

Requerimientos Ágiles



MANIFESTO FOR AGILE SOFTWARE DEVELOPMENT

We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:

INDIVIDUALS AND INTERACTIONS —OVER PROCESSES AND TOOLS
WORKING SOFTWARE —OVER COMPREHENSIVE DOCUMENTATION
CUSTOMER COLLABORATION — OVER CONTRACT NEGOTIATION
RESPONDING TO CHANGE —OVER FOLLOWING A PLAN

That is, while there is value in the items on the right, we value the items on the left more



Los 12 principios del Manifiesto Ágil

Principios

Satisfacer al Cliente con entregas frecuentes y tempranas

Cambios de Requerimientos son bienvenidos

Releases frecuentes (de 2 a 4 semanas)

Técnicos y no técnicos juntos

Individuos motivados

Medio comunicación: cara a cara

Métrica de progreso: software funcionando

Ritmo de desarrollo sostenible

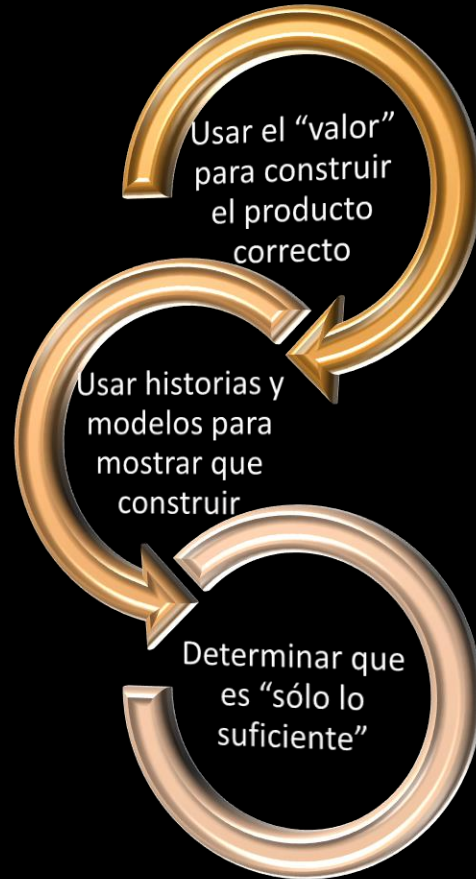
Atención continua a la excelencia técnica

Simplicidad: Maximización del trabajo no hecho

Arquitecturas, diseños y requerimientos emergentes

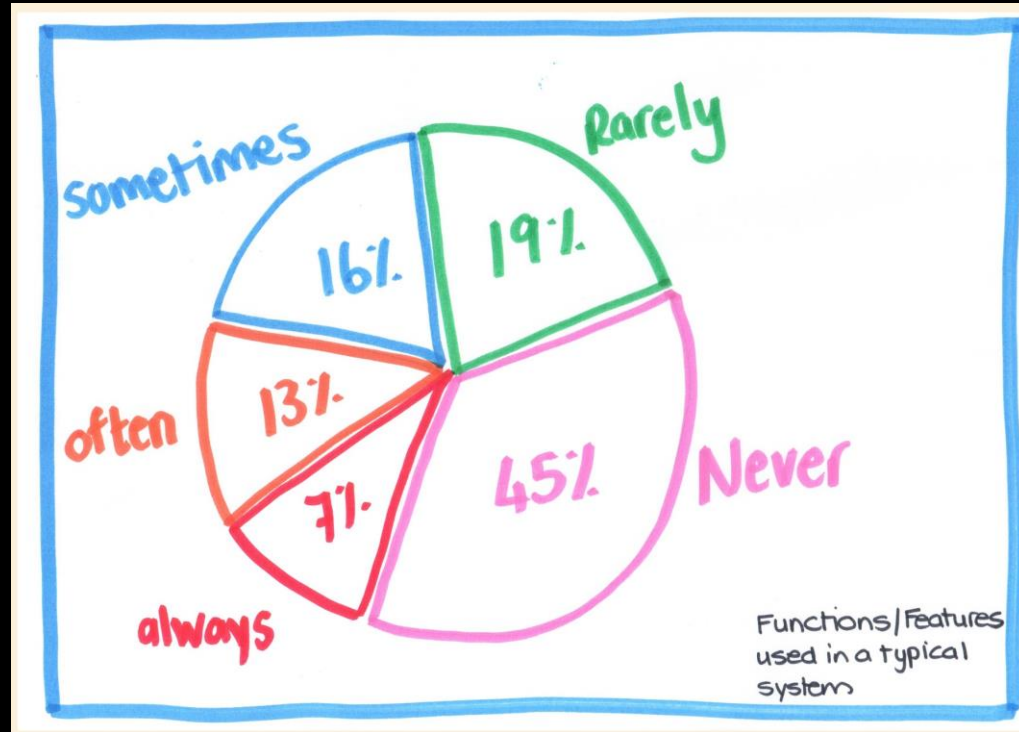
A intervalos regulares el equipo evalúa su desempeño

Requerimientos en Agile



El costo del tradicional BRUF

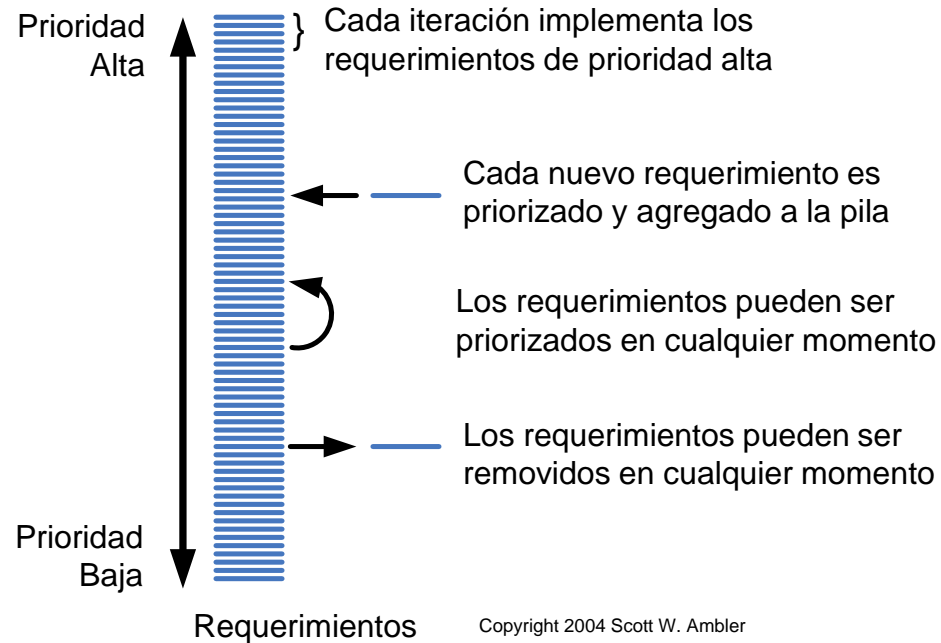
Los productos
“Exitosos”
también tienen
un desperdicio
significante



Fuente: Jim Johnson of the Standish Group, Keynote Speech XP 2002

Gestión Ágil de Requerimientos de Software

Los requisitos cambiantes son una ventaja competitiva
si puede actuar sobre ellos



Copyright 2004 Scott W. Ambler



Analice cuando lo
necesite, no antes

El cara-a-cara permite que fluya información vocal, subvocal, gestual con realimentación rápida.



*“Valor es la obtención de
beneficio **tangible** o
intangible”*

Masa Maeda, Serious LeAP

*“El valor lo asociamos a la
utilidad, beneficio o
satisfacción que le ofreces a
los usuarios finales por cada
funcionalidad completa que le
entregas”*

Pablo Lischinsky, Agile Trainer &
Consultant, Entrepreneur

Tradicional vs Ágil

	Tradicional	Ágil
Prioridad	Cumplir el plan.	Entregar Valor.
Enfoque	Ejecución ¿Cómo?	Estrategia (¿Porqué? ¿Para qué?).
Definición	Detallados y cerrados. Descubrimientos al inicio.	Esbozados y evolutivos – Descubrimiento progresivo.
Participación	Sponsor, stakeholder de mayor poder e interés. Relevo y le entrego al stakeholder para que lo valide	Colaborativo con stakeholders de mayor interés (clientes, usuarios finales).
Equipo	Analista de Negocios, Project Manager y Áreas de Proceso. Roles definidos	Equipo multidisciplinario. Trabajan todos juntos
Herramientas	Entrevistas, observación y formularios.	Principalmente prototipado. Técnicas de facilitación para descubrir. los req.
Documentación	Alto nivel de detalle – Matriz de Rastreabilidad para los Requerimientos	Historias de Usuario Mapeo de Historias (Story Mapping)
Productos	Definidos en alcance	Identificados progresivamente
Procesos	Estables, adversos al cambio	Incertidumbre, abierto al cambio

Tradicional vs. Ágil

el cambio no es en el release, es después del release.
es complicado introducir cambios durante.



Tipos de Requerimientos

meta de negocio que nuestro cliente espera lograr

Requerimiento de Negocio

Disminuir un X% de tiempo invertido en procesos manuales relacionados con atención al cliente.



Requerimiento de Usuario

Realizar consultas en línea del estado de cuenta de los clientes

Derivan de los del usuario

Qué debería hacer para resolver los req del usuario, qué debería hacer el SW



Requerimiento Funcional

Generar reporte de saldos de cuenta. Recibir notificaciones por mail.



Requerimiento No funcional

Formato del reporte PDF. Cumplir niveles de seguridad para credenciales de usuarios según la ley bancaria 9999XX



Uso de una determinada BD, servidores

Requerimiento de Implementación

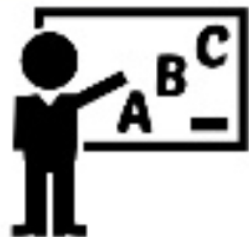
Servidores en la nube



EN RESUMEN...



ENTENDIENDO LA
NECESIDAD Y NEGOCIO...



DESCUBRIENDO LA SOLUCIÓN
DE FORMA COLABORATIVA...



JUNTO A UN EQUIPO
MOTIVADO Y COMPETENTE...



ENTREGAMOS
FRECUENTEMENTE VALOR A
LOS STAKEHOLDERS.

Por último



Los cambios son la única constante.



Stakeholders: no son todos los que están.



Siempre se cumple eso de que: “El usuario dice lo que quiere cuando recibe lo que pidió”.



No hay técnicas ni herramientas que sirvan para todos los casos.



Lo importante no es entregar una salida, un requerimiento, lo importante es entregar, un resultado, una solución de “valor”.

siempre el foco es la entrega de valor.

Principios relacionados a los Requerimientos Ágiles

Principios Ágiles

- 1- La prioridad es satisfacer al cliente a través de releases tempranos y frecuentes (2 semanas a un mes)
- 2 -Recibir cambios de requerimientos, aun en etapas finales
- 4 - Técnicos y no técnicos trabajando juntos TODO el proyecto
- 6 - El medio de comunicación por excelencia es cara a cara
- 11 - Las mejores arquitecturas, diseños y requerimientos emergen de equipos autoorganizados

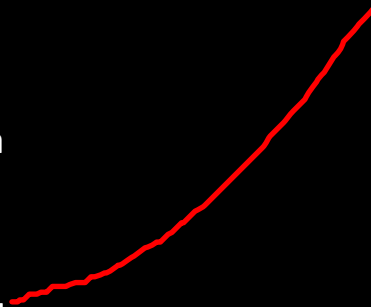
Principios Lean

- 1- Eliminar el desperdicio

4 - Diferir Compromisos

Pensar en lo q necesitamos EN EL MOMENTO que lo necesitamos, y no ANTES.

6 - Ver el todo



Universidad Tecnológica Nacional

Facultad Regional Córdoba

Cátedra de Ingeniería de Software

Docentes: Judith Meles – Laura Covaro

User Stories

tema de parcial práctico

“....se las llama “stories” porque se supone que Ud. cuenta una historia. Lo que se escribe en la tarjeta no es importante, lo que Ud. habla, si!.

--- Jeff Patton, InfoQ,

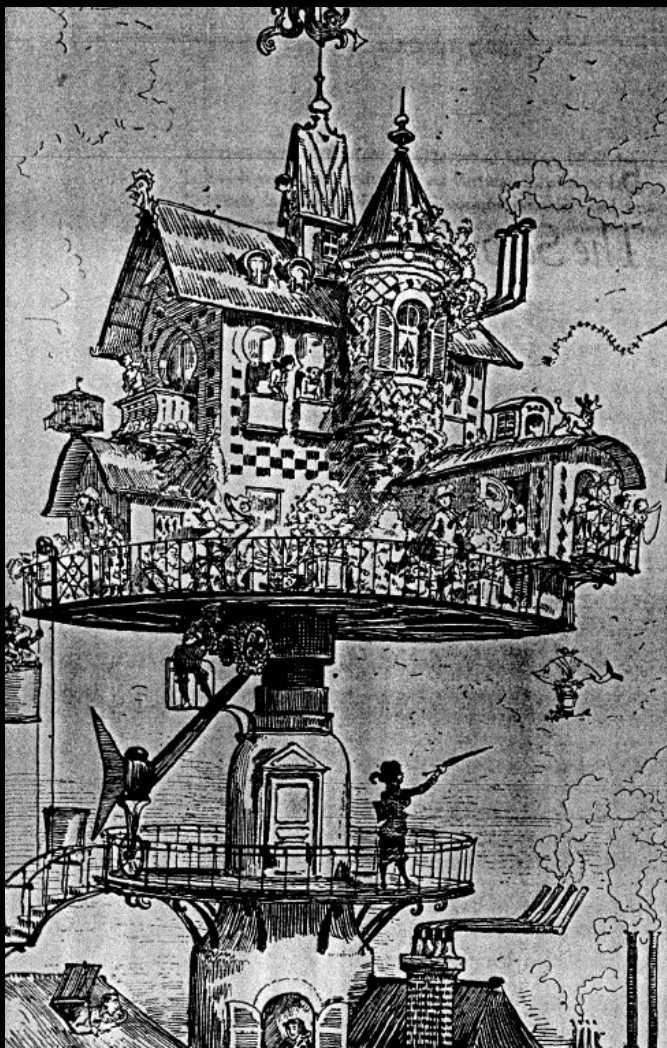
create conversation.



@gapingvoid

Desarrollo ágil de Software (Agile)

Un compromiso útil entre nada de proceso y demasiado proceso (Fowler, 2001)



La parte más difícil de construir un sistema de software es decidir precisamente qué construir. Ninguna otra parte del trabajo conceptual es tan difícil como establecer los requerimientos técnicos detallados... Ninguna otra parte del trabajo afecta tanto el sistema resultante si se hace incorrectamente. Ninguna otra parte es tan difícil de rectificar más adelante”

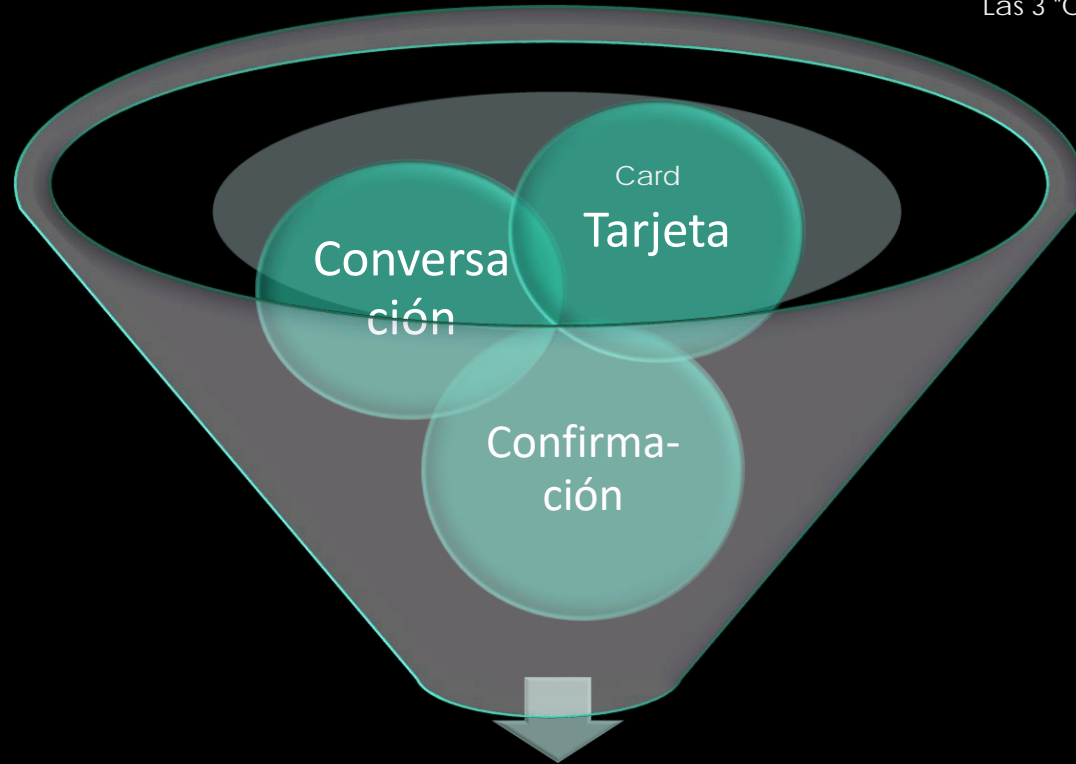
Fred Brooks - “No Silver Bullet - Essence and Accidents of Software Engineering”. IEEE Computer, Abril de 1987.

El cara-a-cara permite que fluya información vocal, subvocal, gestual con realimentación rápida.



¿Cuáles son las partes de una User Story?

Las 3 "C"



User Story

¿Qué es una User Story?

Front of Card

173

As a student I want to purchase
a parking pass so that I can
drive to school

Priority: ~~High~~ Should
Estimate: 4

Back of Card

Confirmations:

- ~~The student must pay the correct amount~~
- One pass for one month is issued at a time
- The student will not receive a pass if the payment isn't sufficient
- The person buying the pass must be a currently enrolled student.
- The student may only buy one pass per month.

Forma de ^{enunciar}expresar las Historias de Usuario

Se hace con un mecanismo:

YO Como **<nombre del rol>**,
yo puedo **<actividad>**
de forma tal que **<valor
de negocio que
recibo>**

Comunica porque es
necesaria la actividad

Representa quién está
realizando la acción o
quién recibe el valor
de la actividad.

Representa la
acción que realizará
el sistema

As who, I want
what so that why.

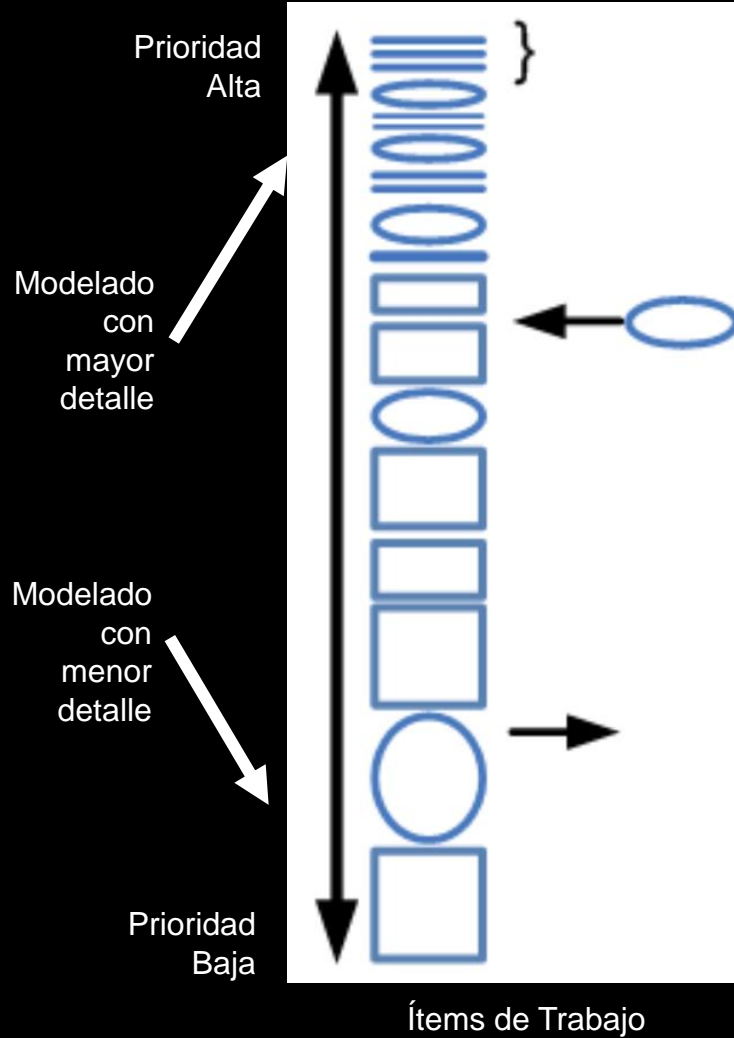
Las User Stories son Multipropósito

- Las historias son:
 - Una necesidad del usuario
 - Una descripción del producto
 - Un ítem de planificación
 - Token para una conversación
 - Mecanismo para diferir una conversación

** Kent Beck coined the term user stories in Extreme Programming Explained 1st Edition, 1999*



El Product Owner Prioriza las historias en el Product Backlog



Cada iteración implementa los ítems de trabajo de mayor prioridad

Cada nuevo requerimiento es priorizado y agregado a la pila

Los ítems de trabajos pueden ser re-priorizados en cualquier momento

Los ítems de trabajo pueden ser removidos en cualquier momento

User stories: Porciones Verticales del SW

Incluye todo lo que implica, todos los lugares donde esa story impacta.

Las necesidades se modelan como User story, pero no es la única estrategia.



Donde impacta
"fuertemente"
esto??

Story 1	Story 2	
GUI		
Business Logic		
Database		

Modelado de Roles

Forma de identificar los roles y modelarlos para poder identificar quienes son y como se comportarian, e imaginar como harian uso del sistema.



Modelado de Roles: Tarjeta de Rol de Usuario

Rol de Usuario: Reclutador Interno

Nos es un experto en computadoras, pero bastante adepto a utilizar la Web. Utilizará el software con poca frecuencia pero muy intensamente. Leerá anuncios de otras compañías para averiguar cuál es la mejor palabra para sus anuncios. La facilidad de uso es importante, pero más importante es que aprenda, lo pueda recordar meses después.

SEARCH AND REPLACE

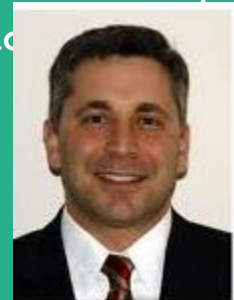
A user realizes he mis-capitalized a word everywhere in his document, so he tells the word processor to search for all occurrences of it and replace them with the corrected word.

Modelado de Roles: Técnicas Adicionales

- Personas

ponerle un nombre, y pensarlo como una persona concreta.

Mario trabaja como reclutador en el departamento de Speedy Networks, una fábrica de componentes de red de alta gama. El ha trabajado para Speedy Networks por 6 años. Mario tiene un arreglo de horario flexible y trabaja desde casa cada viernes. Mario es muy fuerte con las computadoras y se considera a sí mismo un usuario avanzado de los productos que usa. La esposa de Mario, Kim, está terminando su Doctorado en Química en la Universidad de Stanford. Dado que Speedy Networks ha estado creciendo consistentemente, Mario siempre está buscando ingenieros.



Modelado de Roles: Técnicas Adicionales

Encontrar roles extremos

Diseño de un PDA para:

- El Papa
- Una mujer de 20 años con muchos novios
- Un traficante de drogas

Tanto la mujer como el traficante desearán mantener agendas separadas en caso de que la vea la policía o un novio. El Papa probablemente tenga menos necesidad de discreción pero querrá un tamaño de fuente más grande.

Usuarios Representantes (Proxies)

- Tipos de usuarios representantes:
 - Gerentes de Usuarios
 - Gerentes de Desarrollo
 - Alguien del grupo de marketing
 - Vendedores
 - Expertos del Dominio
 - Clientes
 - Capacitadores y personal de soporte.

No usar nombres tan genéricos si se puede especificar

**No son ideales como
los usuarios verdaderos
...EVITELOS !!!**

User Story: un ejemplo de tarjeta

Buscar Destino por Dirección

rol
Como Conductor acción en el sistema **quiero buscar un destino a partir de una calle y altura para**
poder llegar al lugar deseado sin perderme.
valor en el negocio

Criterios de Aceptación: condiciones clave para ponerse de acuerdo cuando el user usa el sistema

Cuando presente la historia funcionando al PO, se deben cumplir los criterios de aceptación. Se deben cumplir si o si.

- **La altura de la calle es un número.**
- **La búsqueda no puede demorar más de 30 segundos.**

Criterios de Aceptación de User Stories



Criterios de Aceptación de User Stories

- Son condiciones de satisfacción.

- Para esta historia de usuario:

Como un cliente, quiero poder ver mi consumo de energía diario, así puedo bajar mis costos y usos de energía.

- Los criterios de aceptación serían:

- Leer los metros DecaWatt cada 10 segundos y mostrarlos en el portal con incrementos de 15 minutos y mostrarlos en forma local en cada lectura.
- No tendencias multi-días por ahora (otra historia)

Debo asegurarme que se cumplan.



Criterios de Aceptación de Historias de Usuario

- Definen límites para una user story (US)
- Ayudan a que los PO respondan lo que necesitan para que la US provea valor (requerimientos funcionales mínimos)
- Ayudan a que el equipo tenga una visión compartida de la US
- Ayudan a desarrolladores y testers a derivar las pruebas.
- Ayudan a los desarrolladores a saber cuando parar de agregar funcionalidad en una US



¿Cuáles son los Criterios de Aceptación buenos?

No dicen como resolver la user, sólo definen una intención. No es la solución.

- Definen una intención, no una solución
 - Ej.: El usuario debe elegir al menos una cuenta para operar
- Son independientes de la implementación
- Relativamente de alto nivel, no es necesario que se escriba cada detalle



¿Y los detalles?

¿Dónde van? no van dentro de la user



- Detalles como:
 - El encabezado de la columna se nombra “Saldo”
 - El formato del saldo es 999.999.999,99
 - Debería usarse una lista desplegable en lugar de un Check box.
- Estos detalles que son el resultado de las conversaciones con el PO y el equipo puede capturarlos en dos lugares:
 - Documentación interna de los equipos
 - Pruebas de aceptación automatizadas

Pruebas de Aceptación de User Stories

Front of Card

1B

As a student I want to purchase
a parking pass so that I can
drive to school

Priority: ~~High~~ Should
Estimate: 4

Back of Card

Confirmations:

- ~~The student must pay the correct amount~~
- One pass for one month is issued at a time
- The student will not receive a pass if the payment isn't sufficient
- The person buying the pass must be a currently enrolled student.
- The student may only buy one pass per month.

Parte de atrás de la user

Pruebas de Aceptación de Historias de Usuario

Expresan detalles resultantes de la conversación

Complementan la User Story

Proceso de dos pasos:

1. Identificarlas al dorso de la US.
2. Diseñar las pruebas completas



Ejemplo: User Stories / Casos de Prueba

materializar en casos concretos
como estos se estan cumpliendo

Como compañía quiero pagar por una búsqueda de puestos con una tarjeta de crédito, así resuelvo mi necesidad en forma más eficiente.

Criterio de Aceptación:

- Se acepta Visa, MasterCard y American Express
- En compras mayores de \$100 se piden el número del dorso de la tarjeta

Probar con Visa (pasa)

Probar con MasterCard (pasa)

Probar con American Express (pasa)

Probar con Dinner's Club (falla)

Probar con números de tarjeta buenos

Probar con números de tarjeta malos

Probar con números de tarjeta faltantes

Probar con tarjetas vencidas

Probar con montos menores de \$100

Probar con montos mayores de \$100

Definición de Hecho – Definition of Done



Definición de listo – Definition of Ready

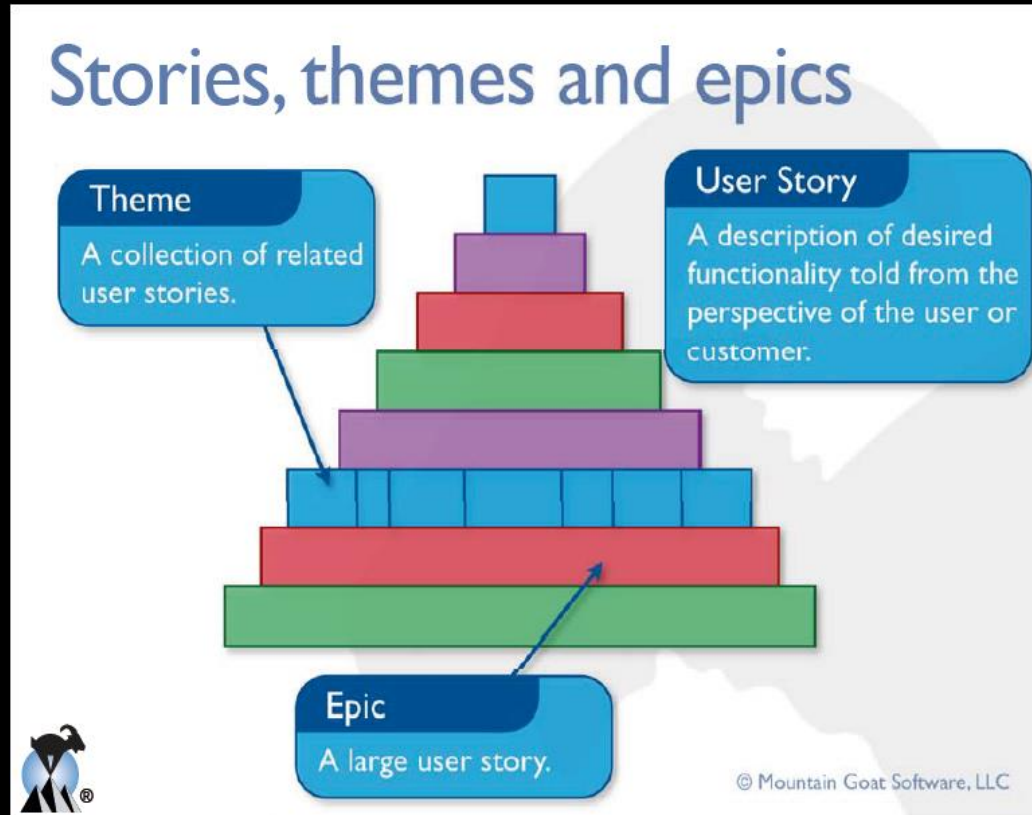
definir cuando una historia está lista en terminos de detalle para poder ser incluida en un sprint o iteracion para poder empezar a utilizarla



Algo más sobre las User Stories...

- No son especificaciones detalladas de requerimientos (como los casos de uso)
- Son expresiones de intención, “es necesario que haga algo como esto...”
- No están detallados al principio del proyecto, elaborados evitando especificaciones anticipadas, demoras en el desarrollo, inventario de requerimientos y una definición limitada de la solución.
- Necesita poco o nulo mantenimiento y puede descartarse después de la implementación.
- Junto con el código, sirven de entrada a la documentación que se desarrolla incrementalmente después.

Diferentes niveles de abstracción



Epicas: en algun momento se detalla y se convierte en muchas user stories. No cumple con la definicion de ready

*INVEST Model

Para llegar al criterio de Ready

tiene cumplir una user story para considerarla lista

- **Independent** – calendarizables e implementables en cualquier orden
no deben depender entre si las stories
- **Negotiable** – el “qué” ^{hacer} no el “cómo” ^{hacerla}
- **Valuable** – debe tener valor para el cliente
- **Estimatable** – para ayudar al cliente a armar un ranking basado en costos
tengo q poder decir cuando esfuerzo me llevara, para luego definir el costo.
- **Small** – deben ser “consumidas” en una iteración pequeñas, no mas de 4 sem en hacerla
- **Testable** – demostrar que fueron implementadas casos de prueba

Que no son las user storys.....



User Stories:
Algo “Huele
mal” ...

¿Qué huele mal
en las siguientes
historias?



¿Qué huele mal en las siguientes historias?

- El usuario puede ejecutar el sistema en Windows XP y Linux.
- Todos los gráficos se harán con una librería de terceros.
- El usuario podrá deshacer hasta 50 comandos.
- El software deberá entregarse el 30 de Julio.
- El software deberá estar desarrollado en JAVA

Spikes

es una tarea de investigación

son importantes porque llevan tiempo

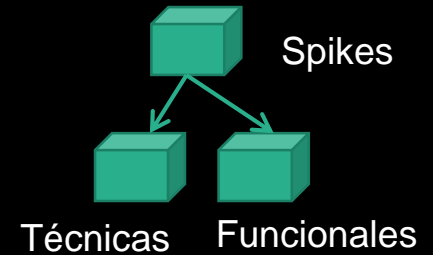
- Tipo especial de historia, utilizado para quitar riesgo e incertidumbre de una User Story u otra faceta del proyecto.
- Se clasifican en : técnicas y funcionales.
- Pueden utilizarse para:
 - Inversión básica para familiarizar al equipo con una nueva tecnología o dominio.
 - Analizar un comportamiento de una historia compleja y poder así dividirla en piezas manejables.
 - Ganar confianza frente a riesgos tecnológicos, investigando o prototipando para ganar confianza.
 - Frente a riesgos funcionales, donde no está claro como el sistema debe resolverla interacción con el usuario para alcanzar el beneficio esperado.

Spikes

Investigación de tecnología

Técnicas uso de tecnologías nuevas

- Utilizadas para investigar enfoques técnicos en el dominio de la solución.
 - Evaluar performance potencial
 - Decisión hacer o comprar
 - Evaluar la implementación de cierta tecnología.
- Cualquier situación en la que el equipo necesite una comprensión más fiable antes de comprometerse a una nueva funcionalidad en un tiempo fijo.

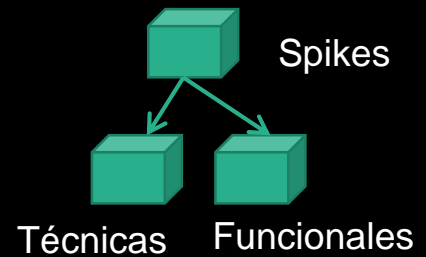


Funcionales

- Utilizadas cuando hay cierta incertidumbre respecto de cómo el usuario interactuará con el sistema.
- Usualmente son mejor evaluadas con prototipos para obtener realimentación de los usuarios o involucrados.

Cómo el usuario se relaciona con el sistema

Spikes



- Algunas User Stories requieren de ambos tipos de spikes. Por ejemplo:
 - Como un cliente, quiero ver mi uso diario de energía en un histograma, para poder comprender rápidamente mi consumo de energía pasado, presente y proyectado.
- En este caso un equipo puede crear dos spikes:
 - Spike Técnico:
 - Investigar cuanto tiempo requiere actualizar un display de un cliente al uso actual, determinando requerimientos de comunicación, ancho de banda y si los datos se actualizan en formato push o pull.
 - Spike Funcional:
 - Crear un prototipo de histograma en el portal web y obtener la retroalimentación de algunos usuarios respecto del tamaño, el estilo de la presentación y los atributos gráficos.

Lineamientos para Spikes

- Estimables, demostrables, y aceptables
- La excepción, no la regla
 - Toda historia tiene incertidumbre y riesgos.
 - El objetivo del equipo es aprender a aceptar y resolver cierta incertidumbre en cada iteración.
 - Los spikes deben dejarse para incógnitas mas críticas y grandes.
 - Utilizar spikes como última opción.
- Implementar la spike en una iteración separada de las historias resultantes
 - Salvo que el spike sea pequeño y sencillo y sea probable encontrar una solución rápida en cuyo caso, spike e historia pueden incluirse en la misma iteración.

que la spike este separada de la historia que resuelve

Algunas cosas para dejar en claro

las user están redactadas como una necesidad del usuario, no son requerimientos en términos de qué debería hacer el sistema

- Diferir el análisis detallado tan tarde como sea posible, lo que es justo antes de que el trabajo comience.
- Hasta entonces, se capturan requerimientos en la forma de “user story”, que son descripciones breves de funcionalidad relevante para el cliente.
- Las user story no son requerimientos; son marcadores para conversaciones más detalladas y análisis que deberán ocurrir conforme esas historias vayan implementándose.
- Por lo tanto, no necesitan ser descripciones exhaustivas de la funcionalidad del sistema, sólo la suficiente información para que los desarrolladores y los clientes tengan una comprensión común.

disminuye la incertidumbre

no es una descripción exhaustiva de lo que el sistema debe hacer

Material Bibliográfico de Referencia

- Libro:

- Cohn, Mike - USER STORIES APPLIED – Editorial Addison Wesley 2004-
Capítulos 1, 2 y 6 aca esta todo

- Papers

- Dean Leffingwell and Pete Behrens – A user story primer (2009) buscar version en español

- Link

- <http://www.mountaingoatsoftware.com/> extra, no hace falta

Estimaciones Ágiles

¿Cuanta gente hay en el aula de al lado? La respuesta es "me levanto y me fijo si puedo"

- En los equipos Agile, las features/stories son estimadas usando una medida de tamaño relativo conocida como story points (SP)

Se usan las estimaciones cuando no tengo la info, hacemos una adivinación del valor.

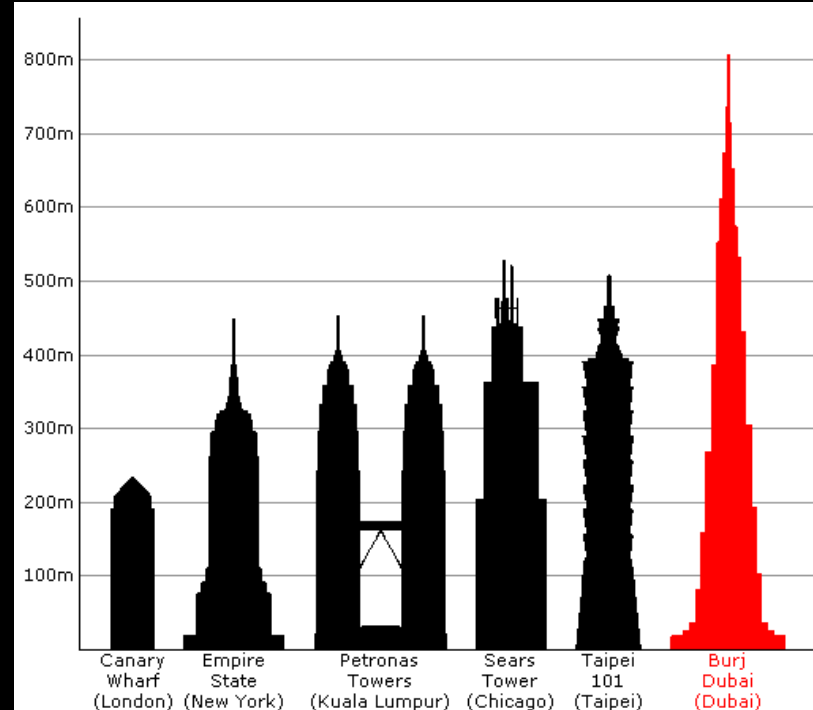
- Las medidas relativas no son absolutas.
- Story Points no es una medida basada en tiempo

Las estimaciones se basan en otorgar un valor con info incompleta.

cono de incertidumbre: las estimaciones en etapas tempranas, con tan poca info para estimar, el riesgo es mucho mas alto. Al ir avanzando el tiempo, se va disminuyendo, mientras tenemos mas info. el error luego queda absorbido por el equipo.

Estimación Relativa

- Las personas no saben estimar en términos absolutos
- Somos buenos comparando cosas
- Comparar es generalmente más rápido.
- Se obtiene una mejor dinámica grupal y pensamiento de equipo más que individual
- Se emplea mejor el tiempo de análisis de las storys



y...pero, el tamaño?

- “La palabra tamaño refiere a cuan grande o pequeño es algo”
- El tamaño es una medida de la cantidad de trabajo necesaria para producir una feature/story.
- El tamaño indica:
 - Cuán compleja es una feature/story
 - Cuánto trabajo es requerido para hacer o completar una feature/story
 - Y cuán grande es una feature/story



Por favor, “dé tamaño” de estos perros



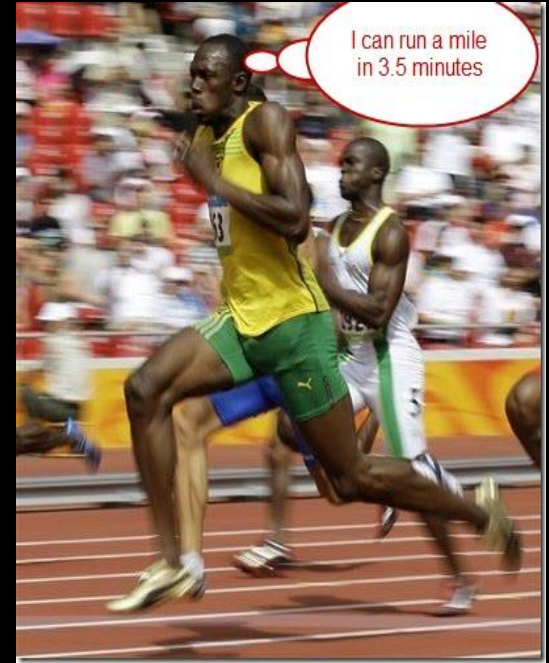
Tamaño Vs. Esfuerzo

lo primero que vamos a estimar es el tamaño de algo.
despues el esfuerzo: en un medida que es la cantidad de hombres lineales
que le llevaira a una persona particular poder contruir ese tamaño
estimado. depende del equipo.



Las estimaciones basadas
en tiempo son más
propensas a errores debido
a varias razones.

- Habilidades
- Conocimiento
- Asunciones
- Experiencia
- Familiaridad con los dominios de aplicación/negocio



Tamaño NO ES esfuerzo

Y qué hacemos con el tamaño!?????

User Stories: podemos asignarles números, un cierto orden

- Tamaño por números: 1 a 10
- Talles de remeras: S, M, L, XL, XXL
- Serie 2ⁿ : 1, 2, 4, 8, 16, 32, 64, etc.
- Fibonacci: 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, etc.

lo importante es q el equipo tenga consenso al fijar estas escalas de medidas y lo mantenga a lo largo de todo el proyecto, porque sino se perderia el punto de referencia y tendríamos un conjunto de US no comparables entre si.

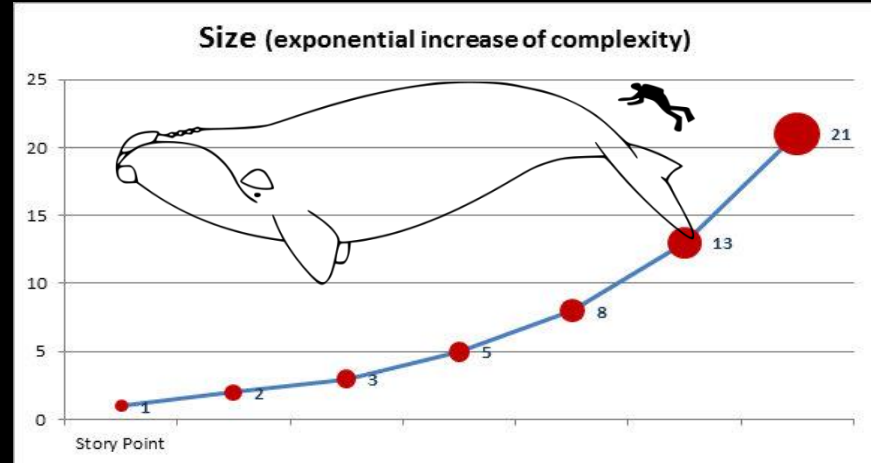
Una vez elegida la escala no se cambia! Si se cambia cambiamos el metro patrón



Y qué es un Story Point? (una de muchas definiciones)

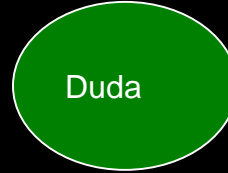
complejidad, incertidumbre o riesgo y esfuerzo

- Es una unidad de medida específica (del equipo) de, complejidad, riesgo y esfuerzo, es lo que “el kilo” a la unidad de nuestro sistema de medición de peso
- Story point da idea del “peso” de cada story y decide cuan grande (compleja) es
- La complejidad de una
 - feature/story tiende a
 - incrementarse exponencialmente.



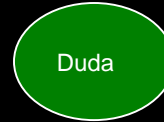
3 Stories que queremos estimar

Story 1



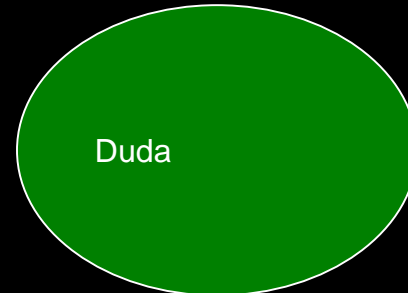
cada una de ellas pueden tener diferentes combinaciones de cant de riesgo, cant de complejidad y cant de esfuerzo.

Story 2



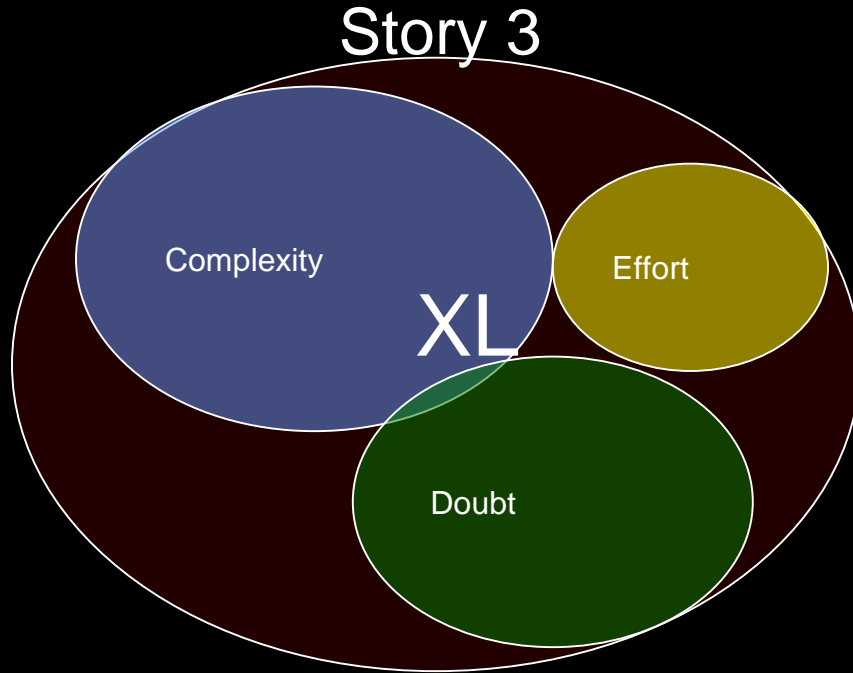
la complejidad se relaciona con la cantidad de grados de libertad, variables en juego a compatibilizar.

Story 3

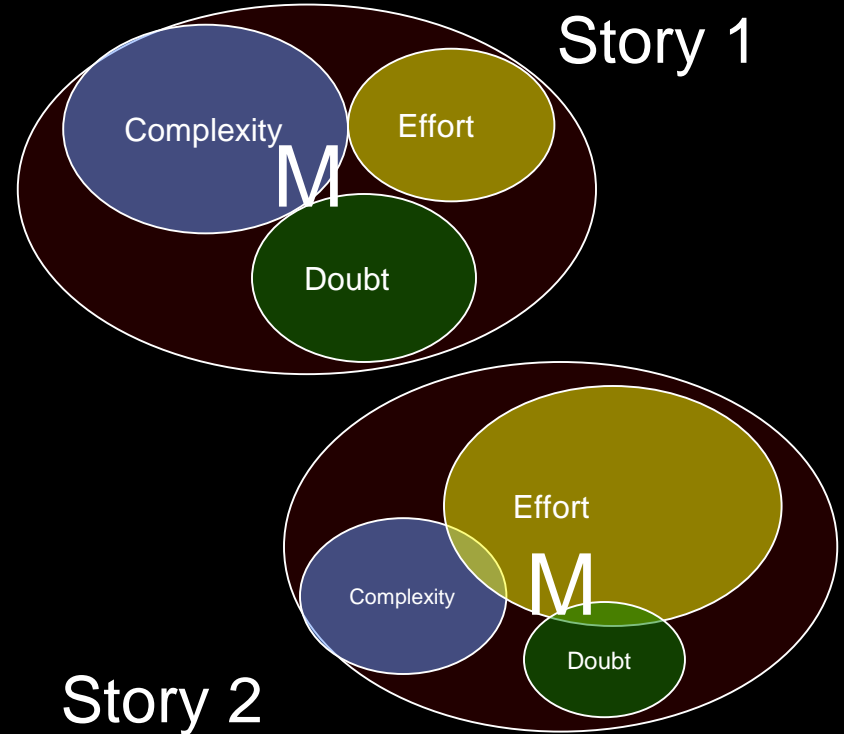


la duda esta relacionada con la incetridumbre que tiene el equipo acerca de la informacion q posee.

“tamaño” de las Stories



asi le vamos asignando valores a las US



Y si usamos números?

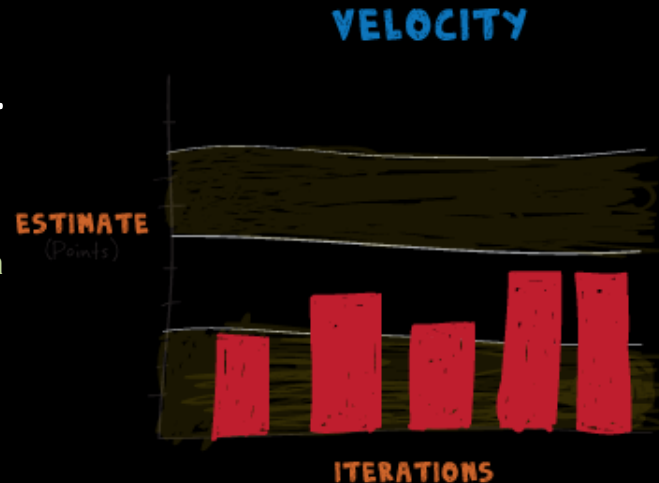


Velocidad (Velocity)

cuando termina la iteracion, se ven las US completas, y se suman los story points.

- **Velocidad/ Velocity** es una medida (métrica) del progreso de un equipo. Se calcula sumando el número de story points (asignados a cada user story) que el equipo **completa** durante la **iteración**.
- Se cuentan los **story points** de las **Users Storys** que están **completas**, no parcialmente completas.
- La Velocidad corrige los errores de estimación.

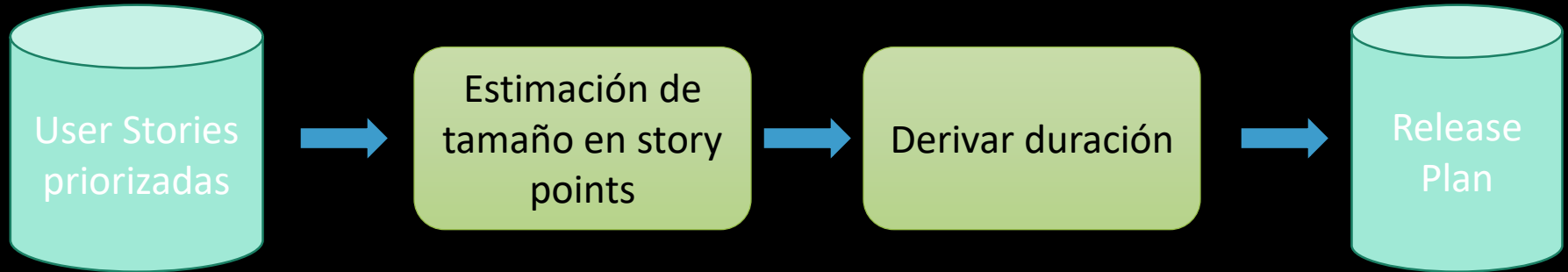
"DONE": no es solo que haya estado implementada en la iteracion, sino q ademas de eso el PO debe estar de acuerdo y AHI esta completa y AHI podemos calcular la VELOCIDAD



La velocidad nos sirve para saber la duración de un proyecto. Tengo las estimaciones en story points, tomo la velocidad, y puedo ver de lo que tengo pendiente, a esa velocidad, cuantas iteraciones mas me faltan para que este listo.

¿Y cómo hago con un proyecto?

- Si estimo User Storys cómo hago para estimar un proyecto?
 - La duración de un proyecto no se “estima”, se deriva.... tomando el número total de story points de sus user storys y dividiéndolo por la velocidad del equipo.



- La velocidad nos ayuda a determinar un horizonte de planificación apropiando
- La estimación en story points separa completamente la estimación de esfuerzo de la estimación de la duración.

Una Propuesta de método de estimación



Poker estimation metodo de estimacion

desagregacion: separar en componentes una funcionalidad para compararlos

- Popular entre los Agile practitioners, publicado por Mike Cohn
- Combina opinión de experto, analogía y desagregación.
- Participantes en “planning poker”
son desarrolladores
 - “Las personas más competentes en resolver una tarea deben ser quienes las estiman”
son los que enfrentan la funcionalidad




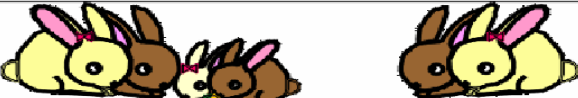




Fibonacci (se acuerdan?) escala que vamos a usar

- La secuencia empieza en 1 y cada numero subsecuente es la suma de los dos precedentes. (1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144....)

crece exponencialmente

se adapta bien al
crecimiento de la
funcionalidad del SW

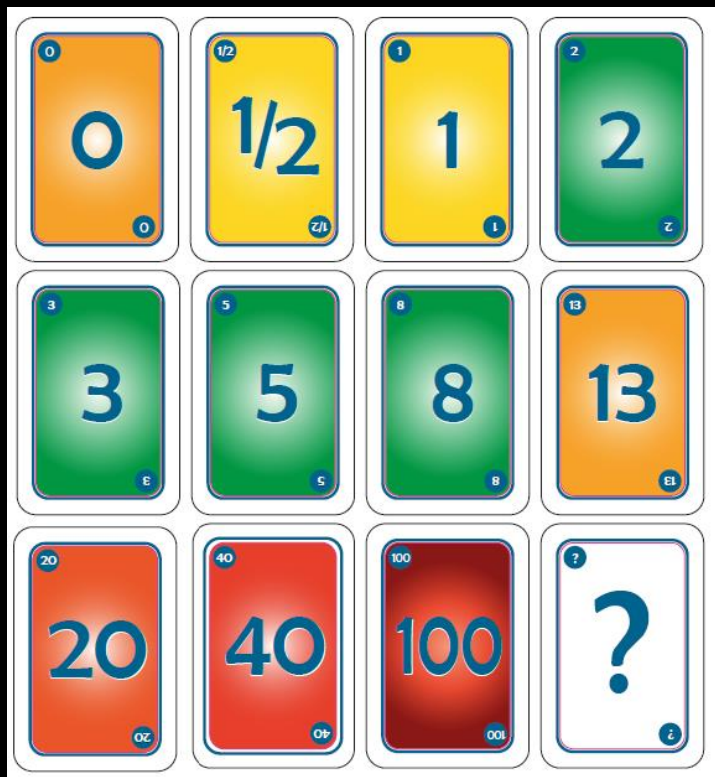
		Pares de conejos 1
1 mes		1
2° mes		2
3° mes		3
4° mes		5
5° mes		8

Poker Planning



<http://www.planningpoker.com/detail.html>

¿Cómo “decodificar” las estimaciones?



el objetivo es que se encuentren entre 0 y 8.
13 ya se divine.

- 0: Quizás ud. no tenga idea de su producto o funcionalidad en este punto.
- 1/2, 1: funcionalidad pequeña (usualmente cosmética).
- 2-3: funcionalidad pequeña a mediana. Es lo que queremos. 😊
- 5: Funcionalidad media. Es lo que queremos 😊
- 8: Funcionalidad grande, de todas formas lo podemos hacer, pero hay que preguntarse sino se puede partir o dividir en algo más pequeño. No es lo mejor, pero todavía 😊
- 13: Alguien puede explicar por que no lo podemos dividir?
- 20:Cuál es la razón de negocio que justifica semejante story y más fuerte aún, por qué no se puede dividir?.
- 40: no hay forma de hacer esto en un sprint.
- 100: confirmación de que está algo muy mal. Mejor ni arrancar. 20,40,100 son valores altísimos. si hay algo así es mucho riesgo asociado, y hay q subdividir esa US.

Agile es empírico,
Inspeccionar y adaptar
es mandatorio!

siempre hay bucles de retroalimentación

