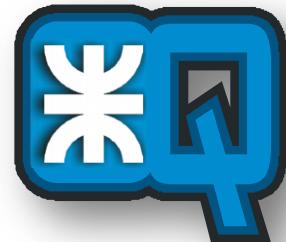


Sprint 1: Básicos

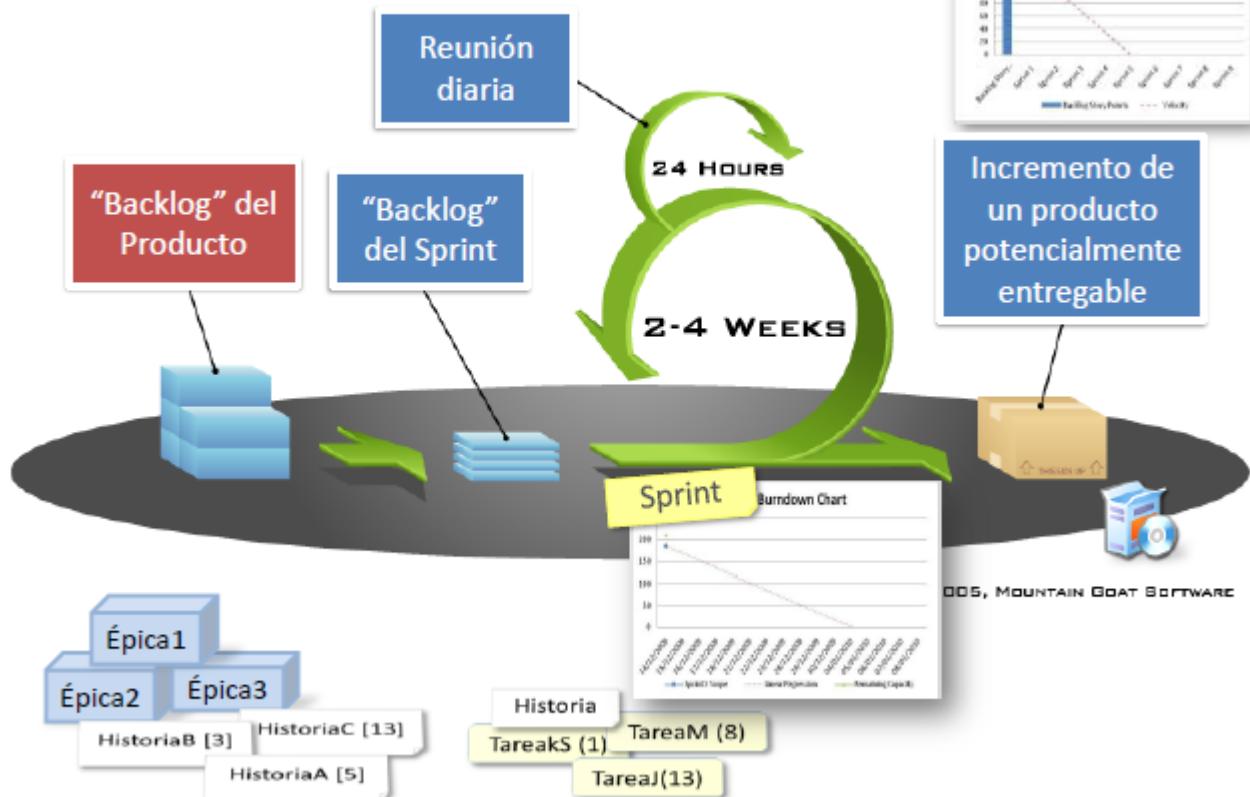
- Mgtr. Diego Rubio
- Mgtr. Natalia Andriano
- Ing. Juan Pablo Bruno
- Ing. Mauricio Silclir



Objetivos específicos

- Desarrollar en el estudiante conceptos acerca cuales son los atributos a tener en cuenta para una correcta **planeación** dentro de las metodologías ágiles
- Desarrollar en el estudiante conceptos básicos acerca de una de las diferentes técnicas existentes para la **administración de requerimientos** siguiendo una metodología ágil. Historias de usuarios.
- Brindar al estudiante conocimiento específico aplicado en relación a las **métricas** más comúnmente utilizadas en desarrollos con metodologías ágiles
- Introducir conceptos básicos sobre **testing ágil** y técnicas a utilizar. Test exploratorio.

Scrum de un vistazo ...



¿Todos los conceptos están claros?



Épicas e historias de usuarios



Un producto puede incluir varias features

Una Épica es una historia de usuario de muy alto nivel que representa una funcionalidad específica

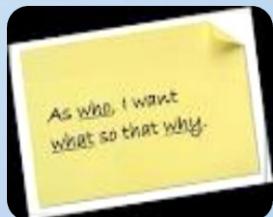
Una historia de usuario está compuesta por varias tareas

Ejemplo de Épica

“Como un usuario, quiero ser capaz de gestionar clientes”



Historias de usuarios (*User stories*)



Definen lo que hay que construir dentro de un proyecto de software.



Son priorizadas por el cliente

- Son descompuestas en tareas y estimadas por los desarrolladores

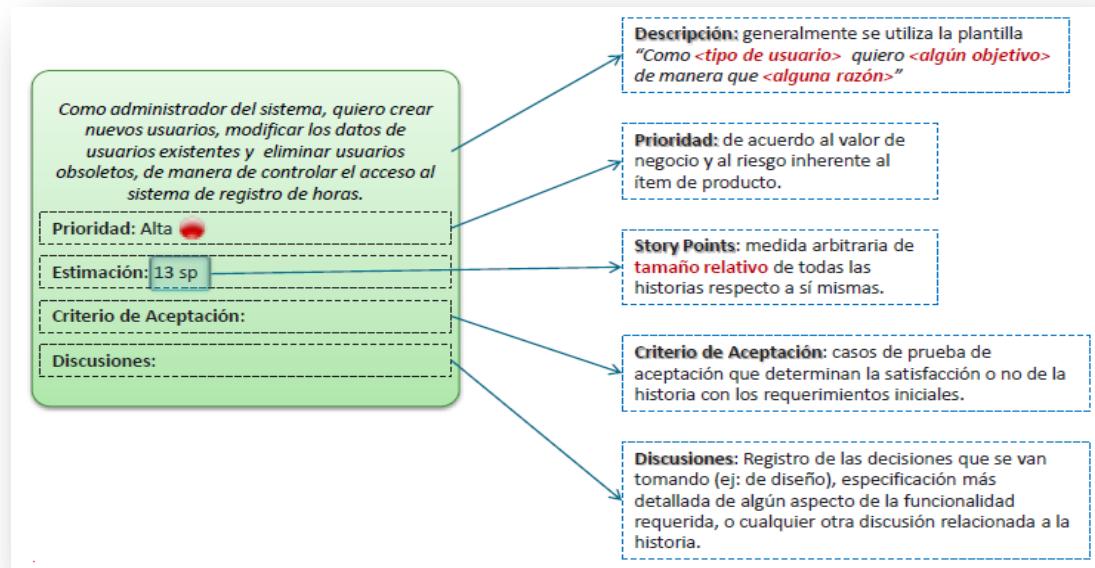


Tienen que tener asociadas al menos un test case de aceptación

- Permiten al desarrollador testear la funcionalidad
- Permiten al cliente validar que la funcionalidad esté implementada correctamente.

Historias de usuarios (*User stories*)

- Es una **descripción simple y corta de una funcionalidad del sistema vista desde la perspectiva de la persona que desea dicha nueva funcionalidad**, generalmente un usuario o cliente del sistema.
- Las historias favorecen el enfoque de **discutir las nuevas funcionalidades más que escribir acerca de ellas**.



Historias de usuarios (*User stories*)



Ejemplo de historia de usuarios

- “Como un **usuario**, quiero **buscar los clientes por nombre de provincia para poder tener listados según ubicación geográfica**”
 - “**Rol**” → *Usuario*
 - **Funcionalidad**” → **buscar los clientes por nombre de provincia**
 - “**Razón**” → **poder tener listados según ubicación geográfica**



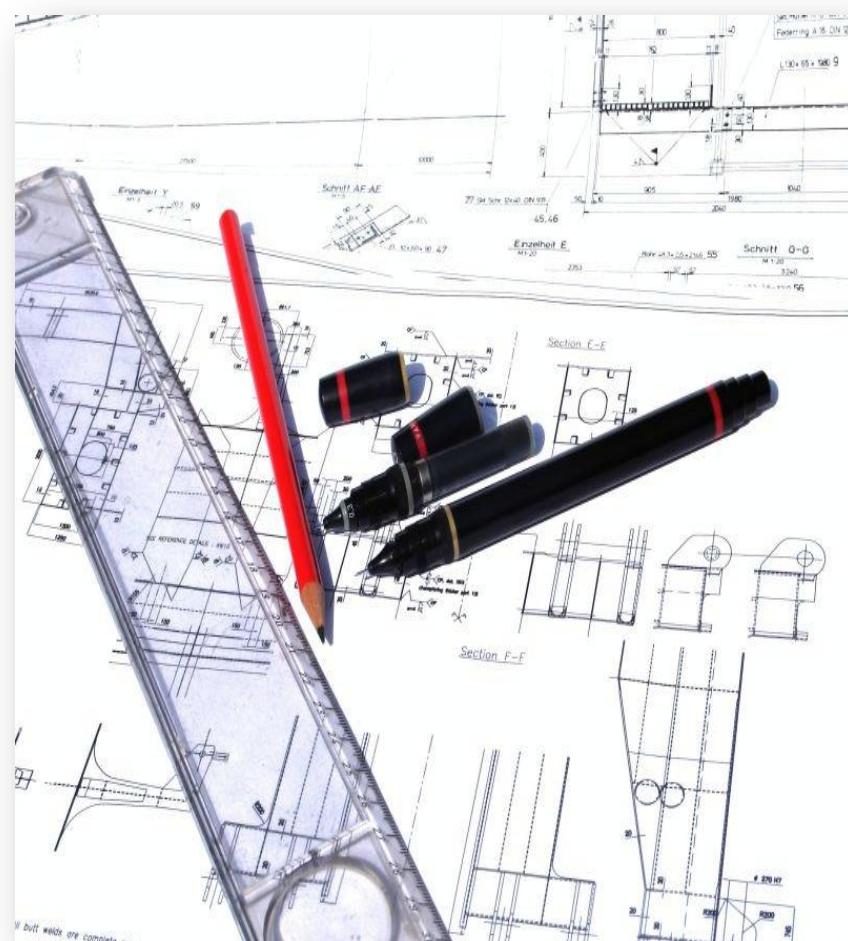
Planning is
everything.

Plans are
nothing

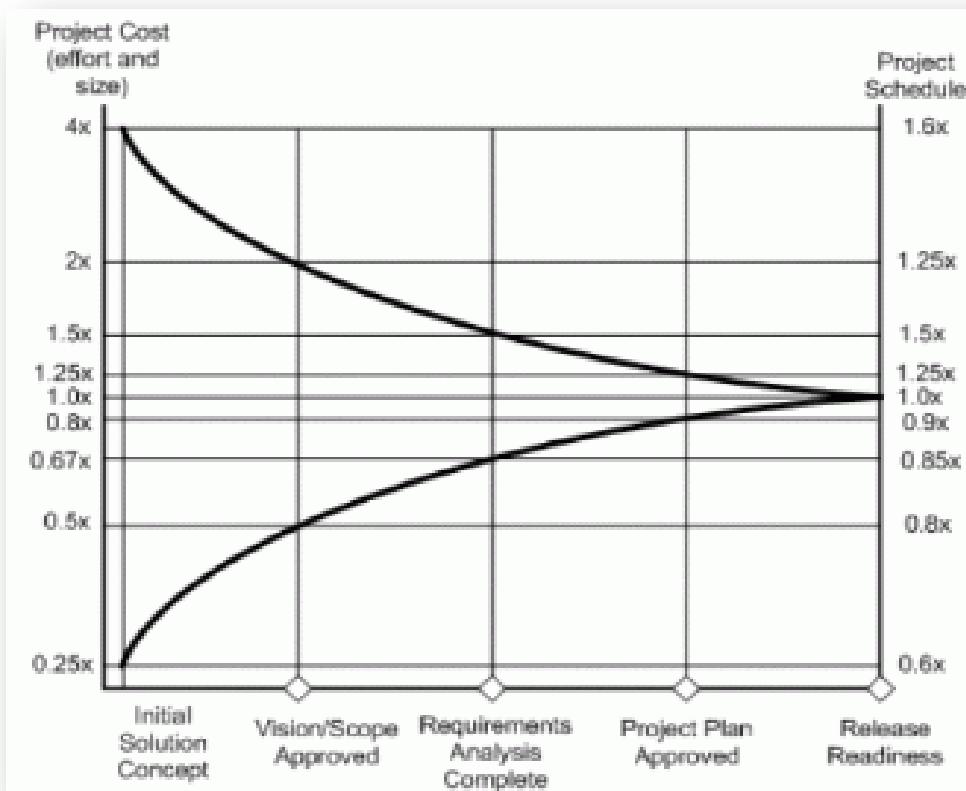
Field Marshal Helmuth
Graf von Moltke

Planificación y estimación...

- Son críticos para el éxito del proyecto
- Los planes guían nuestras decisiones de inversión...
- Sin los planes abrimos a nuestros proyectos a un sin número de problemas..



Cono de incertidumbre...



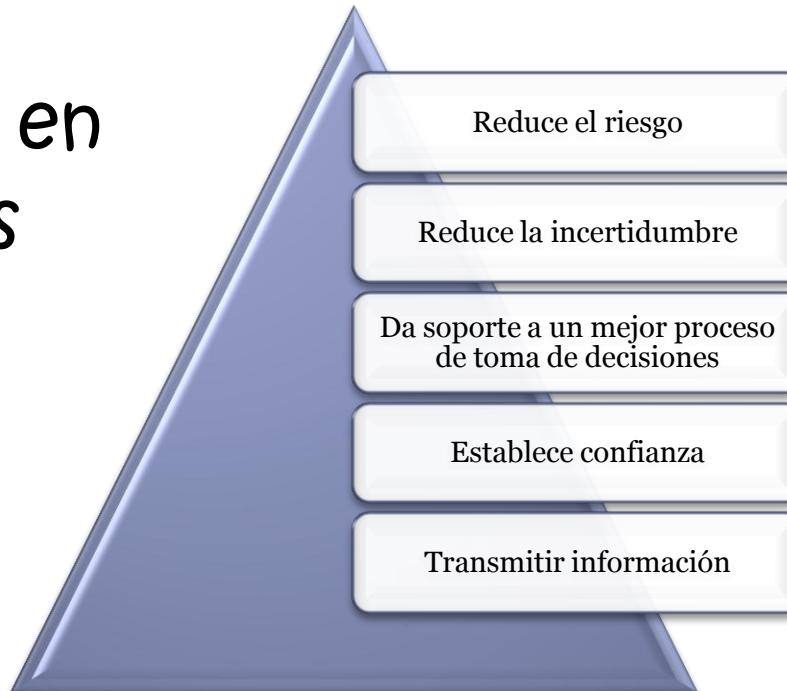
Steve McConnell (1998) later called the “cone of uncertainty.”

Pero entonces...qué es planear y estimar???

- O mejor dicho que NO es planear y estimar???

Planear y estimar

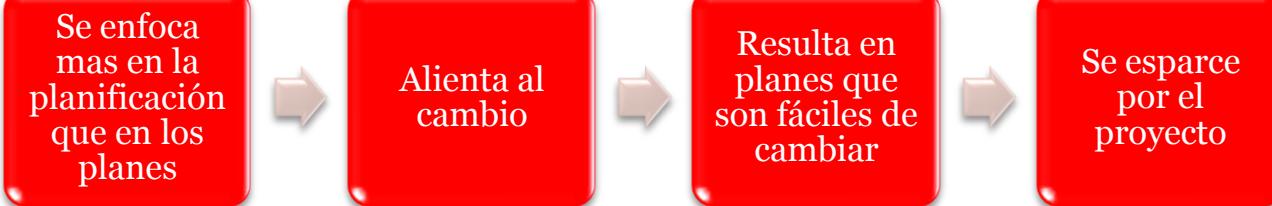
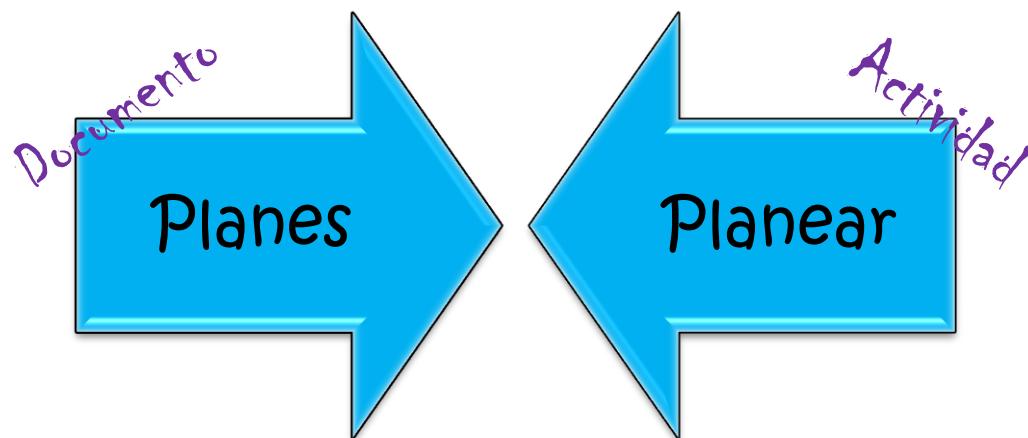
- NO se trata sólo de determinar una fecha de fin o un cronograma.
- Planear, sobre todo en este tipo de enfoques iterativos, es la búsqueda del valor

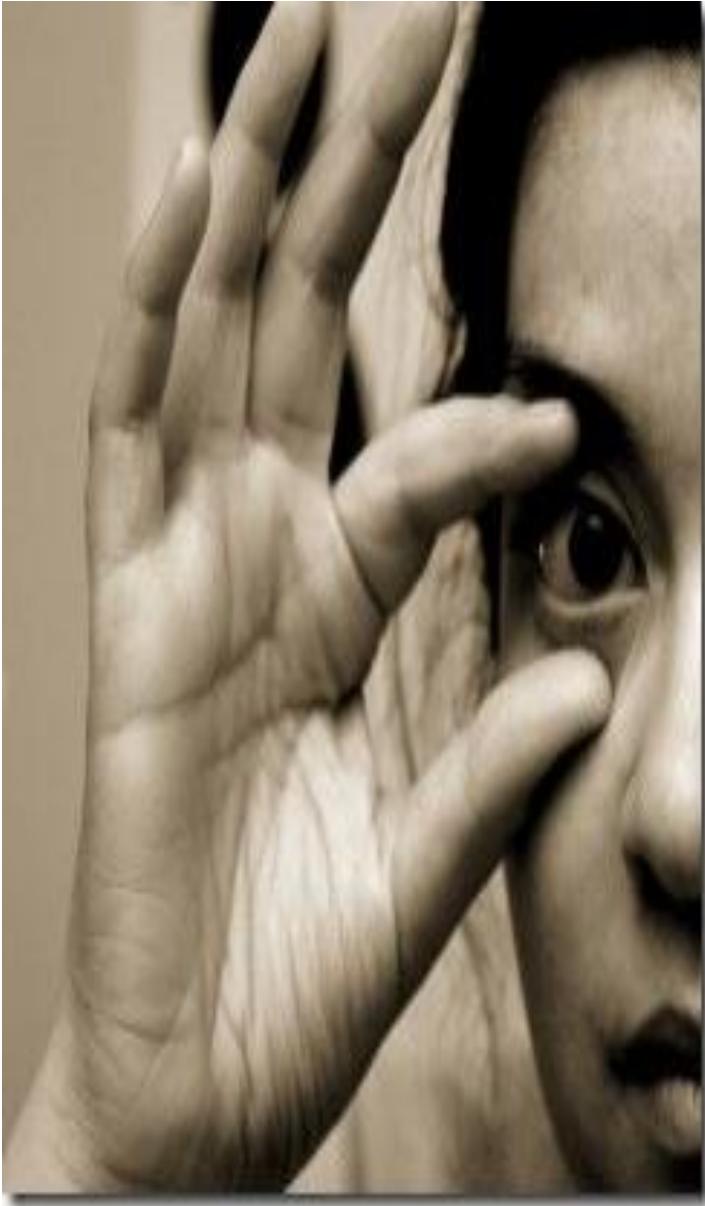


A woman with long brown hair, wearing a light blue shirt, is shown from the chest up. She is looking upwards and to the right with a thoughtful expression, her right hand resting against her chin. A large, semi-transparent dark blue speech bubble originates from her head, pointing downwards and to the right. Inside the bubble, the following text is written in white, sans-serif font.

Un buen plan es aquel que los
stakeholders encuentran lo
suficientemente confiable para
tenerlo como base para tomar
decisiones

Qué es lo que hace ágil una planeación???





¿Cuál es el
enfoque
ágil para
los
proyectos
?

Claves...



Trabajar como un solo equipo



Trabajar en iteraciones cortas



Entregar “algo” en cada iteración



Enfocarse en las necesidades del negocio



Inspeccionar y adaptarse.

Trabajar como un solo equipo



“we’re all in this together”

Roles:

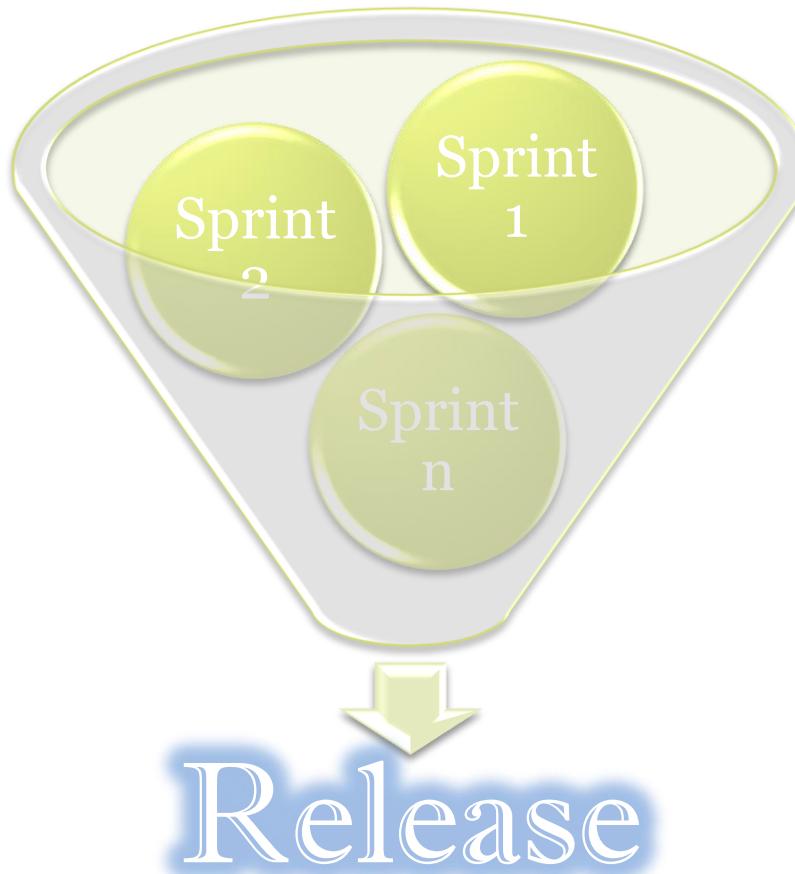
- Dueño del producto
- Scrum Manager
- Miembros del equipo

Trabajar en iteraciones cortas

- Las iteraciones son “time-box”
 - Terminan en tiempo!



Entregar “algo” en cada iteración

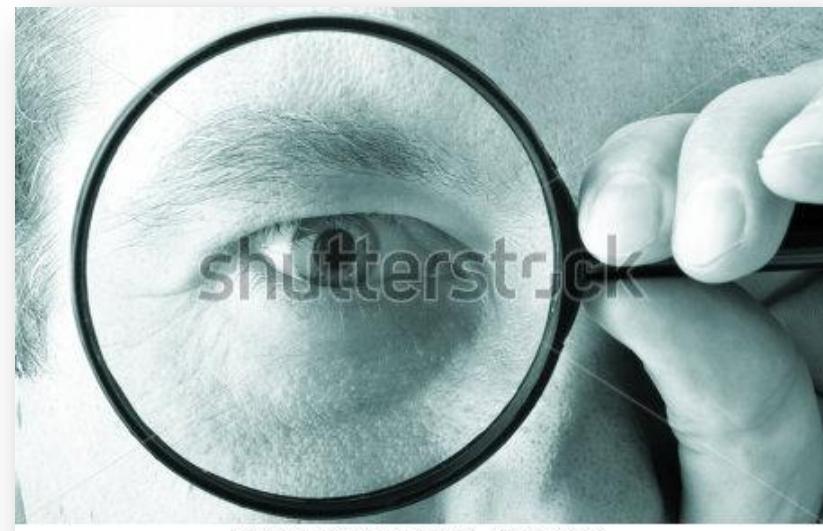


Enfocarse en las necesidades del negocio

- Plan de Release
 - Basado en las capacidades del equipo y
 - Lista priorizada de elementos
- Foco en completar y entregar features de valor para el cliente
 - En vez de terminar tareas aisladas.

Inspeccionar y adaptarse

- Al comienzo de cada iteración
 - Se incorporan los conocimientos adquiridos
 - Si hay algo que afecte la exactitud del plan, entonces se ajusta el plan



IDEA
↓
PLAN
↓
ACTION



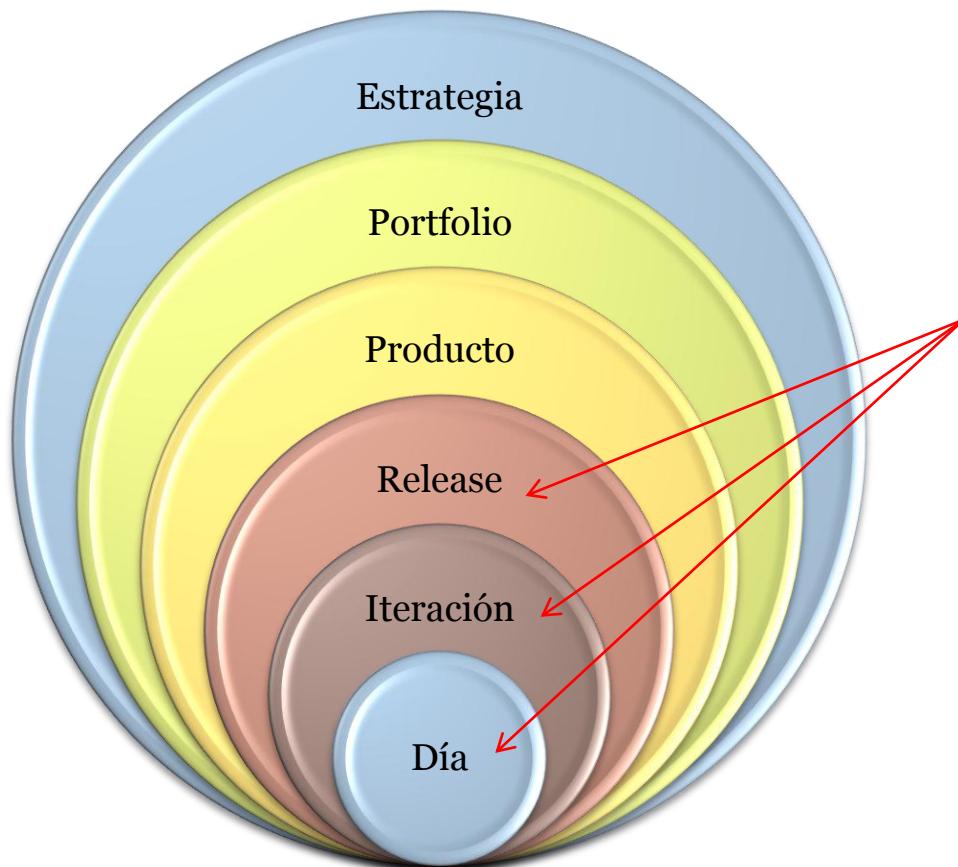
¿Cuál es el
enfoque
ágil para la
planeación?



No deberíamos ver a un proyecto como la sola ejecución de una serie de pasos...

...deberíamos ver a un proyecto como la generación rápida y confiable de **nuevas capacidades** y **nuevo conocimiento**...

Múltiples niveles de planeación..



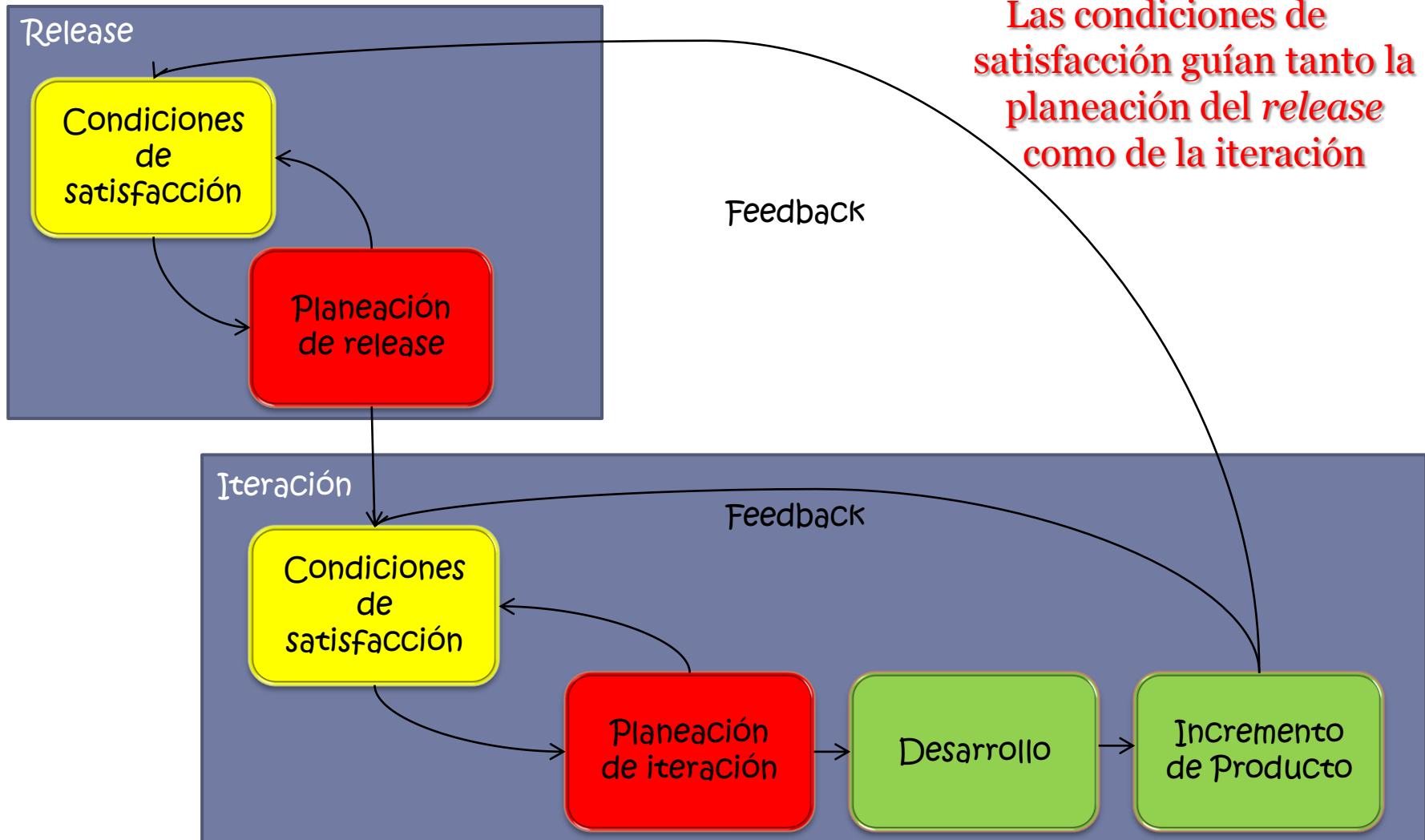
Los equipos ágiles
planean
por lo menos hasta el
nivel de *release*

Al planear en 3 niveles diferentes,
los equipos se enfocan en lo que es
visible e importante

Condiciones de satisfacción



Condiciones de satisfacción



Ejemplo

“Como un usuario, quiero ser capaz de cancelar una reserva”

Historia de usuario

El usuario que cancela con más de 24 hs. de adelantado recibe una devolución completa del dinero

El usuario que cancela con menos de 24 hs. de adelantado recibe una devolución con una multa de \$25.

El código de cancelación se debe mostrar en la aplicación y ser enviado por email al usuario.

Condiciones de satisfacción

Ahora que sabemos todos los conceptos....

Cómo lo implementamos????



Producto

Release 1

Release 2

Release 3

Release n...

Release Backlog

Sprint 1

Sprint 2

Sprint 3

Sprint ?...

Historias de usuarios priorizadas sin estimar!

¿Cómo
hacemos
para
estimar?





“PREDICTION IS VERY
DIFFICULT,
ESPECIALLY ABOUT
THE FUTURE.”

NIELS BOHR, DANISH PHYSICIST



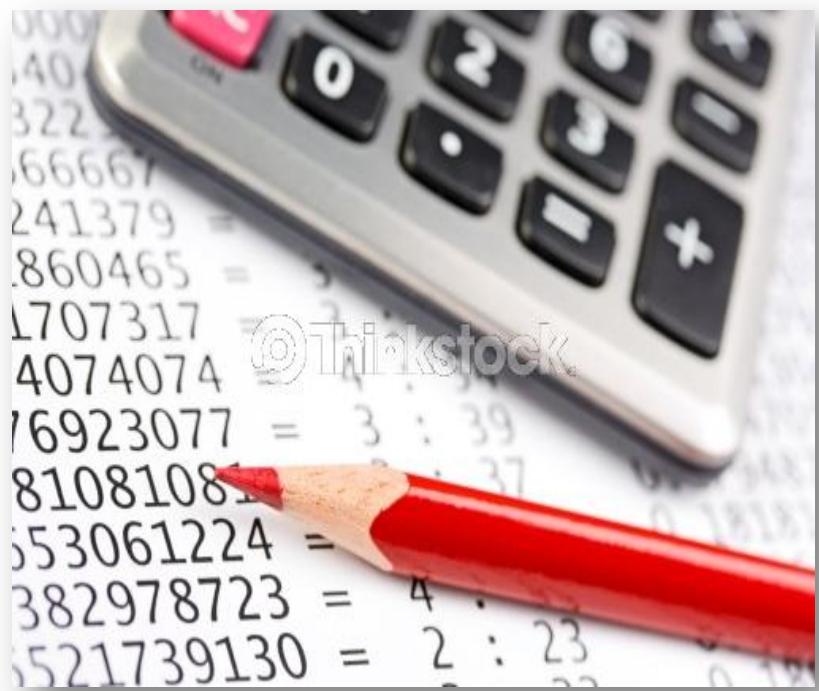
- ¿Sabemos qué es estimar?

....estimar es....

“Una buena estimación es aquella que provee una visión lo suficientemente clara de la realidad de un proyecto para facilitarle al equipo la toma de decisiones para poder conseguir los objetivos propuestos”

- *Steve McConnell*

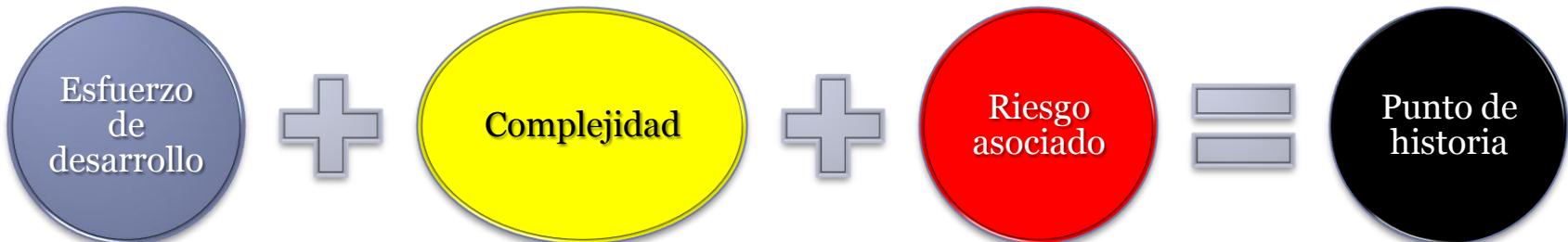
- el propósito es ayudar a mejorar la planificación...



Pero en qué estimamos...?

• Puntos de historia

- Son unidades de medida que expresan el tamaño completo de una historia de usuario.
- El valor “Crudo” que le asignamos no es importante...lo que importa es el **Valor relativo**

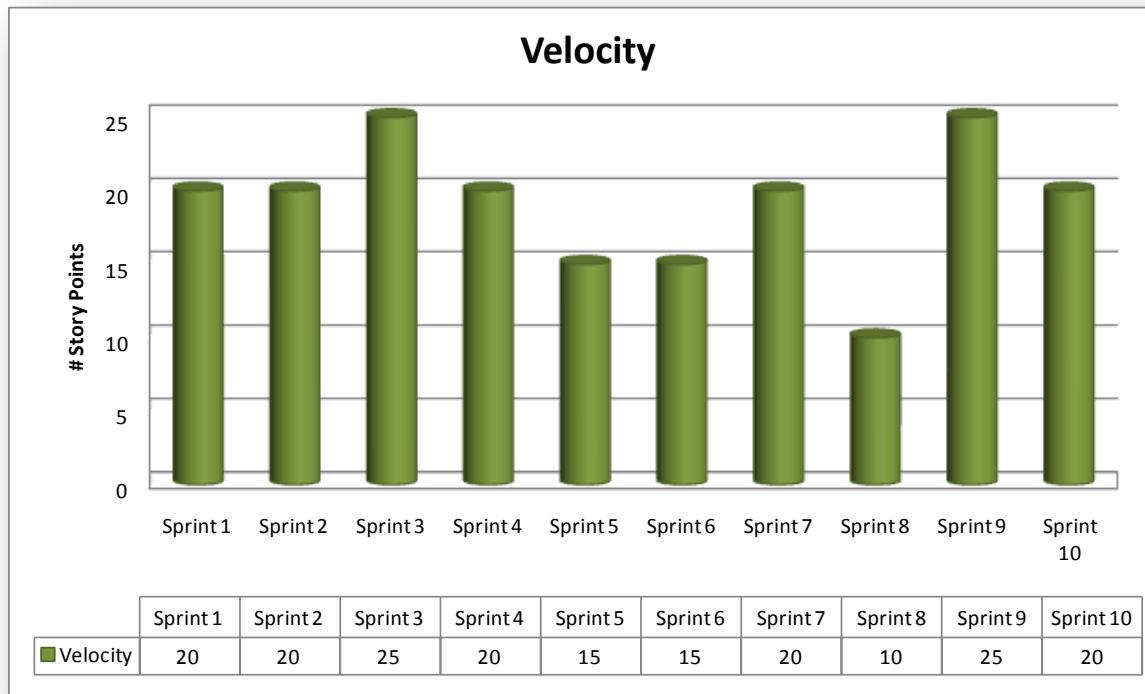


Además de estimar el tamaño...

- **Velocidad**

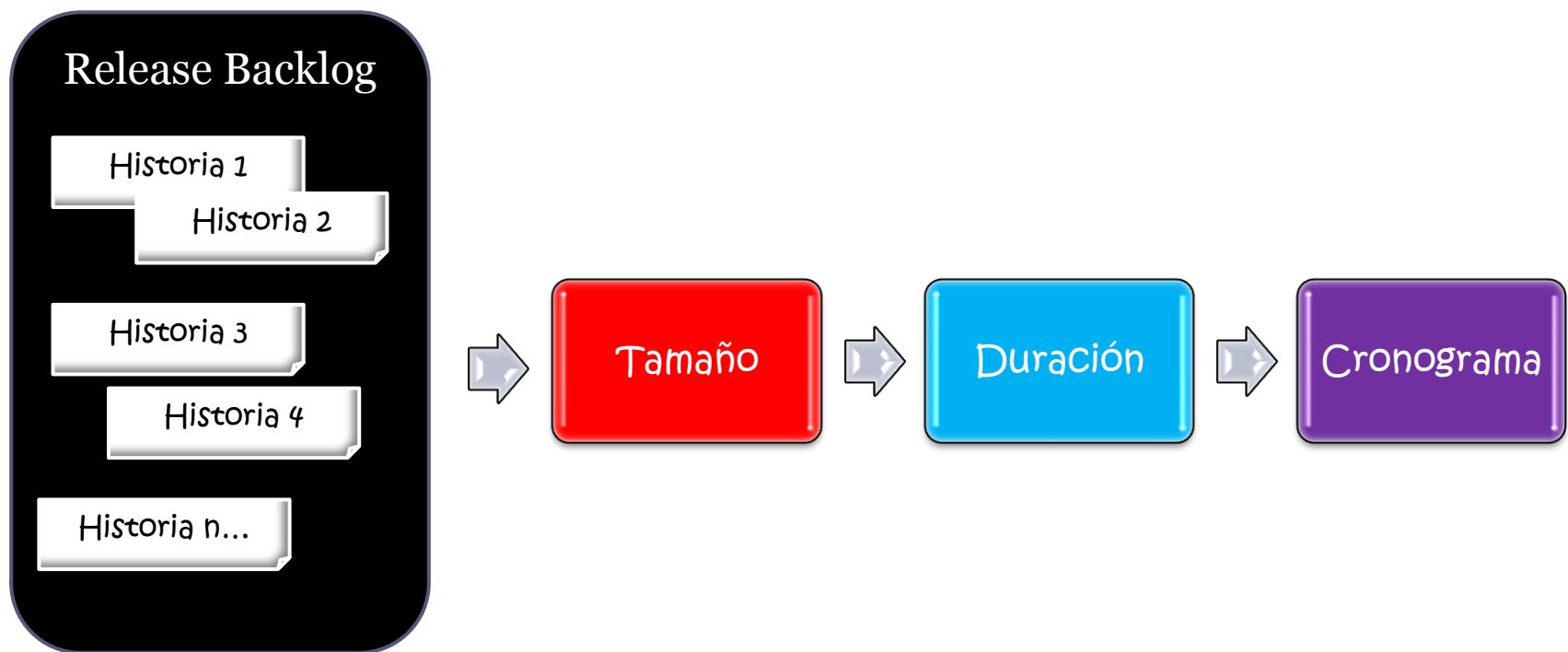
- Permite a los miembros del equipo conocer el **ratio de progreso** del equipo completo.
- **Vel[n] = # puntos de historia por iteración**
- Cuándo? → Se colecta un punto por Sprint.
Se utiliza para la planificación de cada sprint,
y para los ajustes en el *release*.

Velocidad

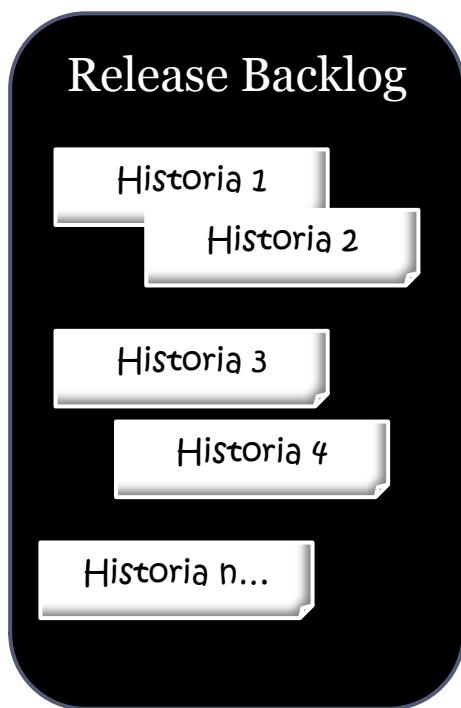


Sirve para corregir errores en las estimaciones

Poniendo todo junto



Ejemplo



Velocidad = 11 sp
(sprint 2 semanas)

Cantidad de iteraciones =>
 $100/11 = 9.1$

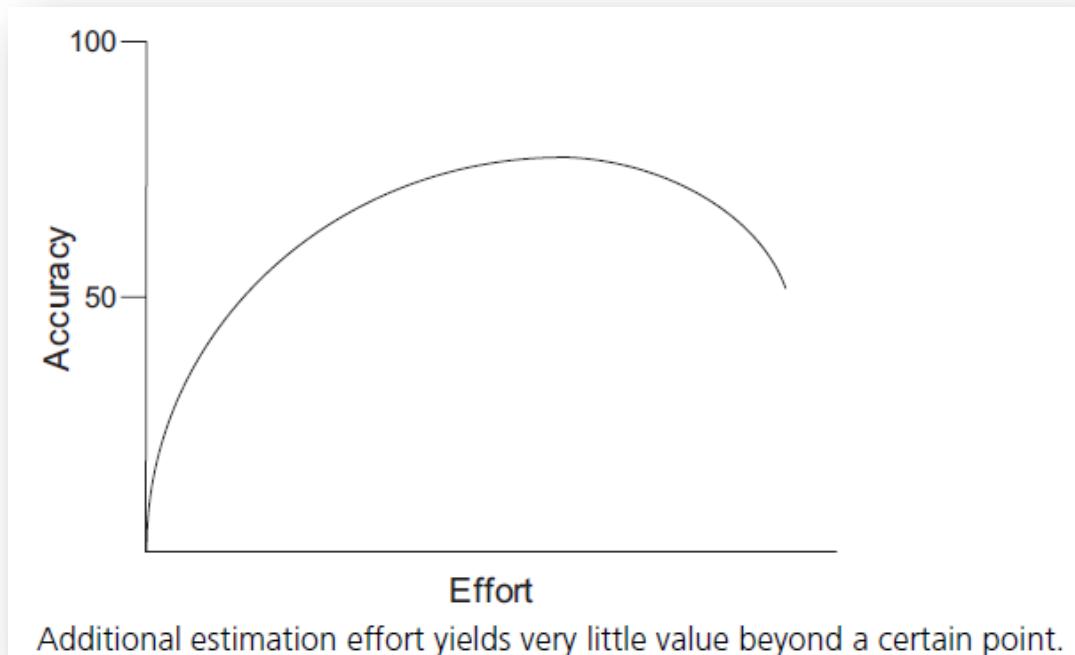
Tamaño

Duración

Cronograma

Estimación = 100 sp

Exactitud vs. Esfuerzo en estimaciones



Sin embargo...tener cuidado!!!

Y qué usamos para estimar...?



Planning Poker



¿Cómo hacemos...?

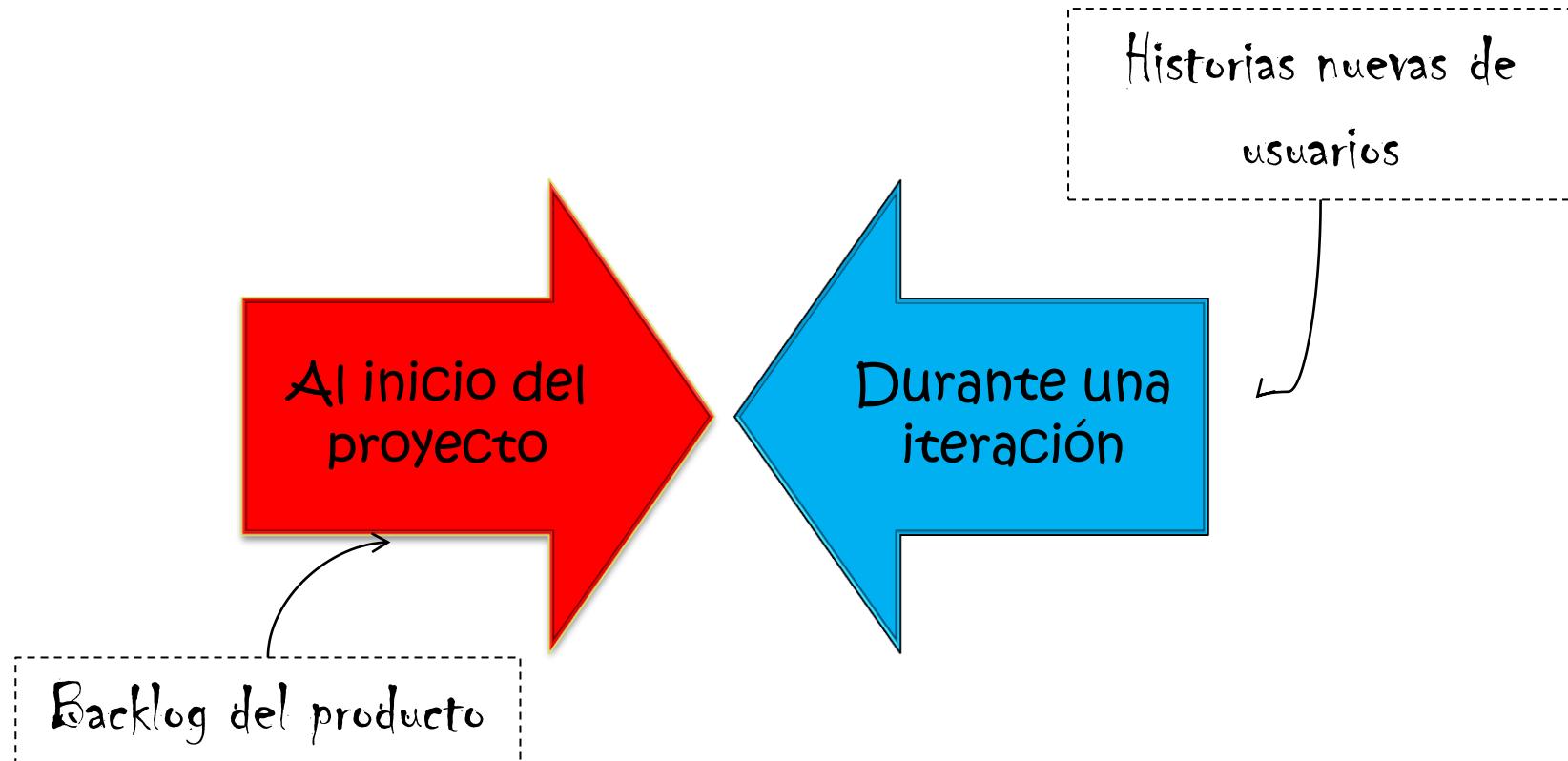
- Moderador * muy importante *



No es derivar una estimación...

El objetivo es caer en algún lugar (del lado izquierdo de la curva de esfuerzo en donde una estimación valiosa se pueda lograr rápidamente

¿Cuándo hacer *planning poker*?



Ejercicio estimaciones

- En grupos, estimar cuantos “puntos de perros” tienen las siguientes razas:

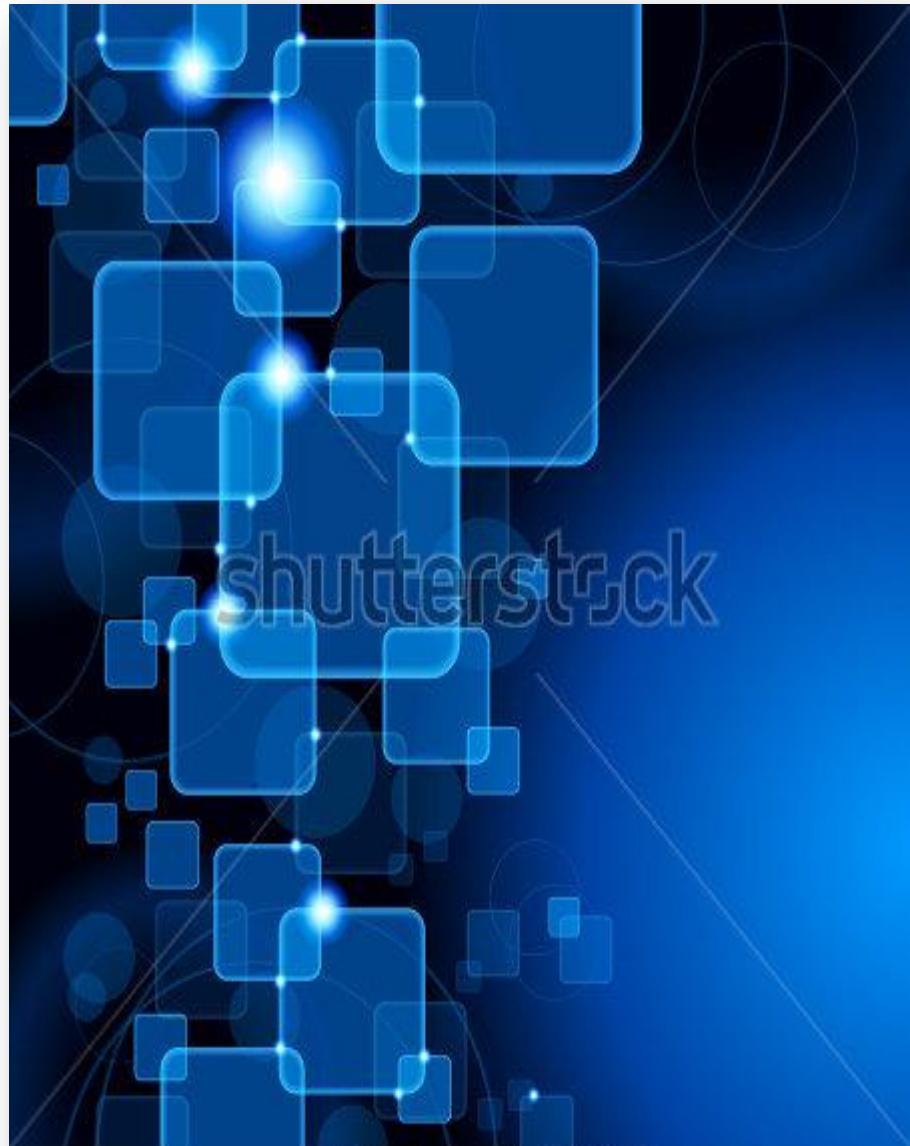
- Labrador Retriever
- Terrier
- Gran Dane
- Caniche
- Dachshund
- Pastor aleman
- San Bernardo
- Bulldog



Ejercicio estimaciones (cont.)

Raza	T1	T2	T3	T4	T5	T6
Labrador Retriever						
Terrier						
Gran Dane						
Caniche						
Dachshund						
Pastor aleman						
San Bernardo						
Bulldog						

Ok...ya
estamos
listos para
comenzar a
desarrollar
?





No

Producto

Release 1

Release 2

Release 3

Release n...

Release Backlog

Sprint 1

Sprint 2

Sprint 3

Sprint ?...

Historias de usuarios priorizadas Y
estimadas

Qué falta entonces????

Release Backlog

Sprint 1

Historia 1

Historia 2

Sprint 2

Historia 4

Sprint 3

Historia 3

Sprint ?...

Historia 5

Sprint Backlog

Historia 1

Historia 2

El equipo descompone las Historias de usuarios en tareas... y volvemos a estimar

Tarea 1

Tarea 2

Tarea 3

Tarea n?...

¿Y ahora en
qué
estimamos?

¿Y qué
usamos para
estimar?



Estimando tareas...

- **Días ideales**

- Es la cantidad de tiempo que lleva hacer algo sacando todas las actividades “extras”.
- Tiempo transcurrido (*elapsed time*) es la cantidad de tiempo que pasa en el reloj



Ejemplo

- Tiempo ideal
 - 4 tiempos de 15 min.
- Tiempo transcurrido (*elapsed time*)
 - 3 horas?
 - Contando faltas, cambios, entretiempos...



Estimando tareas...

- Podemos usar las siguientes técnicas:
 - Juicio de expertos (WBD)
 - PERT
 - Analogía
 - Proxy
 - Etc...



Ok...ahora entonces empezamos a desarrollar?

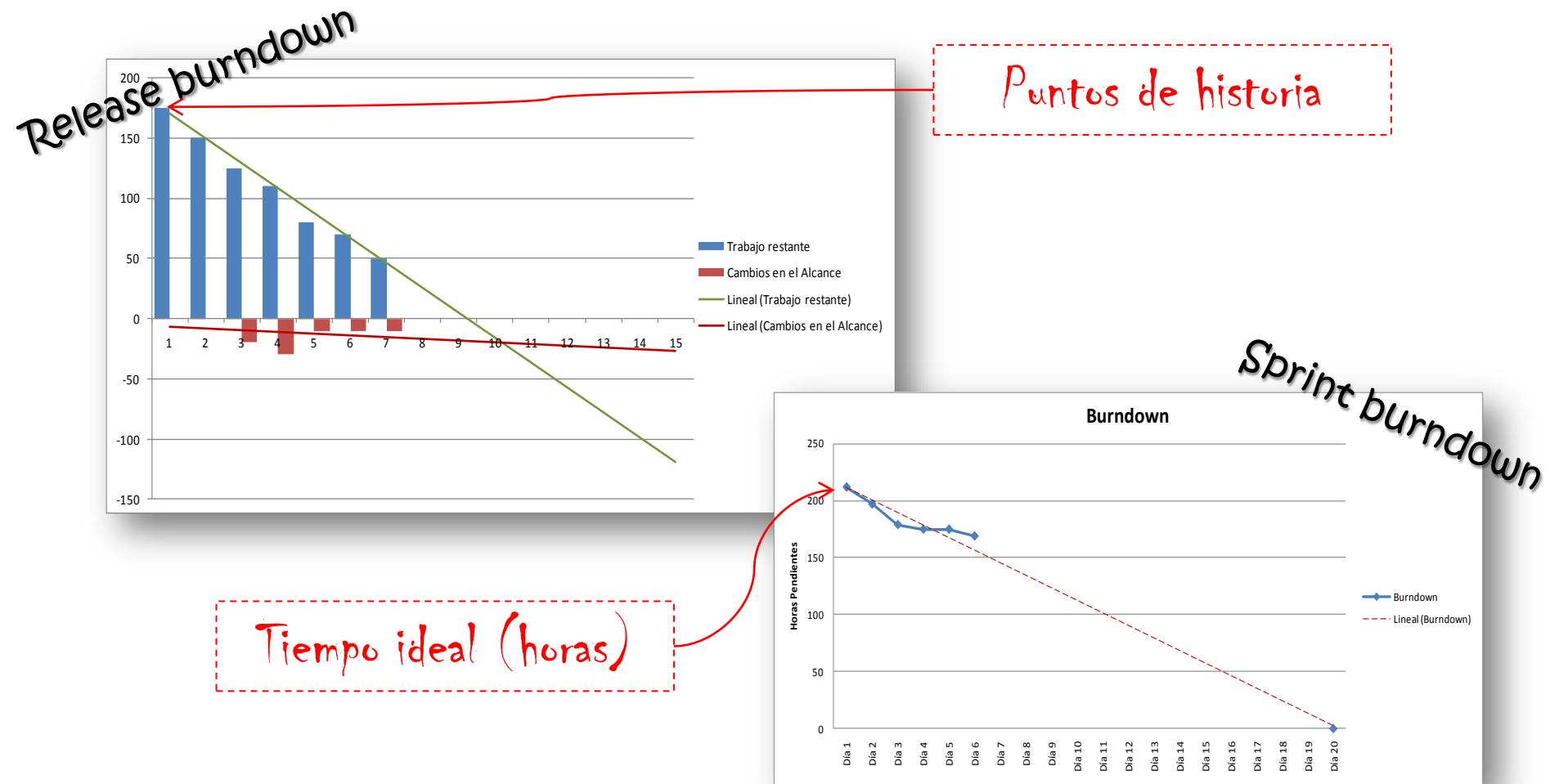
COMPROMISO!!!



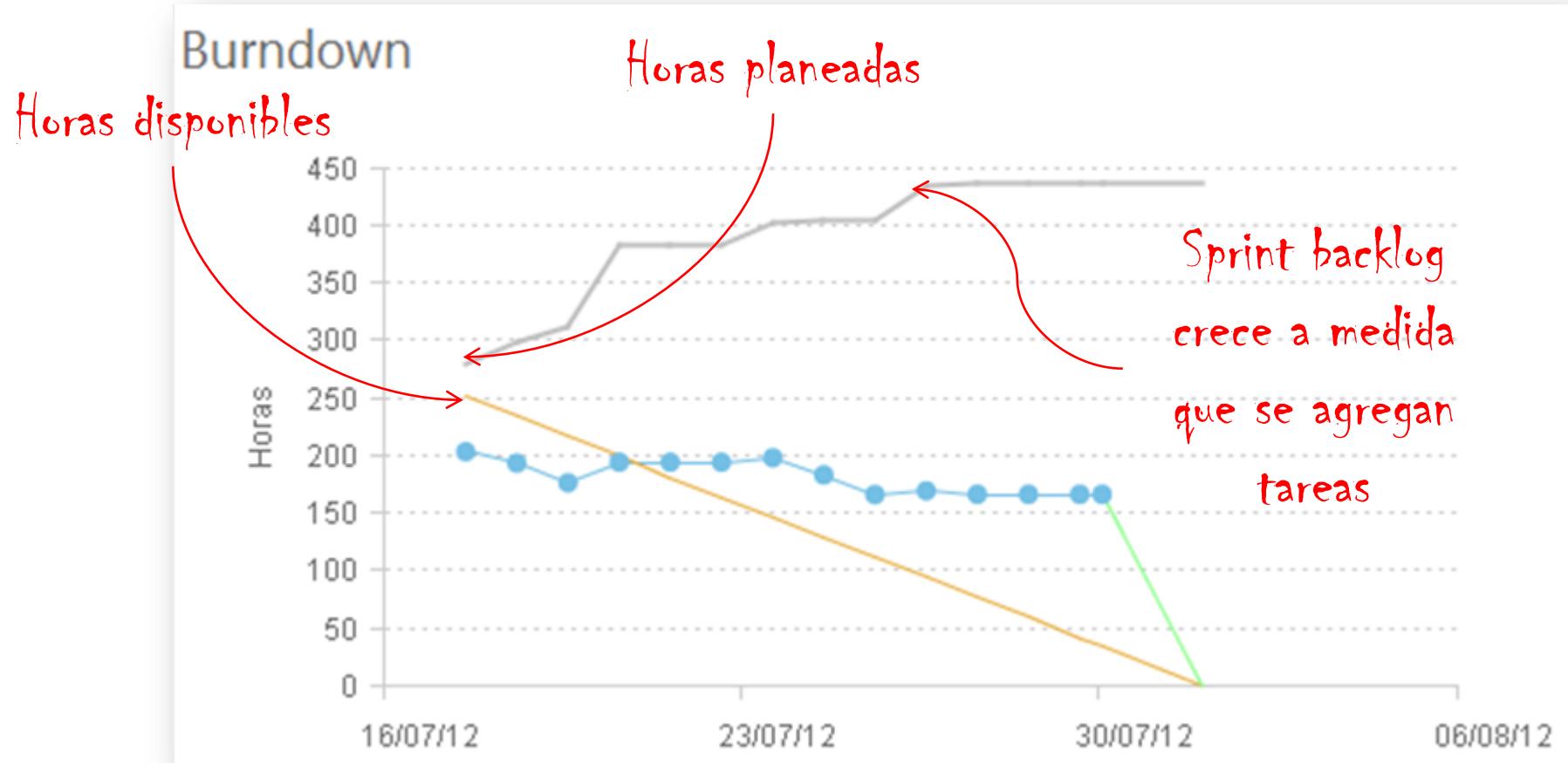
¿Cómo sabemos si estamos avanzando?

	Release Burndown chart	Sprint Burndown chart
Propósito	Ayuda a determinar la probable fecha de finalización del release.	Provee un indicador diario de la velocidad (velocity) del equipo y el progreso respecto del trabajo comprometido para el sprint actual.
Fórmula	Intersección de las líneas de tendencia de la velocidad del equipo y el trabajo introducido al release.	Tendencia de Regresión Lineal Trabajo Pendiente + Tendencia = Trabajo pendiente del día anterior a ayer + Trabajo pendiente de ayer + Trabajo pendiente de hoy) / 3
Cuándo?	Se recolecta un punto por Sprint. Se utiliza para la planificación de cada sprint, y para los ajustes en el release.	Se colecta una vez por día . Es la métrica principal de seguimiento del Sprint.

¿Cómo sabemos si estamos avanzando?



Ejemplo Sprint Burndown





Ya hicimos:

- Requerimientos,
- Planificamos
- Estimamos
- Tomamos
métricas

Ahora qué sigue?

1

Testing Ágil »

Objetivos del Testing
Errores, Fallas y Defectos
Principios del Testing

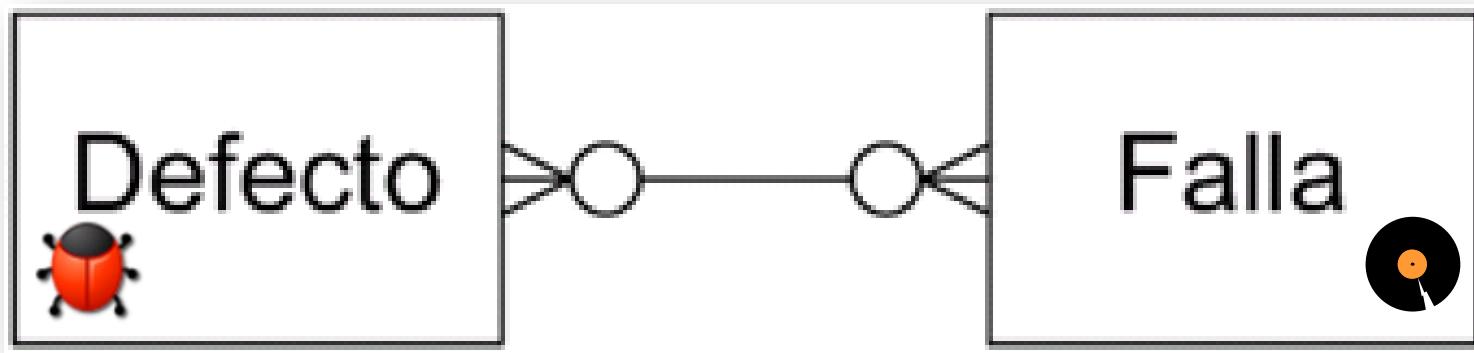
Objetivos del Testing

- Testing NO es:
 - Demostrar que NO hay errores
- Testing ES:
 - Proceso destructivo de tratar de encontrar defectos (cuya presencia se asume!) en un programa
 - Brindar confianza en la calidad del producto

Definiciones de Error

- Equivocación
 - Una acción humana que produce un resultado incorrecto
- Defecto
 - Paso, proceso, o definición de dato incorrecto
 - Ausencia de cierta característica
- Falla
 - Resultado de ejecución incorrecto. Es el producido por el software distinto al resultado esperado

Relación entre Defecto y Falla



- Un **defecto** puede producir 0, 1 o más **fallas**
- Una **falla** puede estar causada por 0, 1 o más **defectos**

Principio N° 1

- El testing muestra la existencia de defectos:
 - No demuestra la ausencia de defectos
 - Reduce la probabilidad de que queden defectos sin descubrir



Principio N°2

- Es imposible probar todo
 - Testear todas las combinaciones y posibilidades es imposible, por ello se debe priorizar y realizar análisis de riesgos para enfocar los esfuerzos del equipo de test.



Principio N° 3

- Comenzar las pruebas lo antes posible
 - Las actividades de test deben comenzar lo antes posible en el ciclo de vida de desarrollo de software.
 - Encontrar y corregir defectos lo antes posible es menos costoso y ayuda a prevenir que aparezcan defectos en etapas posteriores.



Principio N° 4

- Los defectos se agrupan
 - Un pequeño número de módulos contienen la mayoría de los defectos
 - Debemos enfocarnos en las módulos más propensos a contener errores
 - Hacer análisis de causa raíz ayuda a identificar las causa del agrupamiento de los defectos
 - Ojo! Esas agrupaciones cambian a lo largo del tiempo

Principio N° 5

- La paradoja de plaguicidas
 - Por mucho que se repitan los mismos tests, no se van a encontrar más errores.
 - Por ese motivo los test cases necesitan ser regularmente revisados y se deben ir añadiendo nuevos para probar las diferentes partes del software.



Principio N° 6

- **Las pruebas son dependientes del contexto**
 - Los procesos y las metodologías de test aplicadas deben ser diferentes según el contexto.
 - No todos los sistemas de software llevan el mismo nivel de riesgo y no todos los problemas tienen el mismo impacto cuando se producen.

Principio N°7

- **La falacia de la ausencia de defectos**
 - Encontrar y corregir defectos no ayuda si el sistema construido no se puede utilizar y no cumple las necesidades de los usuarios y las expectativas.

2

Testing Ágil »

El Manifiesto Ágil
¿Qué cambia?
Factores claves

Agile manifesto

¿Se acuerdan?

Entendiendo el Agile Manifesto

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.

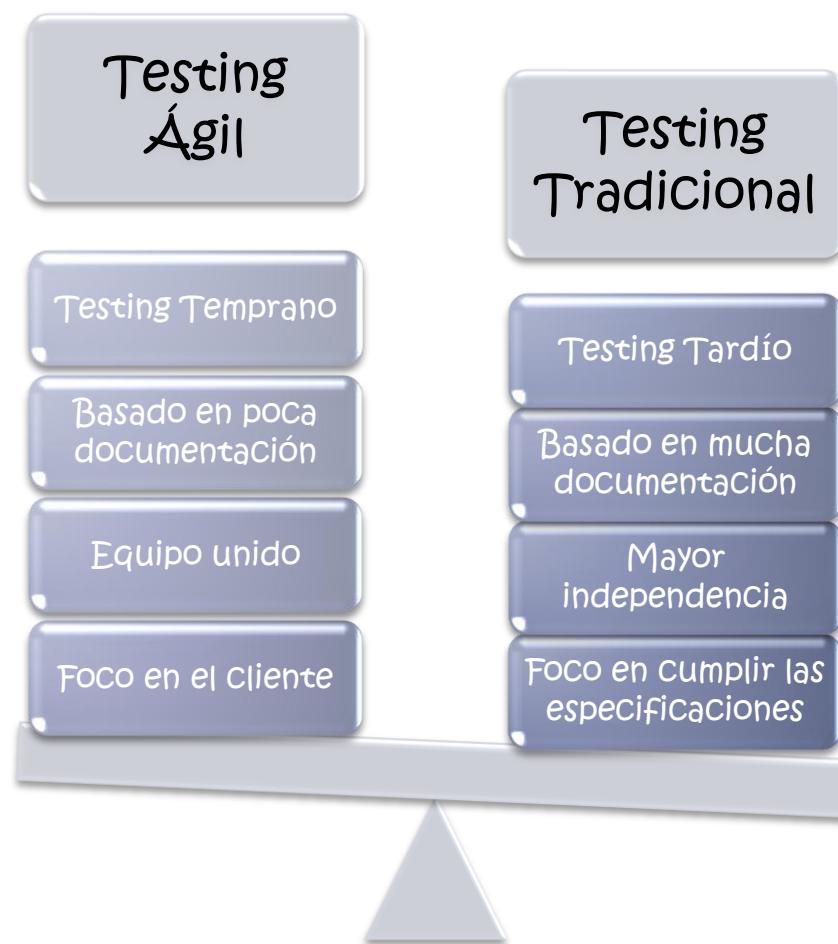
Kent Beck Mike Beedle Arie van Bennekum Alistair Cockburn Ward Cunningham Martin Fowler	James Grenning Jim Highsmith Andrew Hunt Ron Jeffries Jon Kern Brian Marick	Robert C. Martin Steve Mellor Ken Schwaber Jeff Sutherland Dave Thomas
--	--	--

Draft: <http://agilemanifesto.org/>

Diego Rubio –MADS – Sprint 0

Buscamos hacer entregas frecuentes en ciclos muy cortos y que agreguen valor al negocio

Testing Ágil vs Testing Tradicional



¿Qué cambia?

- No existen documentos extensos y detallados describiendo las funcionalidades que deben ser testeadas.
- Las pruebas comienzan antes en el ciclo de vida, incluso antes de estar listo el código.
- Mucho énfasis en la automatización de las pruebas de regresión.

¿Qué cambia? - Mentalidad

- ✓ El testing ocupa una parte muy importante en el proceso de desarrollo
- ✓ Automatización de las pruebas: absolutamente necesarias
- ✓ Cambios en los roles
- ✓ Ya no existen equipos de aseguramiento de la calidad separados
- ✓ Equipos altamente colaborativos
- ✓ Es preciso desarrollar capacidades de programación

¿Qué cambia? - Foco

- Los casos de prueba están más alineados al negocio
- Se ejecutan primero los casos que prueban las funcionalidades que agregan más valor al cliente
- Retroalimentación inmediata a los desarrolladores acerca de los defectos encontrados
- Build e integración continua
- Las historias son definidas pensando en las pruebas

¿Qué cambia? - El rol

Se adapta a cambios en

prioridades o requerimientos

herramientas y técnicas

Colabora con

desarrolladores

expertos del dominio

Entiende que sus tests sirven para

documentar el producto

dirigir el desarrollo

Entiende el negocio

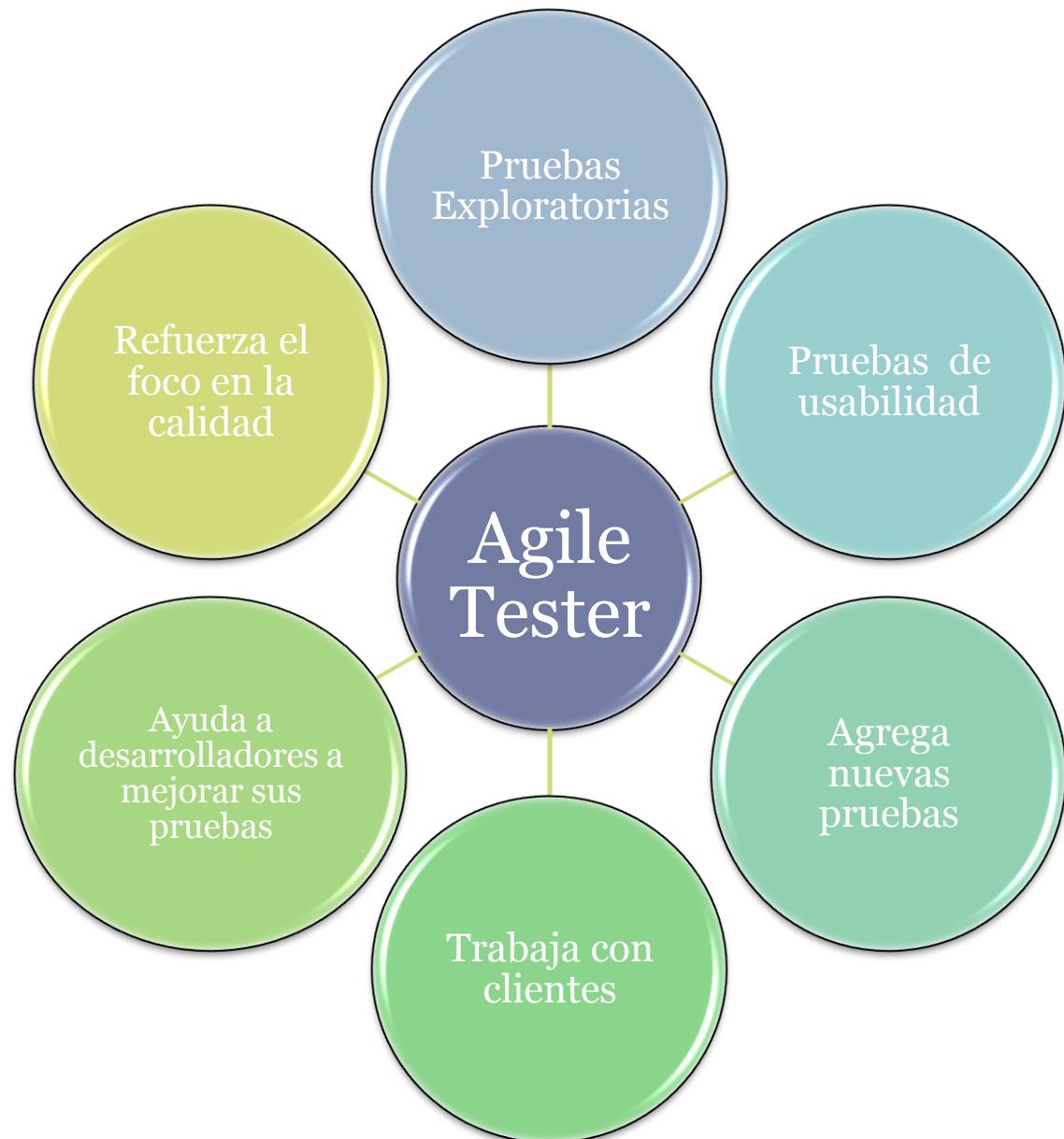
comunicándose con los clientes

escribiendo sus pruebas con ese foco

¿Qué cambia?

En metodologías ágiles, los desarrolladores escriben sus propios test...

Entonces, que valor agrega un tester??



Actividades de testing

Iniciación
del proyecto

Entender
el proyecto

Planificación
del release

Participar
en las
estimaciones

Crear el plan
de prueba

Cada
iteración

Escribir y
ejecutar
casos de
prueba

Trabajar de a
pares

Automatizar
y ejecutar las
pruebas

Revisar
métricas

Pruebas
nivel de
sistema

Pruebas de
Carga

Pruebas de
regresión

Pruebas de
aceptación

Entrega /
Soporte

Participar
en la entrega
a
producción

Participar
en las
retrospectivas

Factores claves en el testing ágil



Probar temprano



Probar seguido



Regresión



Perspectiva del cliente



Automatización



Comunicación

¿Qué cambia?

Los tiempos

La relación con el equipo de trabajo

Las técnicas que podemos usar

Las habilidades que vamos a necesitar

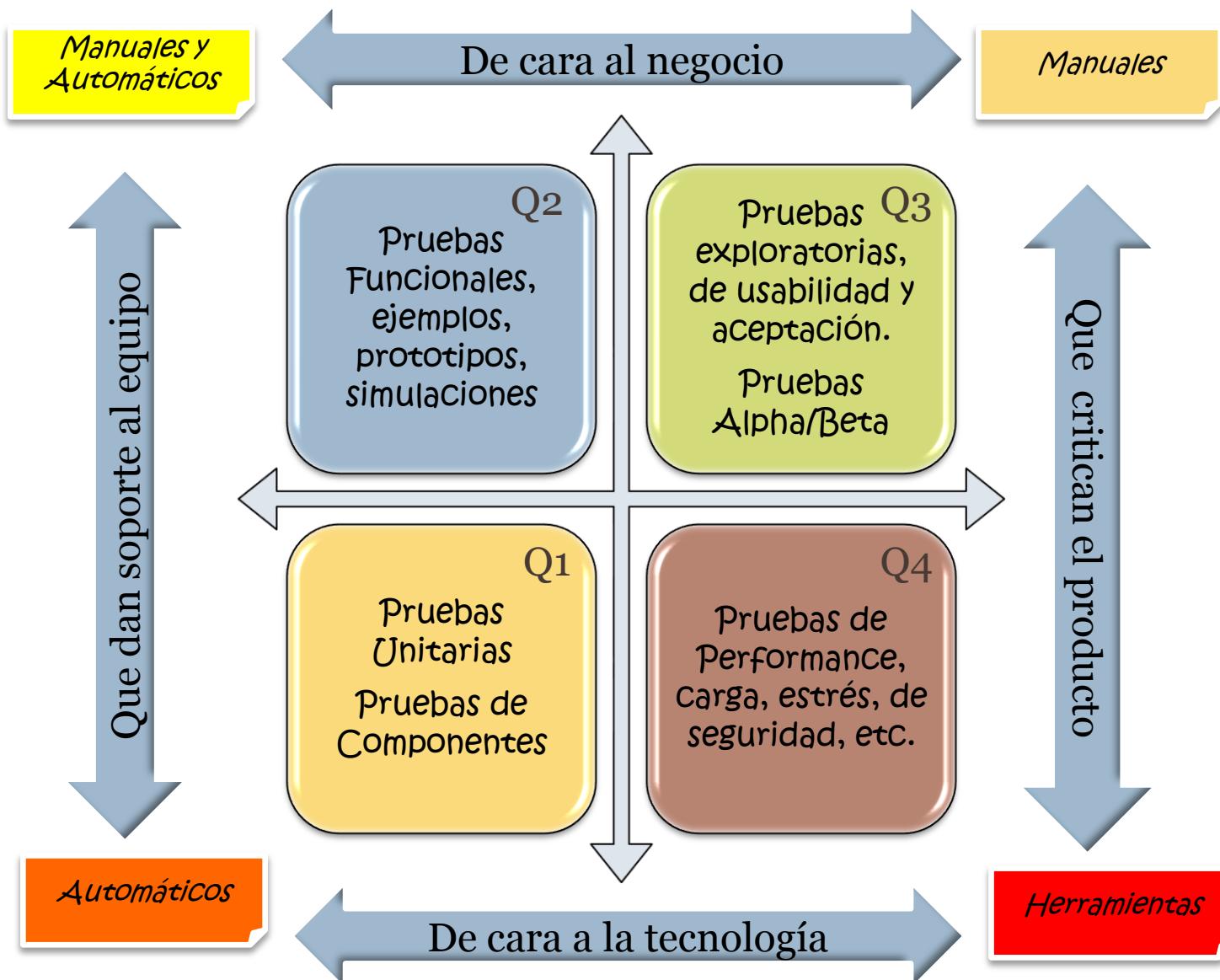
El foco ...

3

Testing Ágil »

Los cuadrantes del Testing Ágil

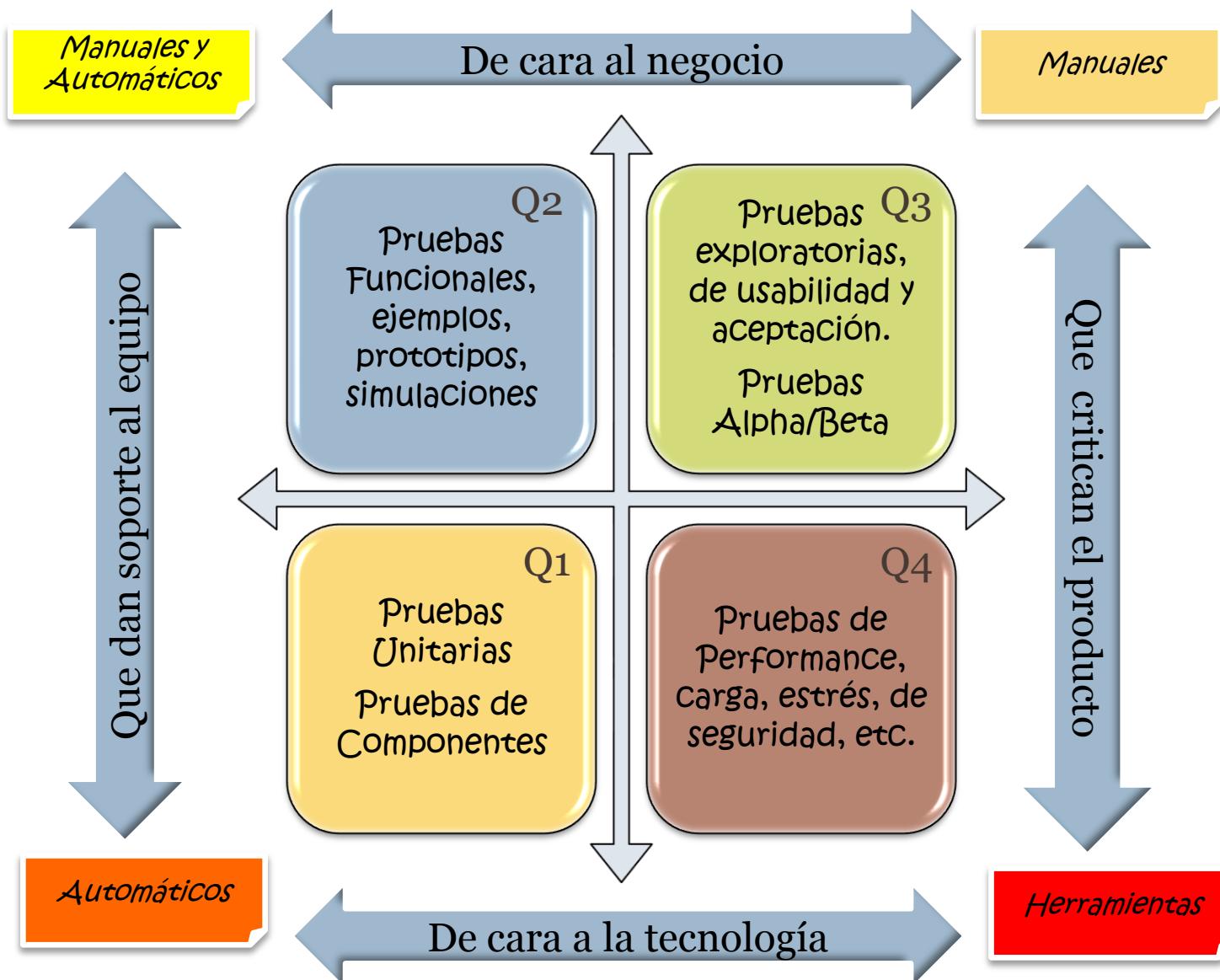
Los cuadrantes del Testing Ágil



Vamos de lo mas simple...

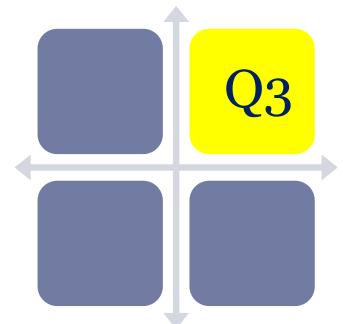
A lo mas complejo

Los cuadrantes del Testing Ágil



Tercer Cuadrante (Q3)

- Pruebas orientadas al Negocio



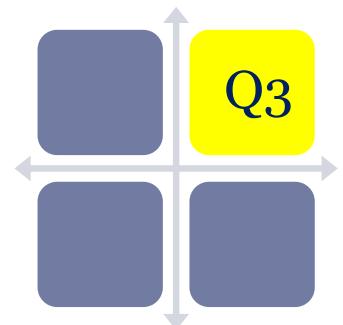
Se basa en criticar el producto una vez construido.



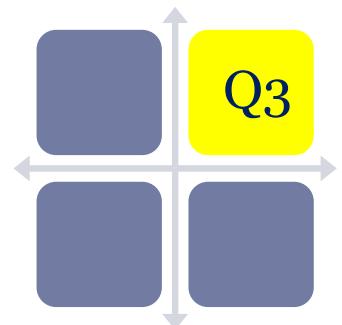
Se basa en criticar el producto utilizando experiencias reales.

Tercer Cuadrante (Q3)

- Pruebas orientadas al Negocio
 - Demostraciones

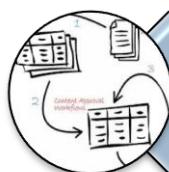


Es recomendable mostrar al cliente qué se está haciendo tempranamente y en forma periódica



Tercer Cuadrante (Q3)

- Pruebas orientadas al Negocio
 - Pruebas de Escenarios



Definir flujos de trabajo (*workflows*)



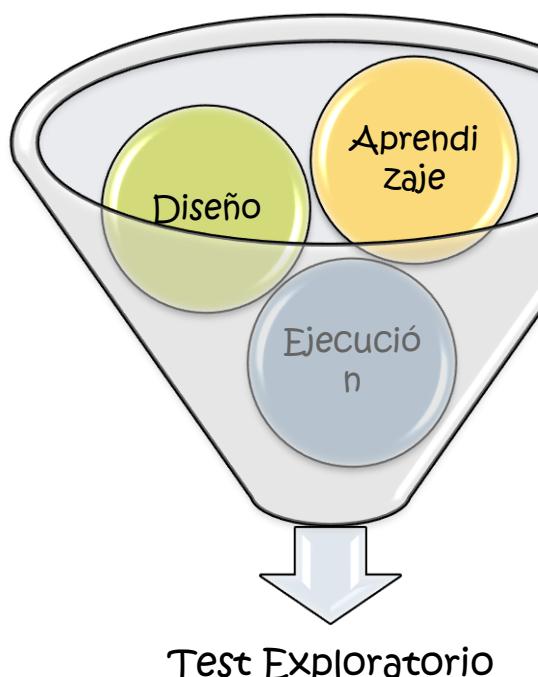
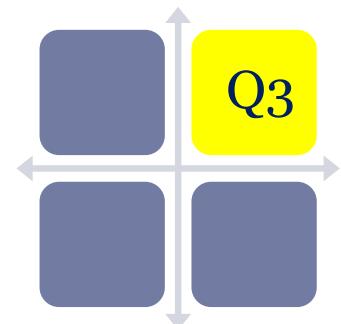
Soap Opera

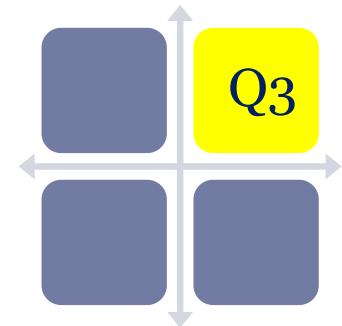


Se necesitan datos y flujos reales

Tercer Cuadrante (Q3)

- Pruebas orientadas al Negocio
 - Pruebas exploratorias





Tercer Cuadrante (Q3)

- Pruebas Exploratorias

¿Qué no es?



Es una técnica en sí misma?

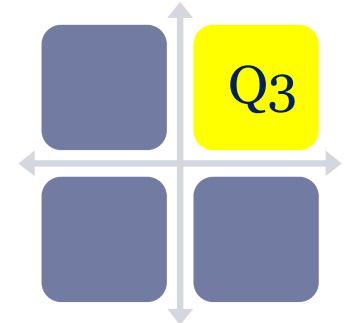


No es sólo ejecución de pruebas



No es ejecutar pruebas desprolijas

Test Exploratorio

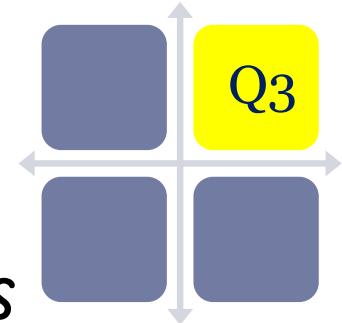


¿Qué hace que una actividad de testing sea exploratoria?

**La capacidad y
compromiso del tester**

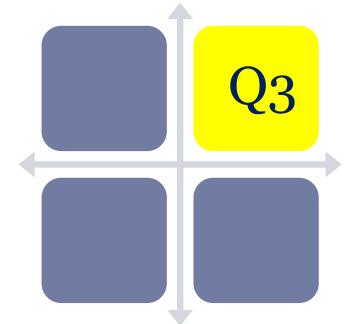
Test Exploratorio

Cuando desarrollamos pruebas exploratorias nos basamos en:



Test Exploratorio

Componentes necesarios para realizar buenas pruebas exploratorias:



Lecturas

Obligatorias

Planning poker → <http://www.mountaingoatsoftware.com/topics/planning-poker>

Scrum → <http://www.mountaingoatsoftware.com/topics/scrum>

Release burndown chart → <http://www.mountaingoatsoftware.com/scrum/release-burndown>

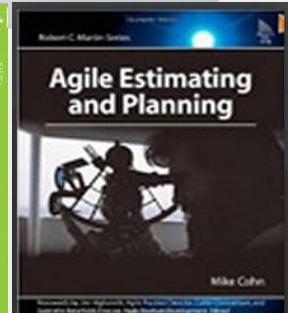
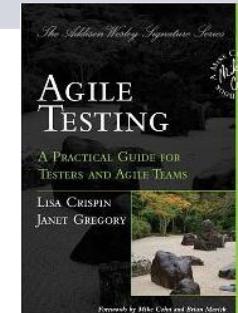
Recomendadas

Scrum → <http://jeffsutherland.com/scrumhandbook.pdf>

Bibliografía

Mike Cohn; Agile Estimating and Planning ; Prentice Hall; 2006; 0-13-147941-5

- Ken Schwaber; Scrum Development Process; 1995
- Ken Schwaber and Jeff Sutherland; Scrum Guide; Scrum Alliance; 2010
- Kent Beck; Embracing Change with Extreme Programming; IEEE; 1999
- Brent Barton et al.; Reporting Scrum Project Progress to Executive Management through Metrics; Scrum Alliance; 2005
- Victory Szalvay et al; Agile Transformation Strategy; Danube; 2005
- Jeff Sutherland et al.; Scrum and CMMI Level 5: The Magic Potion for Code Warriors; 2007;
- Mary and Tom Poppendieck; Lean Software Development: An Agile Toolkit; Addison-Wesley; 2003; 0-321-15078-3
- <http://www.ambyssoft.com/essays/whyAgileWorksFeedback.html>
- Agile Testing - A practical guide for testers and agile teams/ auth. Lisa Crispin - Janet Gregory. - Boston : Pearson Education Inc., 2009. - ISBN-13: 978-0-321-53446-0.



Versión

Versión	Fecha	Comentarios	Autor
1.0.0_Draft_A	24-Jul-2012	Versión inicial	Natalia Andriano
1.0.0_Draft_B	27-agosto-2012	Cambios según comentarios de Diego, agregué un ejercicio de planning poker	Natalia Andriano
1.0.0_Draft_C	11-sep-2012	Se agregó testing. Material adaptado del esting Ágil. Marcela Garay Moyano y Luciano Marzo.	Paula Izaurrealde
1.0.0	14-sep-2012	Baseline	Natalia Andriano