Desarrollo de Software I

Facultad de Ingeniería
Escuela de Ingeniería de Sistemas y Computación

Mauricio Gaona
mauricio.gaona@correounivalle.edu.co
Profesor

2023-I

Desarrollo de Software I

01 **RESUMEN**
Aspectos generales vistos en la clase anterior.

02 **PRÁCTICAS ÁGILES EN SCRUM**
Prácticas ágiles en la metodología Scrum

03 **CONCLUSIONES**
Resumen de aspectos importantes a tener en cuenta.

04 **PROXIMA CLASE**

Resumen

REQUERIMIENTOS DEL SOFTWARE

Clasificación de los requerimientos.

Los requerimientos funcionales

Los requerimientos especifican lo que desea un cliente y definen las acciones o actividades que debe hacer o cumplir un sistema de software.

Requerimientos no funcionales

Limitaciones en los servicios o funciones ofrecidas por el sistema como de tiempo, limitaciones en el proceso de desarrollo, normas, tecnología, etc

Imprecisión en los requerimientos.

RESUMEN

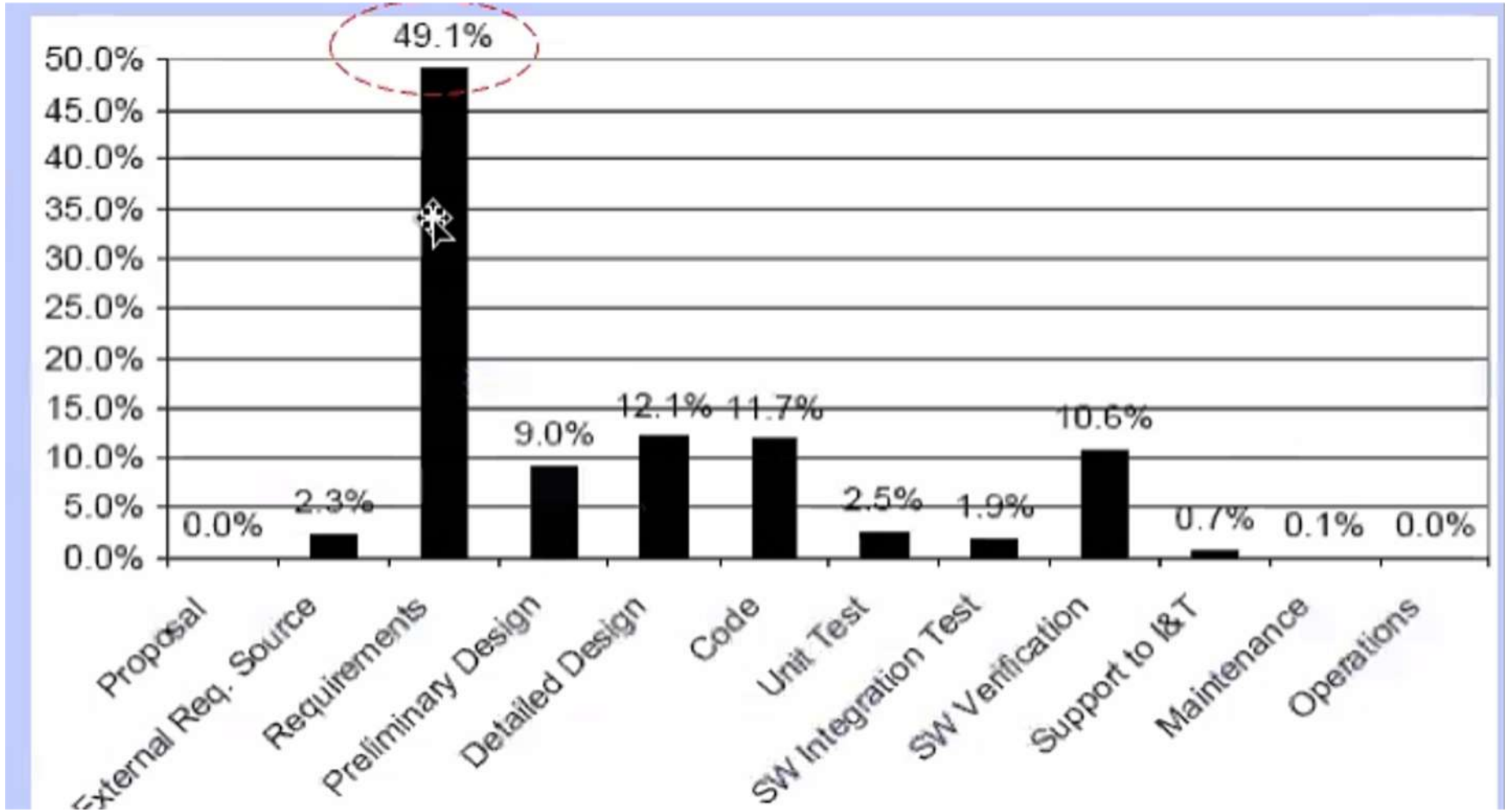
Los problemas surgen cuando los requerimientos no son declarados con precisión.

Requerimientos ambiguos pueden interpretarse de diferentes maneras por los desarrolladores y usuarios.

Deben estar redactados de tal forma que sean comprensibles para usuarios sin conocimientos técnicos avanzados.

¿De donde provienen los errores en el Desarrollo de software?

RESUMEN



Características de los requerimientos en las metodologías tradicionales

RESUMEN

Necesario: Lo que pida un requisito debe ser necesario para el producto.

Correcto: sí y solo sí, cada requisito especificado es un requisito que el software debe cumplir.

No ambiguo: El texto debe ser claro, preciso y tener una única interpretación posible.

Conciso: Debe redactarse en un lenguaje comprensible por los participantes en lugar de uno de tipo técnico y especializado, aunque aún así debe referenciar los aspectos importantes

Consistente: Ningún requisito debe entrar en conflicto con otro requisito diferente. Asimismo, el lenguaje empleado entre los distintos requisitos debe ser consistente también.

Completo: Los requisitos deben contener en sí mismos toda la información necesaria, y no remitir a otras fuentes externas que los expliquen con más detalle.

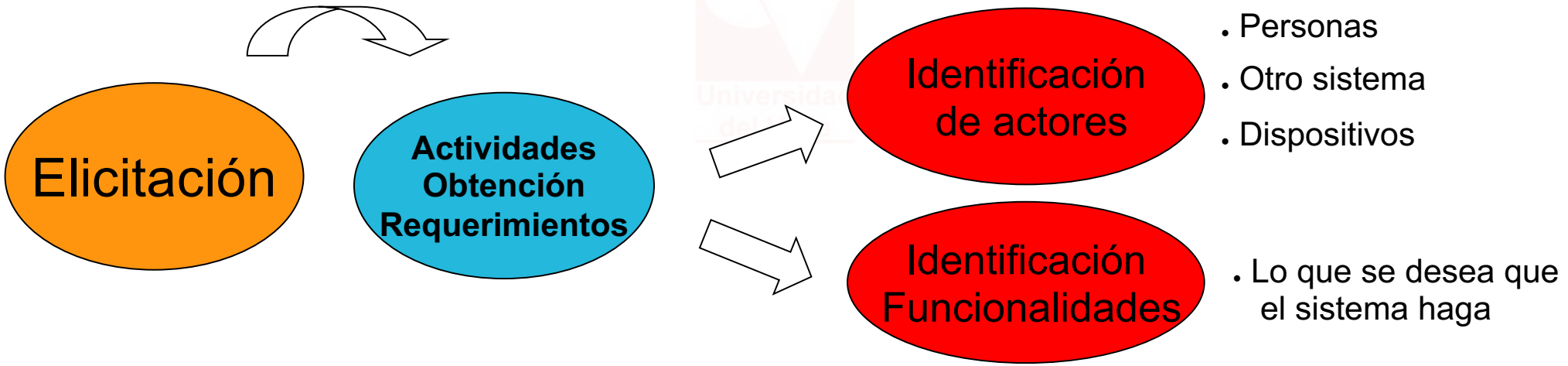
Alcanzable: Un requisito debe ser un objetivo realista, posible de ser alcanzado con el dinero, el tiempo y los recursos disponibles.

Verificable: Se debe poder verificar con absoluta certeza, si el requisito fue satisfecho o no. Esta verificación puede lograrse mediante inspección, análisis, demostración o testeo.

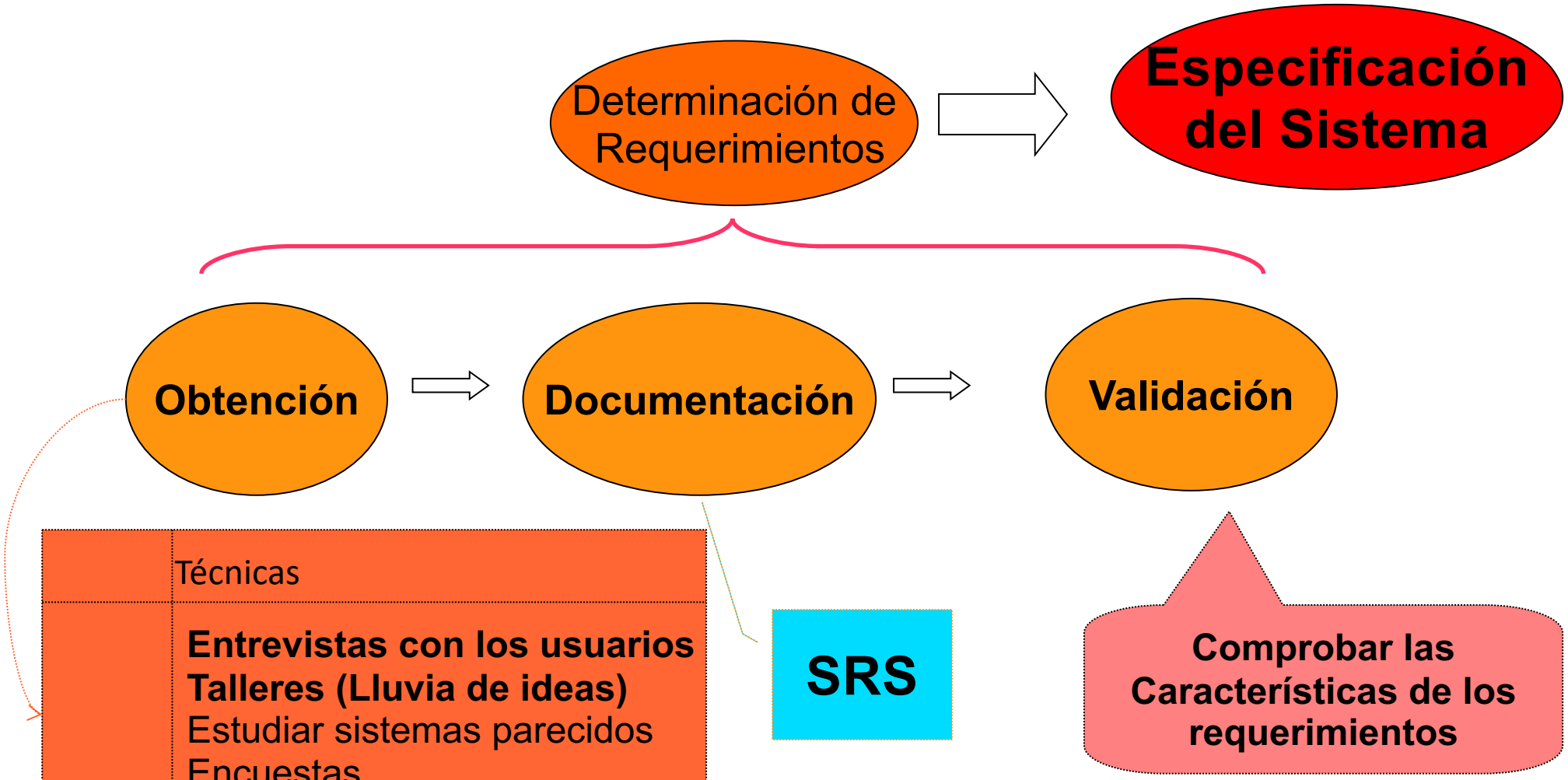
Formato para redactar requerimientos (Detalles de los requerimientos)

Nombre del requerimiento	El sistema debe permitir registrar participantes a un evento
Identificador	**Rq-02**
Tipo de Requerimiento	**Tipo de requerimiento:** Funcional
Datos de Entrada	**Nombre, cédula, edad, género, nivel de formación, correo electrónico, evento**
Descripción	Dado los datos de entrada el sistema debe permitir almacenar cada uno de los datos ingresados y enlazar a un participante con el evento al cual desea asistir.
Datos de salida	**Mensaje: "El participante se registró con éxito"**
Resultados esperados	El sistema tendrá un nuevo participante en uno de sus eventos en caso de que el proceso de registro sea exitoso.
Origen	**Necesidades del cliente.**
Dirigido a	**Operadores.**
Prioridad	**5**
Requerimientos asociados	**Rq-01**

Proceso para especificar requerimientos



Proceso para especificar requerimientos



Ejemplos de requerimientos funcionales en las metodologías tradicionales

El sistema debe permitir crear un usuario del sistema.

El sistema debe permitir registrar participantes a un evento.

El sistema debe permitir registrar la venta de un artículo.

El sistema debe permitir anular una nota débito.

El sistema debe permitir a los usuarios contratista poder registrar sus pagos de certificación al sistema de seguridad social junto al valor pagado por cada concepto como son: ARL, Pensión y EPS junto con el número de la planilla que lo soporta.

Requerimientos en las metodologías ágiles

Requerimientos en las metodologías ágiles

- Las Historias de Usuario (HU) es el mecanismo usado para obtener las necesidades del usuario.

Requerimientos en las metodologías ágiles

RESUMEN

Las Historias de Usuario (HU) es el mecanismo usado para obtener las necesidades del usuario.

Características de la historias de usuario (HU)

El modelo INVEST

Una buena historia de usuario también sigue el modelo de INVEST: Independiente, Negociable, Estimable, Pequeña (Small), y Testeable. Veamos lo que significa.

- **Independiente** - una historia debería ser independiente de otras. Facilitan la planificacion, priorizar y estimación.
- **Negociable** - La "tarjeta" de la historia es tan sólo una descripción corta que no incluye detalles. Los detalles se añaden mediante la conversación.
- **Valiosa** - cada historia tiene que tener valor para el cliente (para el usuario o para el comprador).
- **Estimable** - el equipo necesitan poder estimar una historia de usuario. Historias demasiado grandes o inconcretas, no se pueden estimar.
- **Pequeña** - una buena historia debe ser pequeña en esfuerzo, debería ser realizable en menos de una semana.
- **Testeable** - una historia necesita poder probarse y saber que la HU se ha completado con éxito.

Requerimientos en las metodologías ágiles

Ejemplos de historias de usuario

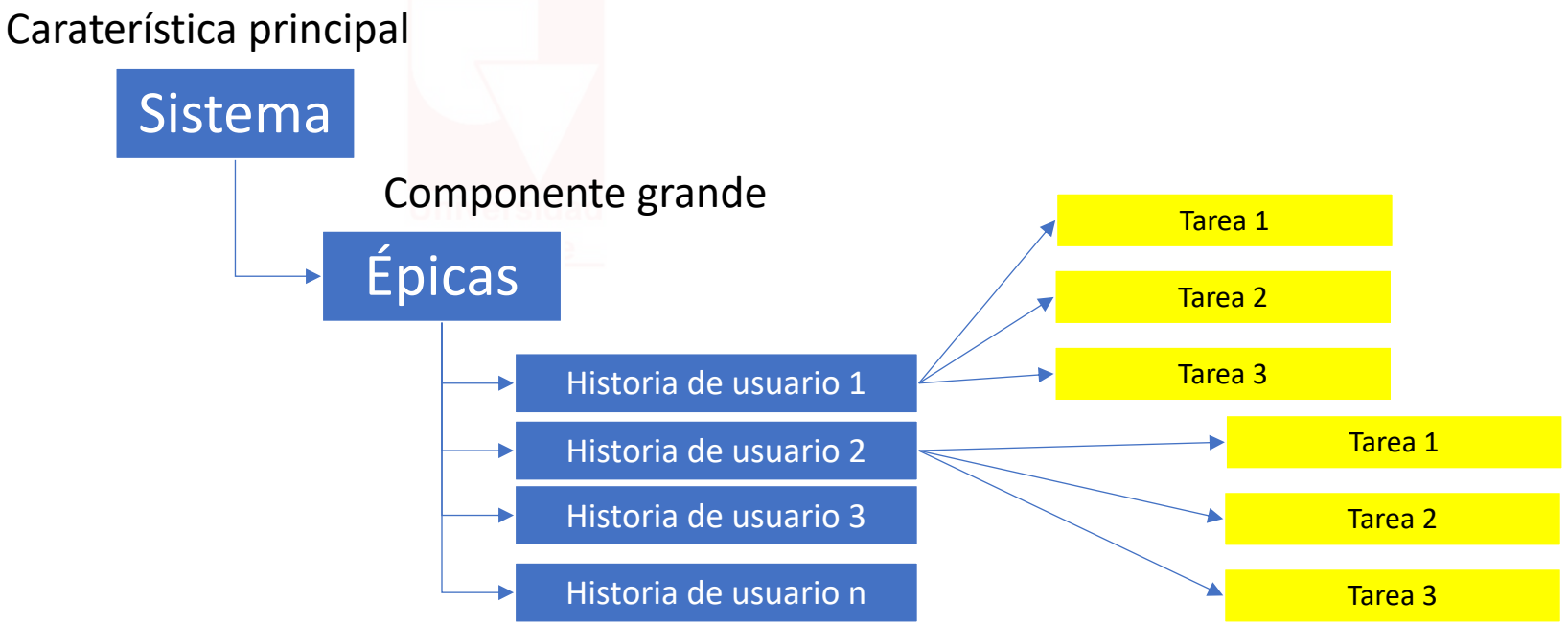
Como estudiante deseo registrarme en un curso para matricularme en la universidad

Actor — La tarea — Propósito

Plantilla: Como <Actor> deseo <tarea> para <propósito>

Historias de usuario épicas

“Una historia épica es una gran historia de usuario que no se puede entregar como se define en una sola iteración o es lo suficientemente grande como para dividirse en historias de usuario más pequeñas.”



Detalles de una historia de usuario

RESUMEN

Historia de Usuario (HU)			
Código HU:	HU0023	Fecha de inicio:	17/02/2022
Sprint:	1	Prioridad:	Alta
Actor(es):	Gerente	Puntos:	3

Descripción:
Como gerente deseo ver las ventas semanales de mi empresa para conocer como se comportan las ventas día a día.

Detalles de la HU:[Colocar aquí toda la información que se requiera para entender la HU: Texto (entradas, proceso, salida), fotos videos, audio, BPM, diseño de interfaces de usuario, etc]

Restricciones: (Escenario)
1. Solo los usuarios con privilegios de gerente pueden acceder al reporte.
2. El tiempo de despliegue del reporte no debe exceder 4 segundos

Criterios de aceptación: (Detalle de las HU desde el punto de vista de calidad y se traducen en pruebas)
Criterio 1, Criterio 2, …

DoD (Definition of Done):
Lista de actividades que se deben cumplir para que la historia esté en donciciones de ser entregado al cliente.

Detalles de una historia de usuario

Item	Criterio de acetantación
1	Un usuario no puede enviar un formulario sin completar todos los campos.
2	El login no puede ser igual al password.
3	El password debe tener al menos 6 caracteres, una letra en mayúscula, un símbolo y un número.
4	Los campos de entrada de datos se deben validar antes de ejecutarlos o guardarlos.

Prueba	Login	Password	Resultado esperado
1			Login o password vacíos
2	123456789	123456789	Login o password incorrecto
3	usuario	Secret()22	Login exitoso
4	prueba	SELECT * FROM Users WHERE Username='1' OR '1' = '1' AND Password='1' OR '1' = '1'	Login o password incorrecto

Detalles de una historia de usuario

DoD (Definition of Done): <<Definición de hecho>>

DoD (Definition of Done): Ejemplo para cada historia de usuario

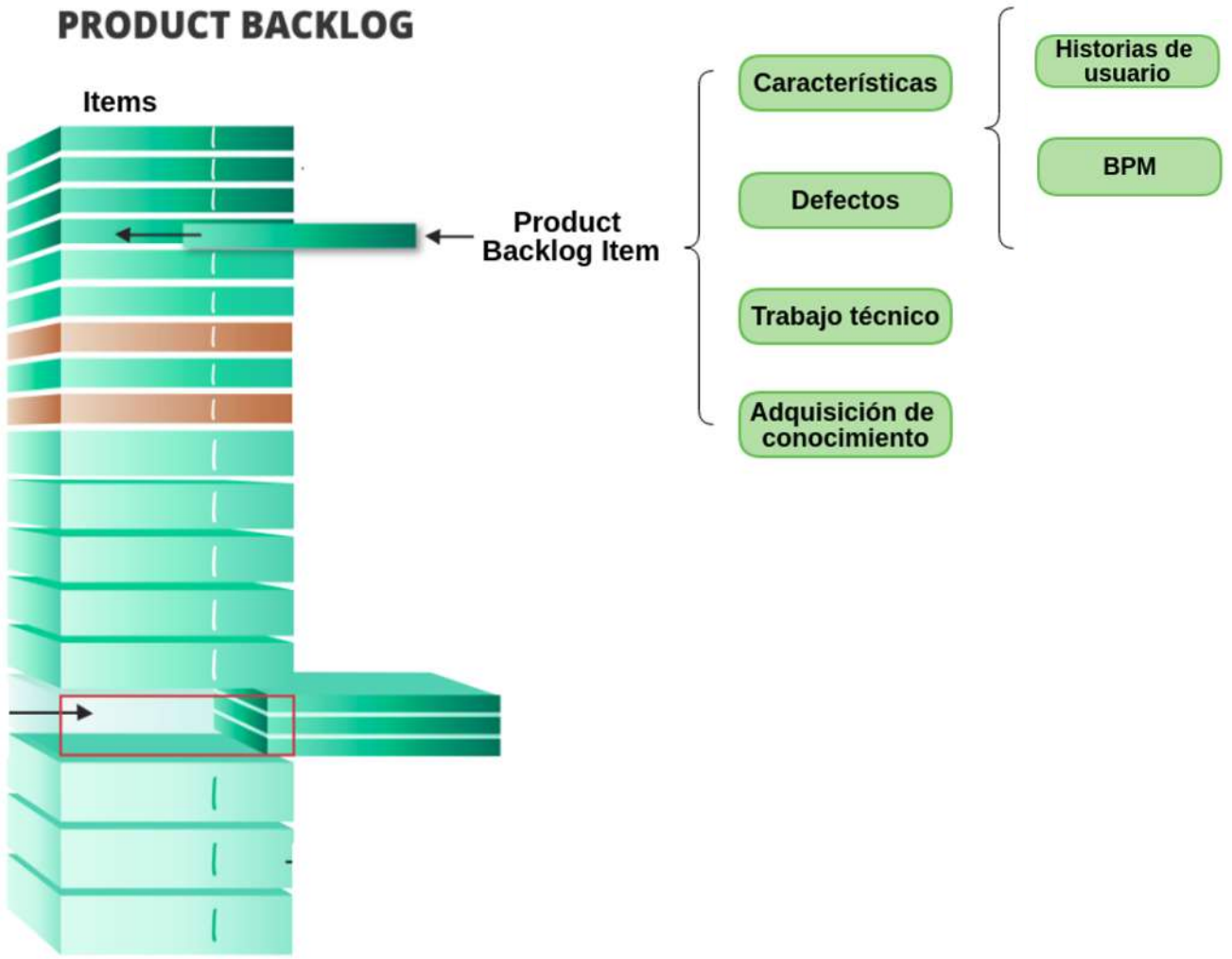
1. El diseño de la historia fue aprobada (Product owner y equipo de desarrollo)
2. La historia esta terminada a criterio del desarrollador
3. La historia paso las pruebas según los criterios de aceptación en el ambiente de desarrollo.
4. La historia paso las pruebas definidas en el ambiente de pruebas y de preproducción
5. Se realizó la documentación (técnica y manuales de usuario si se requiere)
6. [OK de almenos un integrante del equipo de desarrollo y del Product Owner; se verifica si satisface el requerimiento y si el código sigue es estándar de codificación acordado.]
7. Esta todo preparado para la subirla a producción.
8. La historia esta marcada como terminada en la herramienta de control

Historia de usuario

RESUMEN

Product BackLog

Lista priorizada y estimada de las hitorias de usuario



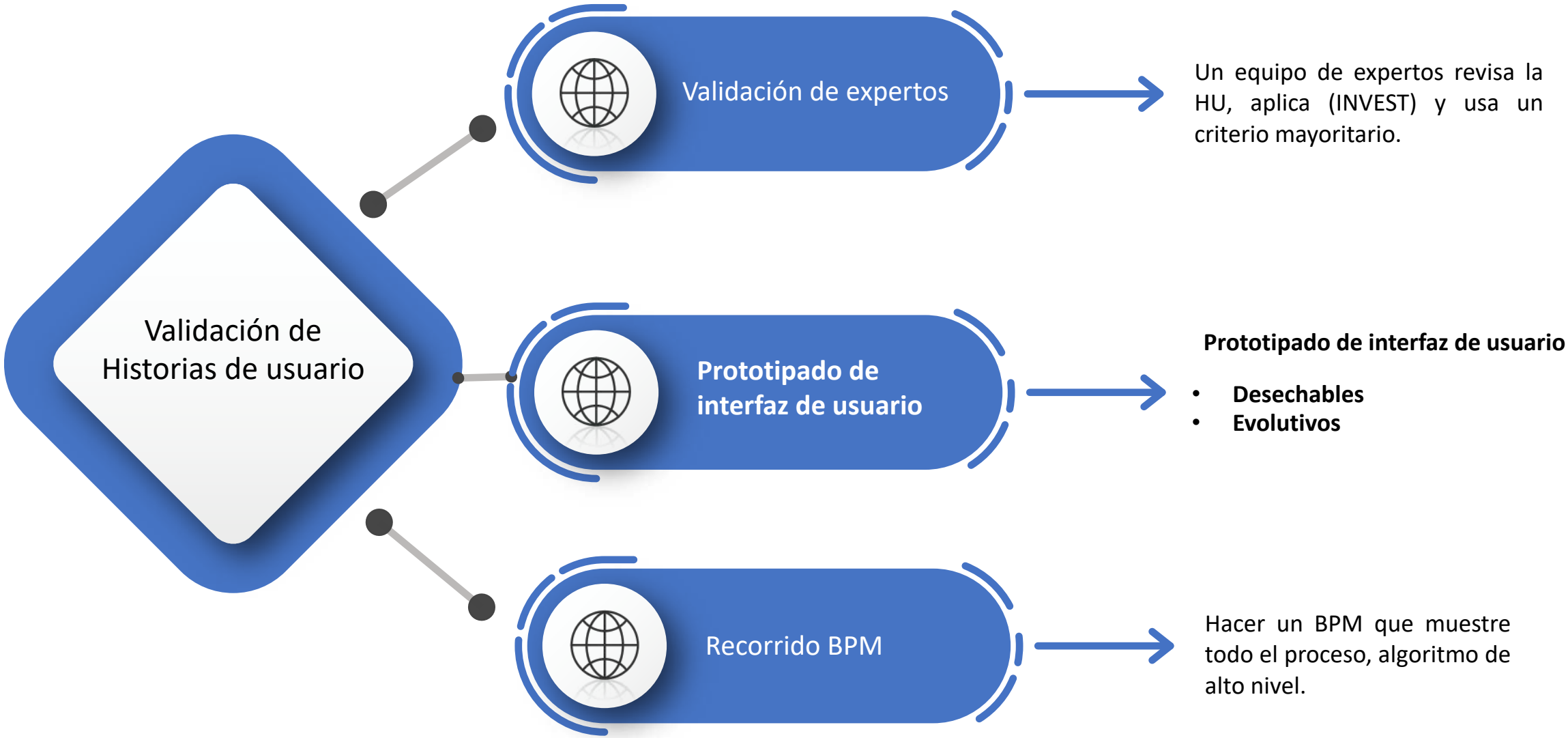
Product Backlog

RESUMEN

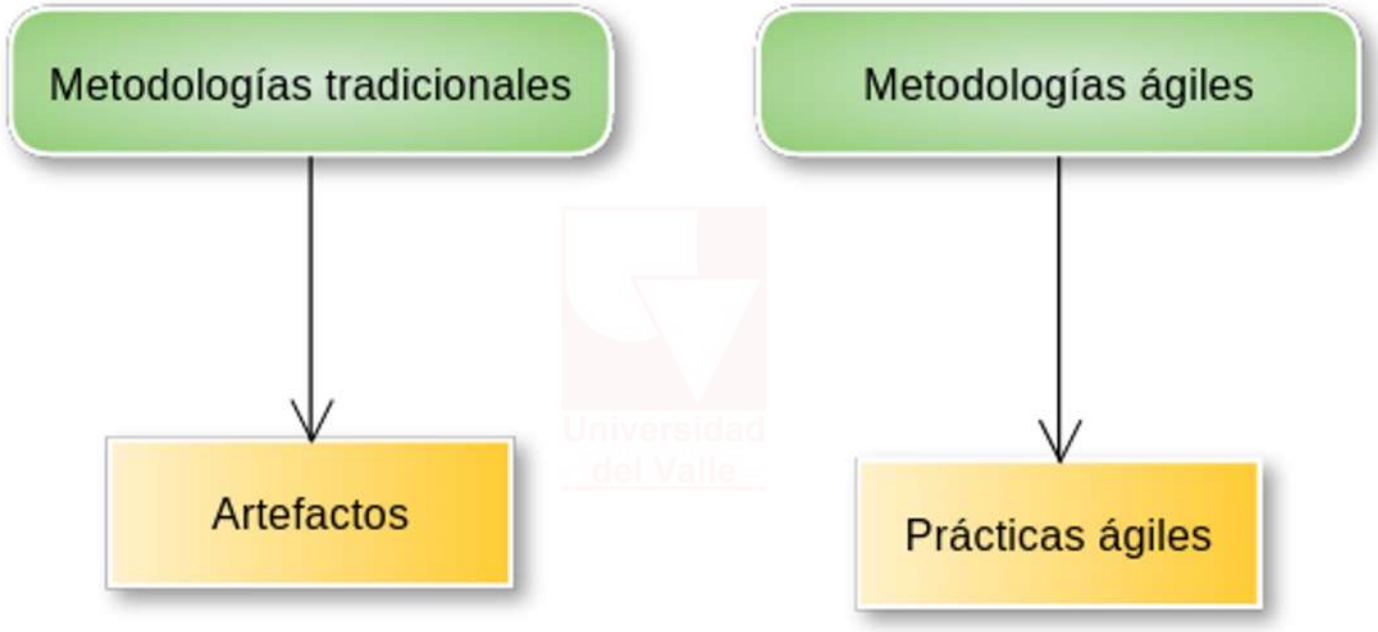
Módulo / Épica	ID	Nombre	Descripción	Prioridad	Estimación
Gestión de usuarios	HU-01	Admin Sistema CREATE	Como administrador del sistema debo poder registrar cualquier tipo de usuario en la aplicación (Subcontratista, o trabajador)	Alta	3
	HU-02	Admin Sistema DELETE	Como administrador del sistema debo poder borrar cualquier tipo de usuario en la aplicación (Subcontratista, o trabajador)	Alta	2
	HU-03	Admin Sistema UPDATE	Como administrador del sistema debo poder modificar cualquier tipo de usuario en la aplicación (Subcontratista, o trabajador)	Alta	2
	HU-04	Admin Sistema READ	Como administrador del sistema debo poder observar (listar) los usuarios registrados en la aplicación (Subcontratista, o trabajador)	Alta	3
	HU-05	Admin Sistema LOGIN	Como administrador del sistema deseo poder iniciar sesión en la aplicación usando mis credenciales	Alta	2
	HU-06	Subcontratista LOGIN	Como subcontratista deseo poder iniciar sesión en la aplicación usando mis credenciales	Alta	2
	HU-07	Trabajador LOGIN	Como trabajador deseo poder iniciar sesión en la aplicación usando mis credenciales	Alta	2
	HU-08	Admin Sistema LOGOUT	Como administrador del sistema deseo poder cerrar sesión en la aplicación	Alta	2
	HU-09	Subcontratista LOGOUT	Como subcontratista deseo poder cerrar sesión en la aplicación	Alta	2
	HU-10	Trabajador LOGOUT	Como trabajador deseo poder cerrar sesión en la aplicación	Alta	2
Gestión de avances de obra	HU-11	Obra STATE READ	Como Subcontratista deseo visualizar los avances de obra publicados por los trabajadores durante el desarrollo de la obra	Alta	4
	HU-12	Obra formulario CREATE	Como Jefe de obra deseo poder registrar mediante formularios el avance de la obra	Media	3
	HU-13	obra audio CREATE	Como técnico, arquitecto residente deseo poder registrar mediante notas de voz el avance de la obra	Media	3
	HU-14	Obra fotos CREATE	Como técnico, arquitecto residente deseo poder registrar mediante fotos el avance de la obra	Media	5
Mapas	HU-15	Mapa READ	Como subcontratista o trabajador deseo poder visualizar en un mapa mis obras en proceso	Media	4
	HU-16	Planos READ	Como subcontratista o trabajador deseo poder visualizar los planos de mis obras en proceso	Media	4
Gestión de Inventarios (almacén)	HU-17	Almacén STATE	Como jefe de almacen deseo conocer el inventario actual de los materiales	Baja	3
	HU-18	Almacén SUPPLY	Como jefe de almacen deseo tener acceso a la lista con los distintos distribuidores	Baja	3
	HU-19	Almacen NOTIFY	Como jefe de almacen deseo recibir una alarma cuando el material este cerca de acabarse (ejemplo solo 10% de disponibilidad) y se requiera más de lo disponible para finalizar la obra.	Baja	3
Flujo de trabajo de obras	HU-20	Solicitud CREATE	Como Jefe de obra, deseo solicitar al Jefe de almacén materiales para desarrollar una obra	Baja	3
	HU-21	Solicitud STATUS	Como Jefe de almacen, deseo aceptar o rechazar una solicitud de materiales por parte de un jefe de obra	Baja	2

Validación de historias de usuario

RESUMEN



Concepto



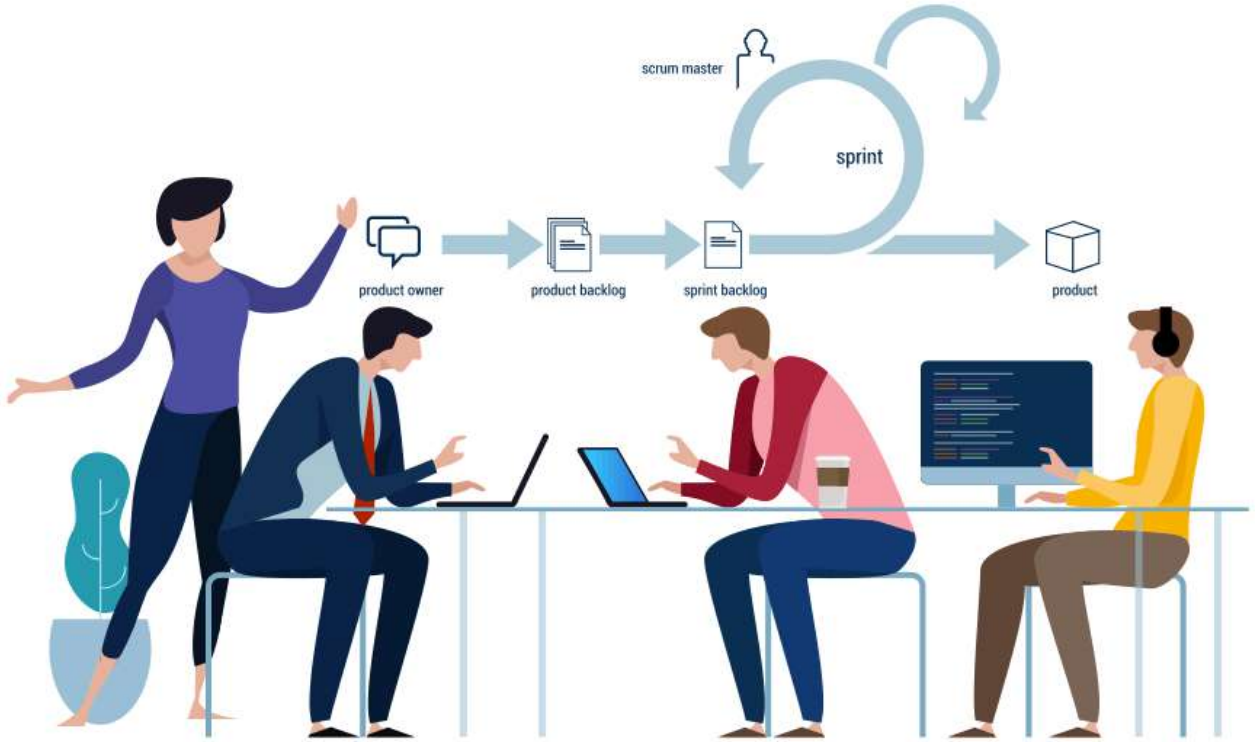
Requerimiento
Requerimientos nofuncionales
SRS, SRD
…

Historia de Usuario
Restricciones
Product Backlog
…

Preguntas ?

Metodologías ágiles

Metodologías ágiles



Desarrollo de software ágil

Ágil es un término que se usa para describir los enfoques iterativos del desarrollo de software que enfatizan en la entrega incremental, la colaboración en equipo, la planificación continua y el aprendizaje continuo, en lugar de intentar entregarlo todo de una vez cerca del final.

Desarrollo de software ágil

El desarrollo de software ágil representa una forma de pensar y hacer las cosas.

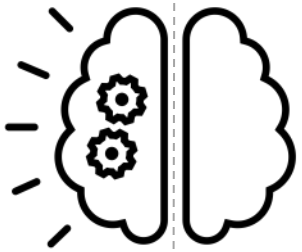
Ágil es una **Mentalidad**

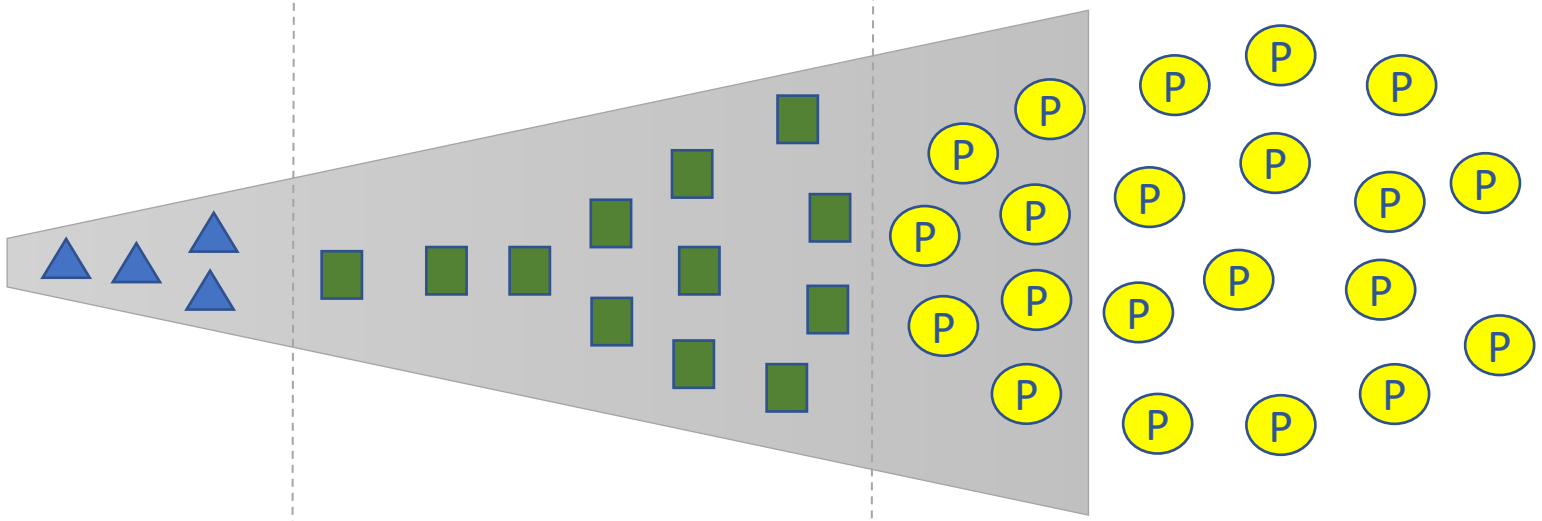
Descrita por **4 Valores**

Definida por **12 Principios**

Manifestada a través de un **Número ilimitado de prácticas**

Ser Ágil





Agilismo

Desarrollo de software ágil

- El Desarrollo ágil de software es un paradigma usado en las metodologías de desarrollo de software basado en procesos ágiles.
- Se concibieron como una alternativa a las prácticas de desarrollo de software tradicional.
- Es una “sombrilla” para un conjunto de valores, principios y prácticas.
- El desarrollo ágil es una forma diferente de gestionar equipos y proyectos de desarrollo de Software.
- Una metodología ágil no es hacer lo mismo más rápido y tampoco es que nunca más vamos a planear y hacer contratos. (scrumcolombia)

Manifiesto ágil

Manifiesto ágil
"Biblia"

4 Valores del Manifiesto ágil

Manifiesto Agil *(Agile Manifesto)*

(a.k.a. Manifiesto por el Desarrollo Ágil de Software)

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

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

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

Web: http://agilemanifesto.org/ ...en español: http://agilemanifesto.org/iso/es/

12 Principios definidos en el manifiesto ágil ?

01 Nuestra mayor prioridad es satisfacer al cliente mediante la entrega temprana y continua de software con valor.

02 Aceptamos que los requisitos cambien, incluso en etapas tardías del desarrollo.

03 Entregamos software funcional frecuentemente, entre dos semanas y dos meses, con preferencia por periodos de tiempo lo más corto posibles.

04 Los responsables de negocio y los desarrolladores trabajamos juntos de forma cotidiana durante todo el proyecto.

05 Los proyectos se desarrollan en torno a individuos motivados. Hay que darles el entorno y el apoyo que necesitan, y confiarles la ejecución del trabajo.

06 El método más eficiente y efectivo de comunicar información al equipo de desarrollo y entre sus miembros es la conversación cara a cara (Presencial o virtual).

12 Principios definidos en el manifiesto ágil ?

07 El software funcionando es la medida principal de progreso.

08 Los procesos Ágiles promueven el desarrollo sostenible.

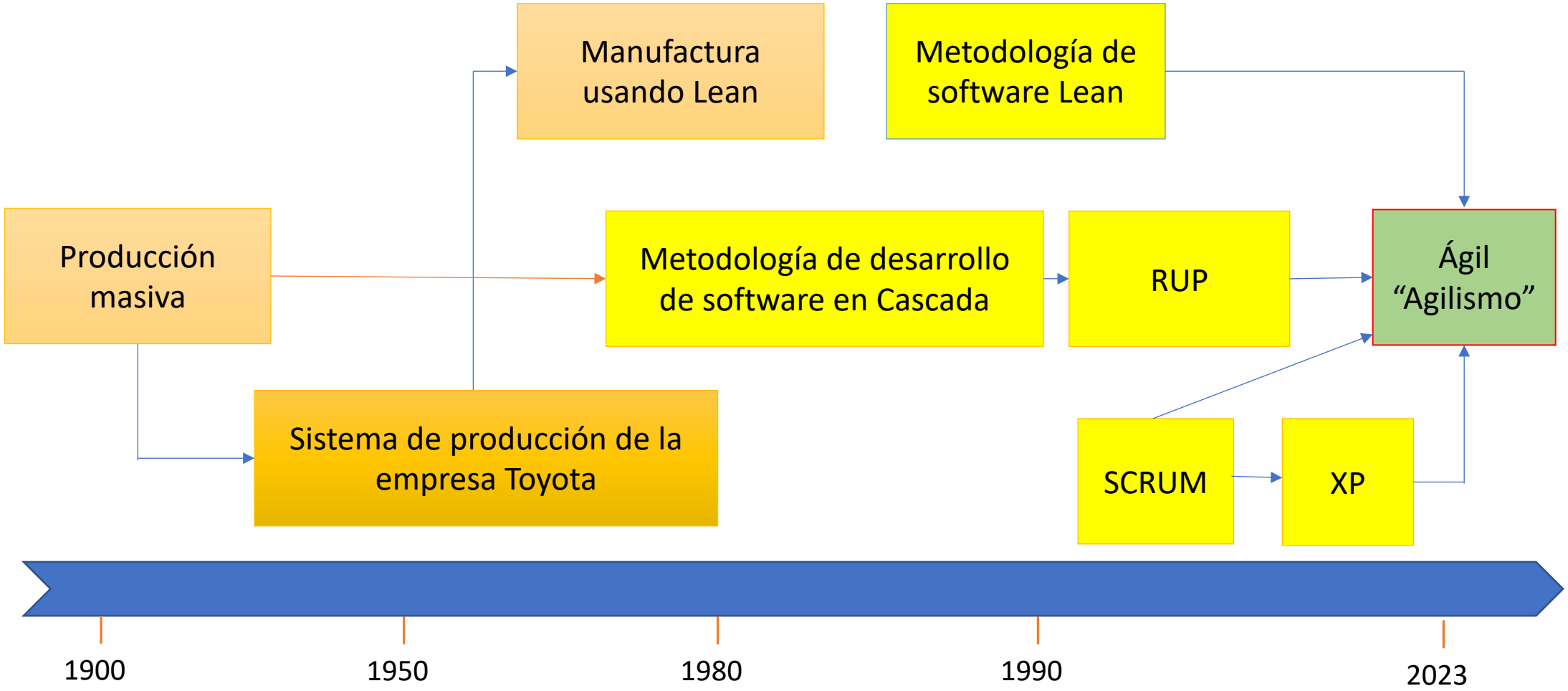
09 La atención continua a la excelencia técnica y al buen diseño mejora la Agilidad.

10 La simplicidad, o el arte de maximizar la cantidad de trabajo no realizado, es esencial.

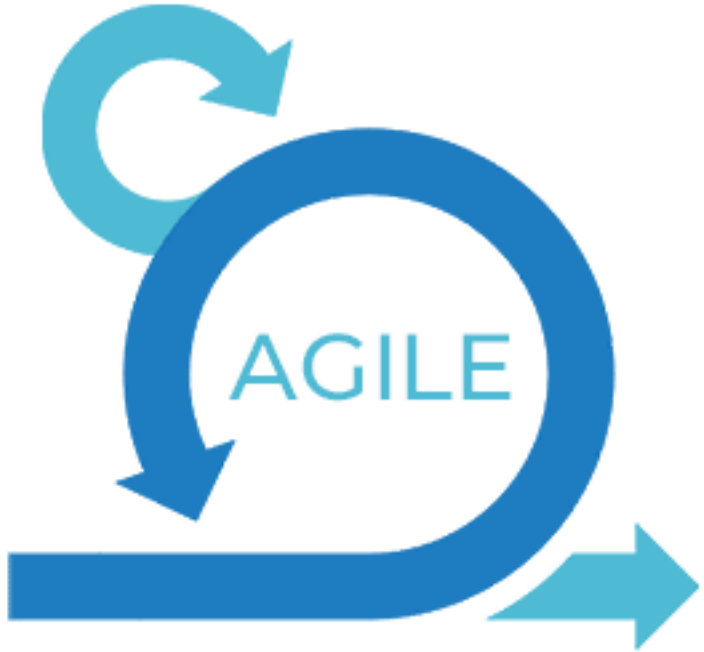
11 Las mejores arquitecturas, requisitos y diseños emergen de equipos auto-organizados.

12 A intervalos regulares el equipo reflexiona sobre cómo ser más efectivo para a continuación ajustar y perfeccionar su comportamiento en consecuencia.

Historia de las metodologías ágiles



Metodologías ágiles



Scrum

Metodología ágil de desarrollo de software.

Scrum es un proceso en el que se aplican de manera regular un conjunto de buenas prácticas para trabajar colaborativamente, en equipo, y obtener el mejor resultado posible de un proyecto de desarrollo de software.

Las fases en las que se divide y define un proceso de SCRUM son las siguientes:

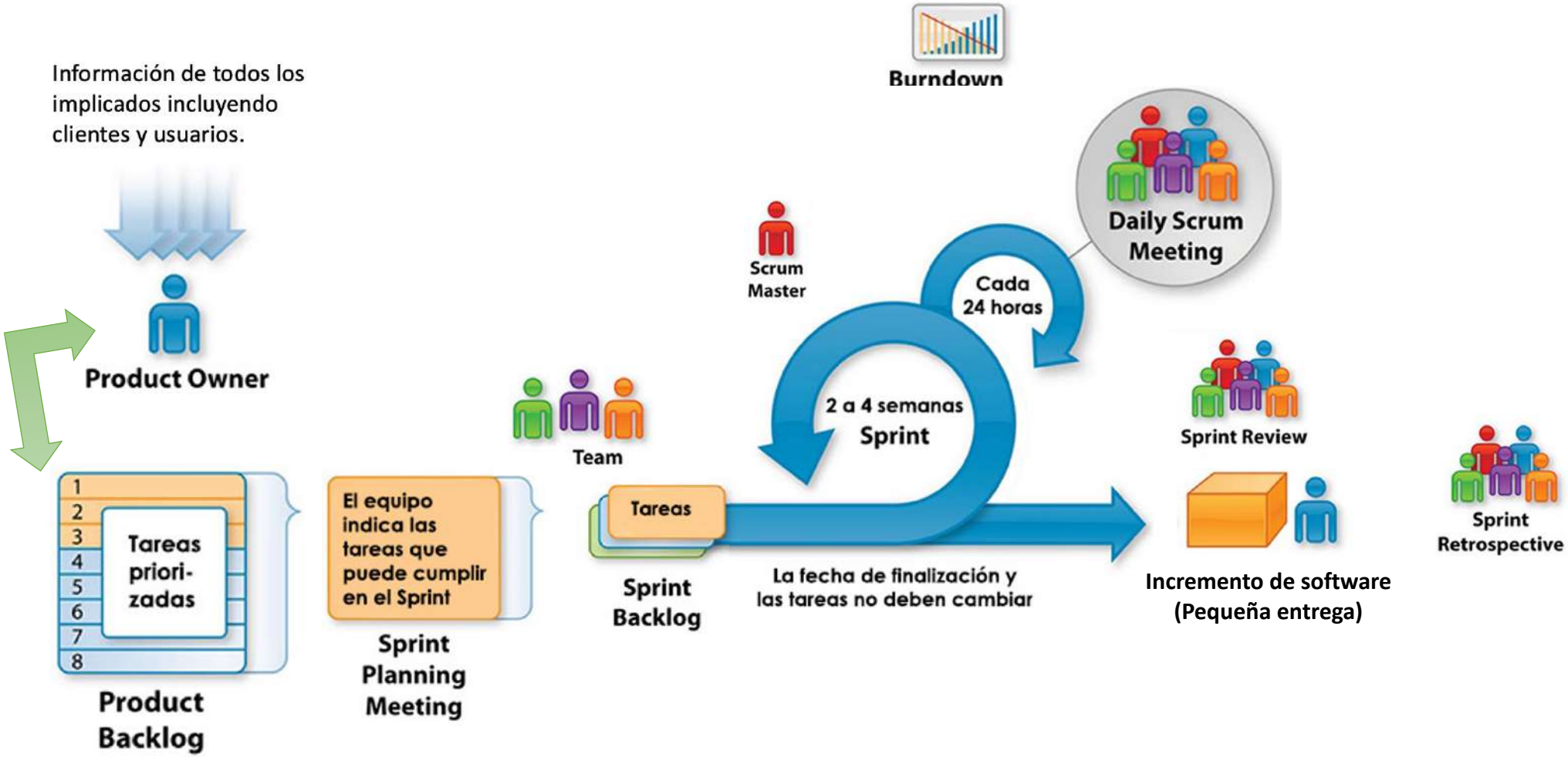
El ¿**Quién**? y el ¿**Qué**?: Identifica los roles de cada uno de los miembros del equipo y define su responsabilidad en el proyecto.

El ¿**Dónde**? y el ¿**Cuándo**?: que representan el Sprint (iteración)

El ¿**Por qué**? y el ¿**Cómo**?: representan las herramientas que utilizan los miembros de Scrum

Conceptos

Metodología ágil Scrum



PRÁCTICAS ÁGILES EN SCRUM

Historias de Usuario: Scrum utilizan las historias de usuario como el instrumento principal para identificar los requerimientos de usuario. Las historias de usuario son descripciones cortas y simples de una funcionalidad, escritas desde la perspectiva de la persona que necesita una nueva funcionalidad de un sistema, por lo general el usuario, área de negocio o cliente.

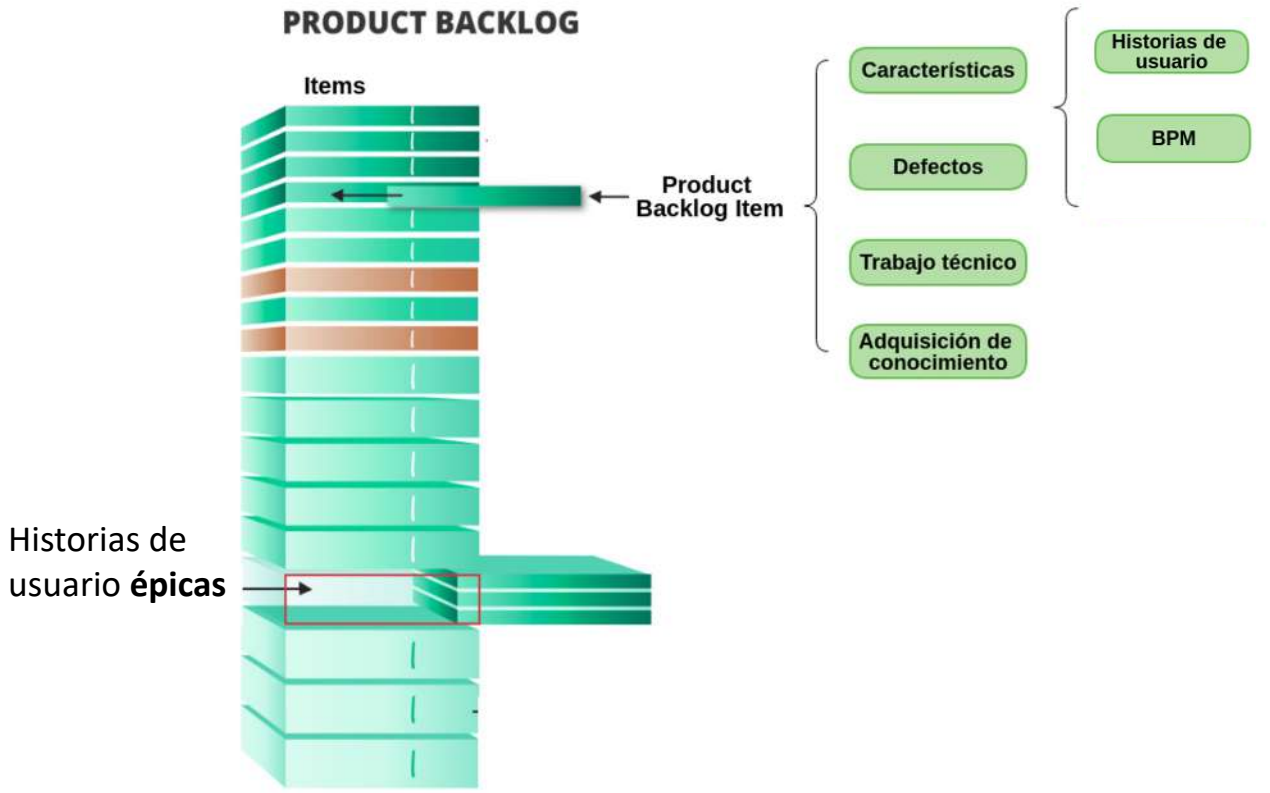
Es la práctica ágil que permite identificar las funcionalidades que se desean hacer en el producto de software a desarrollar.

Para documentar una Historia de Usuario se utiliza el formato propuesto por Mike Cohn

Como estudiante deseo poder registrar un curso para matricularme en la universidad

PRÁCTICAS ÁGILES EN SCRUM

Product Backlog : Una lista priorizada y estimada de las historias de usuario, que representan las funcionalidades del sistema que se va a construir; en algunas ocasiones se incluyen otros aspectos requeridos durante las etapas de desarrollo del sistema.



Spyke: Es un tipo de HU en los cuales el equipo no conoce y se debe hacer una investigación para poder enterder, estimar y priorizar.

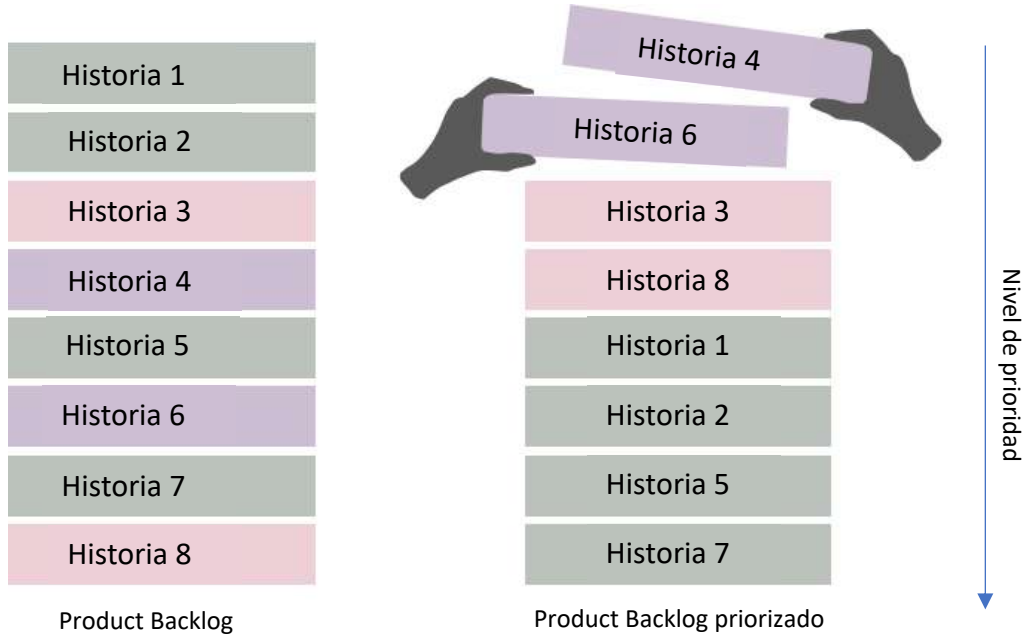
PRÁCTICAS ÁGILES EN SCRUM

Priorización de las Historias de Usuario

¿Qué es priorizar?

La priorización como principio significa "hacer lo primero". Como proceso, significa "evaluar un grupo de elementos y clasificarlos en orden de importancia o necesidad".

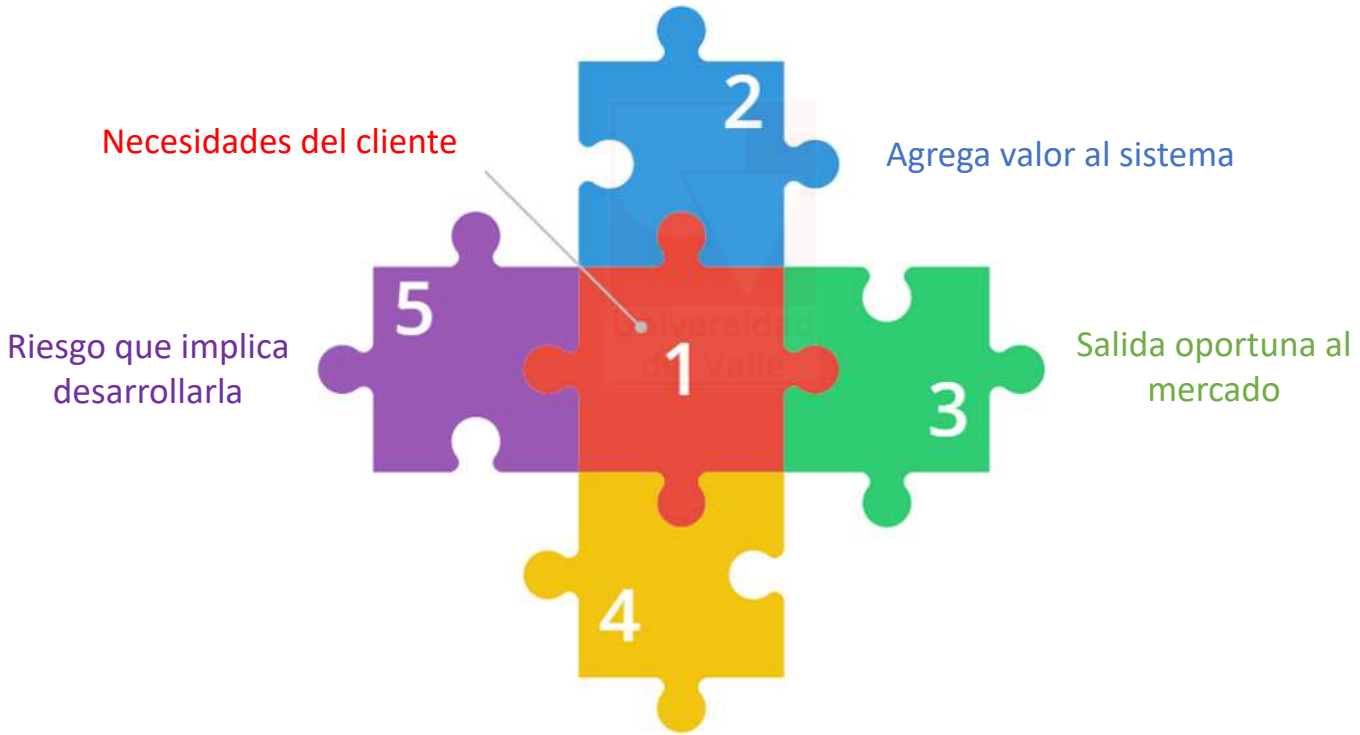
La priorización de requerimientos en todas las metodologías de desarrollo de software se considera una parte vital del proyecto, pero es especialmente importante en el desarrollo de software Agile.



PRÁCTICAS ÁGILES EN SCRUM

Priorización de HU

Criterios para priorizar



PRÁCTICAS ÁGILES EN SCRUM

Priorización de HU

Técnica de clasificación de lista

Esta técnicas es un ordenamiento simple y que no requieren ningún tipo de entrenamiento ni preparación.

En la gestión de Proyectos, se trata de indicar a cada HU un orden de prioridad, empezando por el 1, luego el 2, 3, y continuando hasta n, que es la cantidad total de HU.

Ventajas	Desventajas
• Solo puede haber un número uno • Aporta precisión y evita la confusión	• Requiere un conocimiento profundo de todas las HU definidas y un esfuerzo grande por parte del equipo, para situar cada una de las HU en la posición correcta.

PRÁCTICAS ÁGILES EN SCRUM

Priorización de HU

Técnica de clasificación por rangos

Esta técnicas es un ordenamiento simple.

Para asignar una historia de usuario a un rango se aplican los criterios de prioridad.

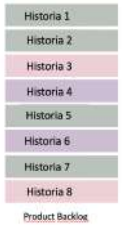
Listas de tres rangos: Alta, Media y Baja

#	Historia de Usuario	Prioridad
1	HU1	**ALTA**
2	HU2	**MEDIA**
3	HU3	**ALTA**
4	HU4	**ALTA**
5	HU5	**BAJA**
7	HU6	**MEDIA**
8	HU7	**BAJA**
9	HU8	**ALTA**

#	Historia de Usuario	Prioridad
1	HU1	**ALTA**
2	HU3	**ALTA**
3	HU4	**ALTA**
4	HU8	**ALTA**
5	HU2	**MEDIA**
7	HU6	**MEDIA**
8	HU5	**BAJA**
9	HU7	**BAJA**

PRÁCTICAS ÁGILES EN SCRUM

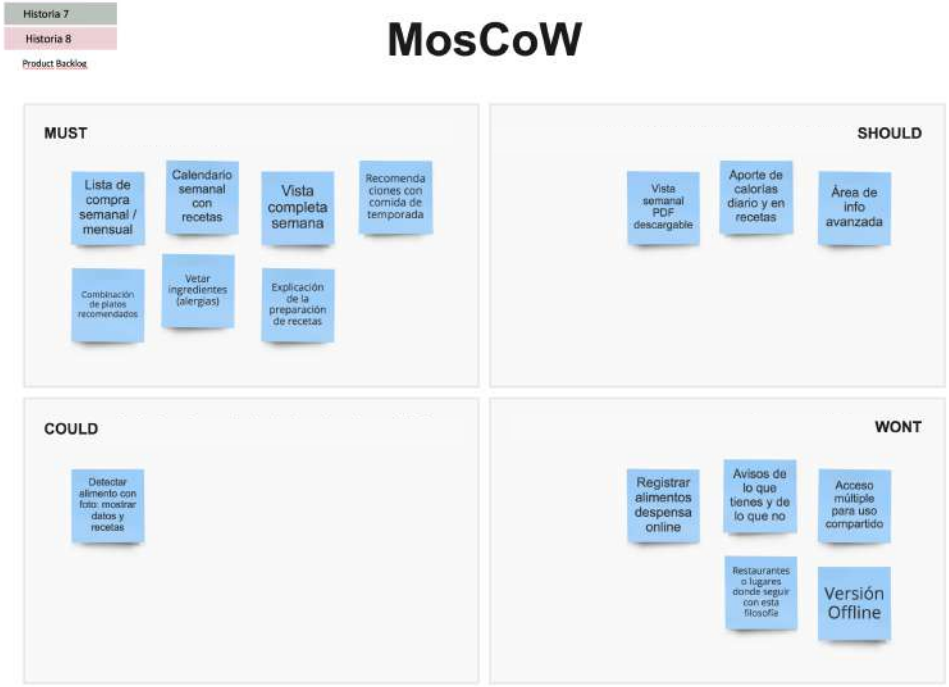
Priorización de HU



MoSCoW es una técnica que aporta un valor semántico de lo que realmente es importante.

A diferencia de las técnicas de rangos, se ha desarrollado esta técnica que plantea una categorización de las HU en función a palabras que tengan un significado concreto:

- M: Esta funcionalidad debe estar (MUST).
  No negociable, la aplicación la requiere es vital.
- S: Esta funcionalidad debería estar (SHOULD).
  Cosas importantes pero no vitales.
- C: Esta funcionalidad podría estar (COULD).
  Cosas de bajo impacto en la aplicación
- W: Esta funcionalidad no estará ahora, quizás en un futuro (WON'T).
  Cosas no prioritarias y aportan poco valor



PRÁCTICAS ÁGILES EN SCRUM

Priorización de HU usando MosCow

Ejemplo: Funcionalidades requeridas para un sistema

- como usuario deseo registrarse en el sistema para apoder usarlo
- Como usuario deseo iniciar sesión para poder ingresar al sistema
- Como usuario deseo restablecer la contraseña para tener control de acceso al sistema
- Como usuario deseo elegir la forma de pago para tener diferentes opciones de pago
- Como usuario deseo eliminar la cuenta para retirarme del sistema
- Como usuario deseo abrir una página de seguimiento de un producto para hacer seguimiento
- Como usuario deseo elegir opciones de seguimiento de un producto para saber donde esta en cada momento
- Como usuario deseo tener una versión de móvil de la aplicación para usarla desde el teléfono
- Como usuario deseo elegir el tema visual de la aplicación para tener diferentes formas de ver la aplicación.

PRÁCTICAS ÁGILES EN SCRUM

Priorización de HU usando MosCow

#	Historia de Usuario	Prioridad
1	Como usuario deseo registrarse en el sistema para apoder usarlo	**MUST**
2	Como usuario deseo iniciar sesión para poder ingresar al sistema	**MUST**
3	Como usuario deseo restablecer la contraseña para tener control de acceso al sistema	**MUST**
4	Como usuario deseo abrir una página de seguimiento de un producto para hacer seguimiento	**MUST**
5	Como usuario deseo elegir la forma de pago para tener diferentes opciones de pago	**SHOULD**
6	Como usuario deseo eliminar la cuenta para retirarme del sistema	**SHOULD**
7	Como usuario deseo elegir opciones de seguimiento de un producto para saber donde esta en cada momento	**SHOULD**
8	Como usuario deseo tener una versión de móvil de la aplicación para usarla desde el teléfono	**COULD**
9	Como usuario deseo elegir el tema visual de la aplicación para tener diferentes formas de ver la aplicación.	**WON'T**

PRÁCTICAS ÁGILES EN SCRUM

Estimar Historias de Usuario

Determinar el esfuerzo, complejidad y riesgo que se requiere para desarrollar una HU.

La estimación se mide en puntos (Puntos de Historias de Usuario)

Un Punto de HU mide:

a) La cantidad de esfuerzo que supone desarrollar la historia de usuario
b) la complejidad de su desarrollo y
c) el riesgo inherente.

Nota: Un punto de HU = X horas
Ejemplo: 1 punto = 8 horas. (Esto de pende de las características de cada equipo de desarrollo)

PRÁCTICAS ÁGILES EN SCRUM

Estimar Historias de Usuario

¿Qué es estimar?

Estimar es una forma de cuntificar la magnitud del proyecto y de las tareas que tenemos que asumir, a hacernos una idea aproximada del tiempo y recursos que vamos a consumir.
Importante: las estimaciones son sólo valores aproximados, no son valores exactos.

La estimación en la metodología ágil consiste en asignar puntos a una tarea o Historia de usuario.
No hay una fórmula para realizarlo de una manera exacta.

La estimación ágil se basa en la estimación relativa, y una técnica colaborativa, sencilla, divertida y efectiva de poder estimar historias de usuario es la de **Planning Poker.**

PRÁCTICAS ÁGILES EN SCRUM

Estimar Historias de Usuario

Planning Poker

El objetivo del *planning poker* es obtener una medida de **tamaño relativo** de todas las historias respecto una historia base.

Planning Poker es una técnica de estimación puesta en marcha por primera vez por James Grenning en un equipo Ágil utilizando XP en 2002, donde se utiliza una baraja de cartas con una distribución de números muy parecida a la secuencia de Fibonacci (0, 1/2, 1, 2, 3, 5, 8, 13, etc.).

PRÁCTICAS ÁGILES EN SCRUM

Estimar Historias de Usuario

Planning Poker

Para realizar una sesión de planning poker hay que seguir los siguientes pasos:

1. Reunir a todo el equipo y repartir las barajas a cada miembro o (iniciar la App).

2. El product owner, o moderador, lee una historia de usuario, y responde cualquier duda que tengan los miembros del equipo de desarrollo. Esto sirve para que todos tengan una comprensión en común de lo que hay que desarrollar.

3. Cada miembro del equipo de desarrollo selecciona una carta, equivalente a la estimación que el considera adecuada, y la pone boca abajo, cuando todos tengan seleccionada una carta, se ponen boca arriba todas a la vez.

4. Una vez mostradas las cartas, nos quedamos con la estimación media más elegida (moda), o se debate hasta conseguir la unanimidad.

Esto se repite con cada historia de usuario que haya que estimar.

PRÁCTICAS ÁGILES EN SCRUM

Estimar Historias de Usuario

Ventajas Planning Poker

- Estimar colaborativamente ayuda a detectar riesgos, que se pueden ir solventando antes de comenzar la historia de usuario.
- Promueve la participación y la colaboración.
- Abstrae el concepto de tiempo.
- Estimar entre varias personas nos acerca al máximo a la realidad, evitando márgenes de error muy grandes que afecten al desarrollo de producto.

Desventajas Planning Poker

- El llegar a un consenso, no garantiza el acierto
- Requiere de un histórico de información
- Cultura

PRÁCTICAS ÁGILES EN SCRUM

Estimar Historias de Usuario

Ventajas Planning Poker

- Estimar colaborativamente ayuda a detectar riesgos, que se pueden ir solventando antes de comenzar la historia de usuario.
- Promueve la participación y la colaboración.
- Abstrae el concepto de tiempo.
- Estimar entre varias personas nos acerca al máximo a la realidad, evitando márgenes de error muy grandes que afecten al desarrollo de producto.

Desventajas Planning Poker

- El llegar a un consenso, no garantiza el acierto
- Requiere de un histórico de información
- Cultura

PRÁCTICAS ÁGILES EN SCRUM

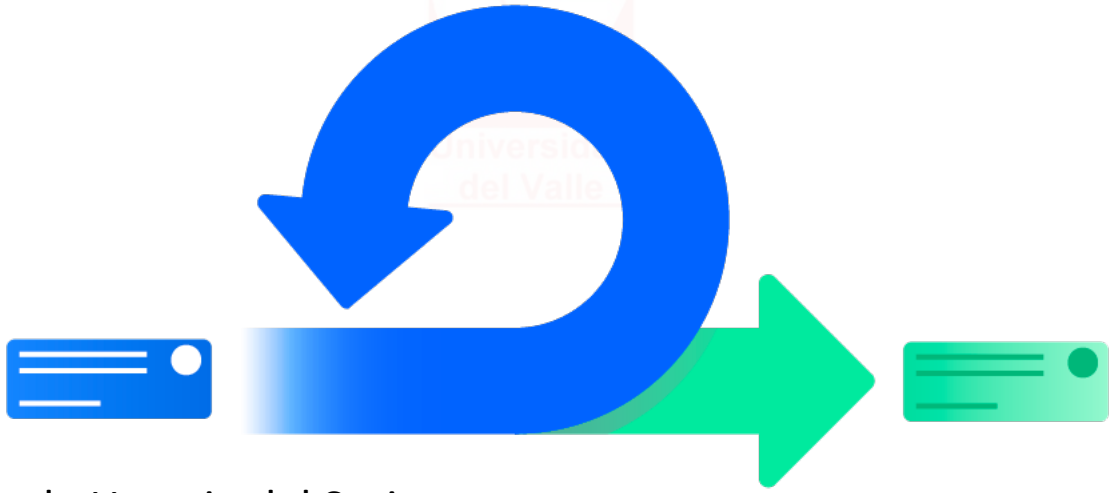
Estimación de HU usando Planning Poker

#	Historia de Usuario	Prioridad	Estimación
1	Como usuario deseo registrarse en el sistema para apoder usarlo	MUST	**1**
2	Como usuario deseo iniciar sesión para poder ingresar al sistema	MUST	**2**
3	Como usuario deseo restablecer la contraseña para tener control de acceso al sistema	MUST	**2**
4	Como usuario deseo abrir una página de seguimiento de un producto para hacer seguimiento	MUST	**2**
5	Como usuario deseo elegir la forma de pago para tener diferentes opciones de pago	SHOULD	**1**
6	Como usuario deseo eliminar la cuenta para retirarme del sistema	SHOULD	**1**
7	Como usuario deseo elegir opciones de seguimiento de un producto para saber donde esta en cada momento	SHOULD	**2**
8	Como usuario deseo tener una versión de móvil de la aplicación para usarla desde el teléfono	COULD	**24**
9	Como usuario deseo elegir el tema visual de la aplicación para tener diferentes formas de ver la aplicación.	WON'T	**2**

Recomendación: Si una historia de usuario tiene más de 8 puntos divídale en varias historias.

PRÁCTICAS ÁGILES EN SCRUM

Sprint : Es un intervalo de tiempo prefijado durante el cual se realizan HU y el resultado es un incremento de software, **potencialmente entregable**. Un sprint inicia con una planeación de las HU a realizar. Un sprint típicamente dura de 1 a 4 semanas.



Historias de Usuario del Sprint
"Sprint Backlog"

Incremento
de software

Durante el sprint a cada HU se le hace el Análisis, Diseño, Codificación, Pruebas y Despliegue.

REUNIONES EN SCRUM

Reuniones o ceremonias que se realizan en un Sprint

PRÁCTICAS ÁGILES EN SCRUM

Sprint Planning

Es el primer evento de Scrum en dónde se planifican las tareas a realizar en el Sprint en curso.
En esta reunión participan, de manera colaborativa, todo el equipo Scrum: Scrum Master, Product Owner y Equipo de Desarrollo.

- El tiempo de esta reunión es de máximo 8 horas para Sprints de 4 semanas de duración. Para Sprints de menor duración, esta reunión debe proporcionalmente ser más corta. El Scrum Master es el encargado de asegurar que esta reunión se realice, se enseñe la importancia de la misma, y además debe asegurarse de que se realiza en el tiempo establecido.

- La labor del Product Owner es la de describir las tareas con mayor prioridad al resto del equipo. El equipo de desarrollo pregunta todo lo necesario para convertir estas historias de usuario en tareas más específicas.

El Sprint Planning responde a las siguientes preguntas:
¿Qué se puede hacer en este Sprint? Objetivo del Sprint
¿Cómo haremos el trabajo elegido? Equipo auto organizado define como realizara los items del sprint

PRÁCTICAS ÁGILES EN SCRUM

PRÁCTICAS ÁGILES EN SCRUM

Reunión diaria (Daily Scrum) : Es un evento de 15 minutos, cuyo objetivo es que el equipo de desarrollo sincronice actividades y cree un plan para las próximas 24 horas. Los integrantes del equipo responden las siguientes preguntas:

¿Qué has hecho desde la última reunión?
¿Qué problemas has encontrado para realizar el trabajo previsto?
¿Qué planeas hacer antes de la próxima reunión?

- 15 Minutos
- De pie
- Sin café

PRÁCTICAS ÁGILES EN SCRUM

Sprint Review

El *Sprint Review* es la reunión que ocurre al final del Sprint, donde el *product owner* y el *Equipo de Desarrollo* presentan a los *stakeholders* el incremento terminado para su inspección y adaptación correspondiente.

Se revisará el incremento terminado. Se mostrará el software funcionando y los *stakeholders* tendrán la oportunidad de hacer cuantas preguntas estimen oportunas sobre el mismo.

Es una gran oportunidad para poder recibir feedback sobre el desarrollo del producto y podría conducir a actualizar el *Product Backlog.*

El software funcionando ha sido validado previamente por el *product owner*, que se ha encargado de trabajar con el equipo durante el Sprint para asegurarse que cumple con la *Definition of Done (DoD).*

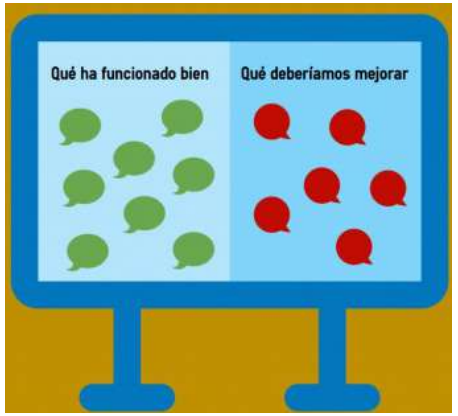
La duración típica es de 4 horas para un sprint de 4 semanas.

PRÁCTICAS ÁGILES EN SCRUM

Retrospectiva

El objetivo es analizar cómo les fue en el último sprint, cómo trabajamos, qué problemas tuvimos, qué cosas funcionaron bien y cuáles no. Participan el equipo de desarrollo y el scrum master.

El equipo y el Scrum master analizan cómo ha sido su manera de trabajar durante el sprint, por qué está consiguiendo o no los objetivos a que se comprometió al inicio del sprint y si el incremento de producto que acaba de demostrar al cliente era lo que él esperaba o no:



- Qué cosas han funcionado bien.
- Cuales hay que mejorar.
- Qué cosas quiere probar hacer en la siguiente iteración.
- Qué ha aprendido.
- Cuales son los problemas que podrían impedirle progresar adecuadamente. El Scrum master se encargará de ir eliminando los obstáculos identificados que el propio equipo no pueda resolver por sí mismo.

PRÁCTICAS ÁGILES EN SCRUM

Backlog Refinement (Sprint Grooming)

Backlog refinement (anteriormente conocido como Backlog grooming) es cuando el product owner y el equipo revisan los elementos del product backlog para asegurarse de que éste contenga los items adecuados y que estos esten bien priorizados.

Esta actividad ocurre de manera regular y puede ser una reunión programada oficialmente o una actividad continua.

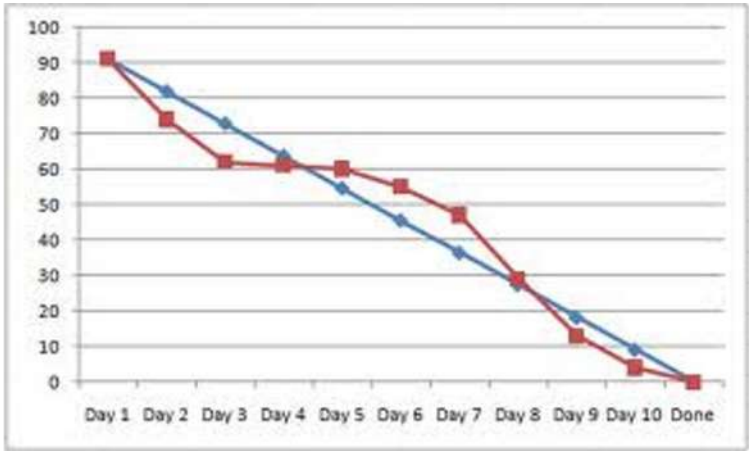
Algunas de las actividades que ocurren durante Backlog refinement incluyen:

- Eliminar historias de usuarios que ya no se requieren.
- Crear nuevas historias de usuario en respuesta a necesidades recién descubiertas.
- Reevaluar la prioridad relativa de las historias.
- Asignar estimaciones a historias.
- Corregir estimaciones a la luz de la información recién descubierta.
- Dividir historias de usuarios que son de alta prioridad pero demasiado generales para caber en una próxima iteración.

PRÁCTICAS ÁGILES EN SCRUM

BurnDown chart

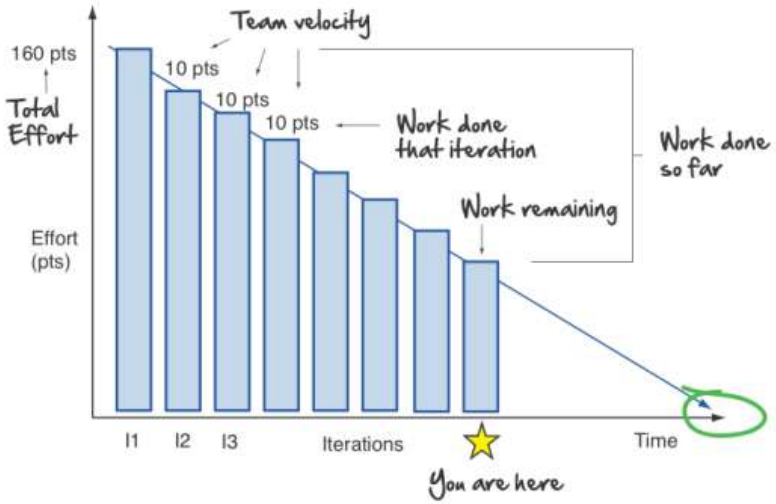
Gráficos de trabajo pendiente (**Burndown charts**)
Un gráfico de trabajo pendiente a lo largo del tiempo muestra la velocidad a la que se está completando los objetivos/HU. Permite extrapolar si el Equipo podrá completar el trabajo en el tiempo estimado.



Nos recuerda cuantas HU quedan pendientes.

Se pueden utilizan los siguientes gráficos de esfuerzo pendiente:

- Días pendientes para completar las HU del producto o proyecto (product burndown chart), realizado a partir de la lista de HU priorizada (Product Backlog).



- Puntos pendientes para completar las HU de la iteración (sprint burndown chart), realizado a partir de la lista de tareas de la iteración.

PRÁCTICAS ÁGILES EN SCRUM: Release plan

Un release plan, es donde se representan los prints que tendrá el proyecto, las HU a desarrollar por sprint y se acuerdan las entregas a realizar. Representa el plan de trabajo para el proyecto

Release Plan

Release:

Versión del producto que se pone a disposición del usuario.

Incremento:

Suma de todos los ítems del Product Backlog completados durante un Sprint.

Sprint	HU	Puntos de HU	Desarrollador
1	HU1 HU2	3 4	Pedro Juan
2	HU3 HU4 HU6	2 4 2	Maria Pedro/Ricardo Juan
3	HU5 HU7	4 3	Maria Juan
4	HU9 HU8	3 3	Pedro Juan
5	HU10 HU11	4 3	Maria Pedro

PRÁCTICAS ÁGILES EN SCRUM

Release Plan

Velocidad

Sprint	HU	Puntos de HU		Desarrollador
1	HU1 HU2	3 4	7	**Pedro/María** **Juan/Ricardo**
2	HU3 HU4 HU6	2 4 2	8	Maria Pedro Juan
3	HU5 HU7	4 3	7	Maria Juan
4	HU9 HU8	3 3	6	Pedro Juan
5	HU10 HU11	4 3	7	Maria Pedro

Road Map
(Hoja de ruta)

Release 1
Octubre 5

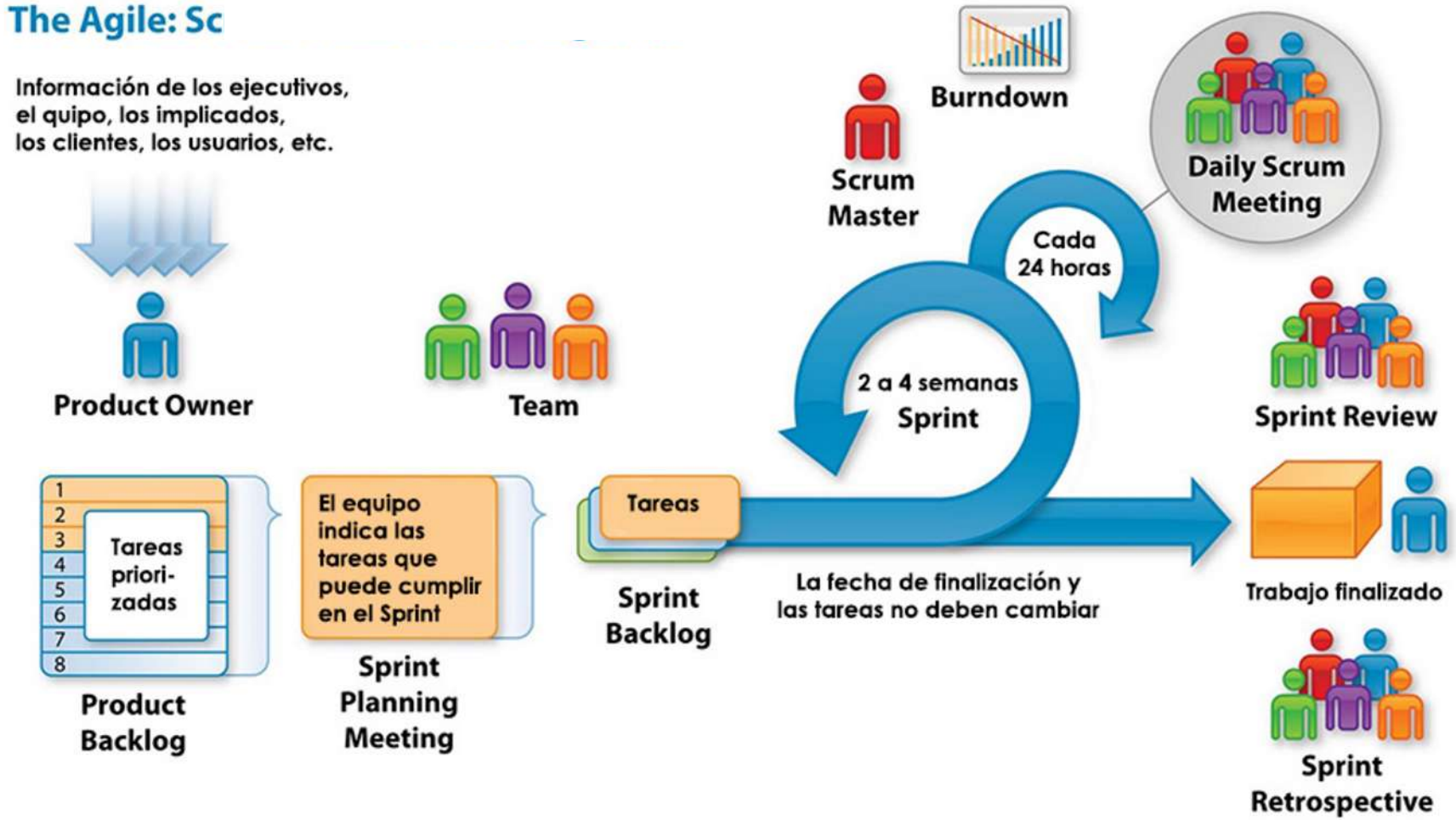
Release 2
Noviembre 5

Release 3
Diciembre 8

Velocidad: Cantidad de puntos que un equipo es capaz de hacer en un sprint

Conclusión

Metodología ágil Scrum



Prácticas técnicas

- UNIT TESTING
- CODING STANDARDS
- CONTINUOUS INTEGRATION
- REFACTORING
- CONTINUOUS DELIVERY
- AUTOMATED ACCEPTANCE TESTING
- CONTINUOUS DEPLOYMENT
- PAIR PROGRAMMING

Preguntas ?

Para la próxima clase:

Actividades de evaluación y como Iniciar un producto de software

Gracias

Universidad del Valle