiero saber que usuarios están conectado en el sistema

Buscar un Usuario

3



Partir HU Grandes

Priority	Feature
1	Book search capability (title, author, subject)
2	Online shopping cart
3	Credit card processing integration
4	Book details
5	Order tracking
6	Book preview

Priority	Feature
1	Book Search by Title
2	Online shopping cart
3	Book Search by Author
4	Credit card processing integration
5	Book Search by Subject
6	Book details
7	Order tracking
8	Book preview
9	Book Search by ISBN



Lecturas / Bibliografía

- ✗ Dean Leffingwell, Pete Behrens, “*A User Story Primer*”, (2009)
http://scalingsoftwareagilityblog.com/wp-content/uploads/2010/01/user-story-primer_1.pdf
- ✗ Dean Leffingwell, “*A Lean and Scalable Requirements Information Model for the Agile Enterprise*”, (2009)
<http://scalingsoftwareagility.files.wordpress.com/2007/03/a-lean-and-scalable-requirements-information-model-for-agile-enterprises-pdf.pdf>
- ✗ Mike Cohn, “*User Stories Applied for Agile Software Development*”, Addison Wesley, 2004
- ✗ Mike Cohn, “*Agile Estimating and Planning*”, Prentice Hall, 2005