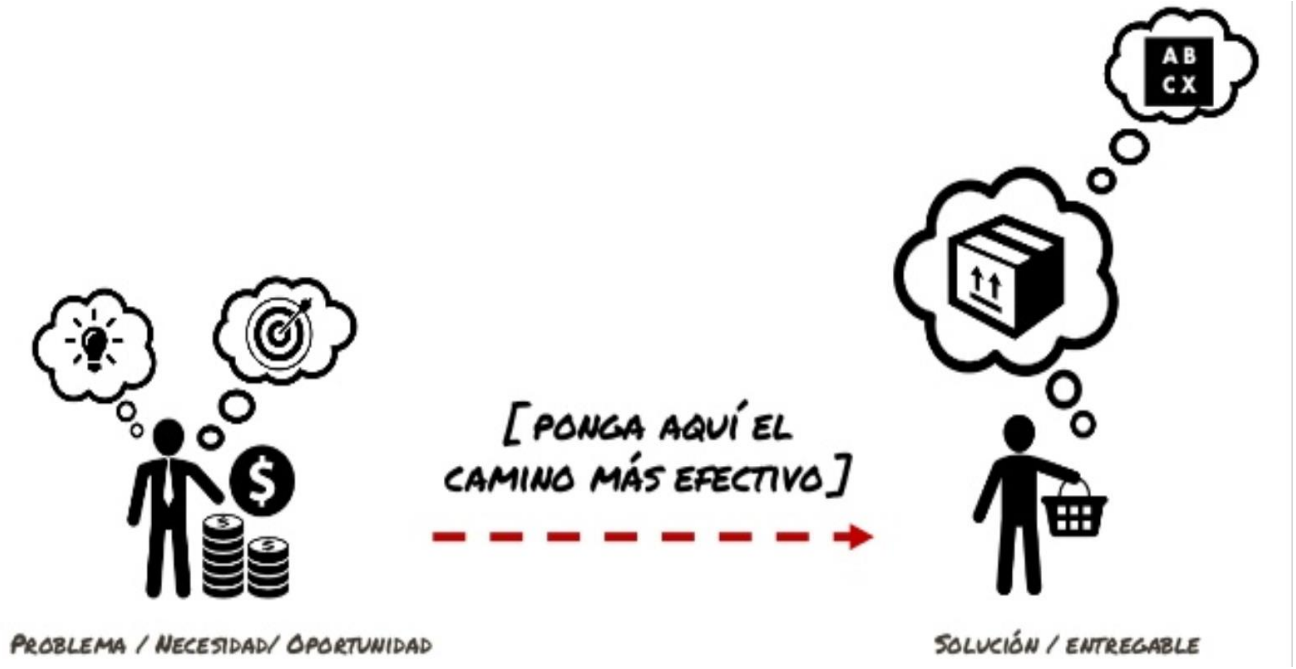
Requerimientos Ágiles



MANIFESTO FOR AGILE SOFTWARE DEVELOPMENT

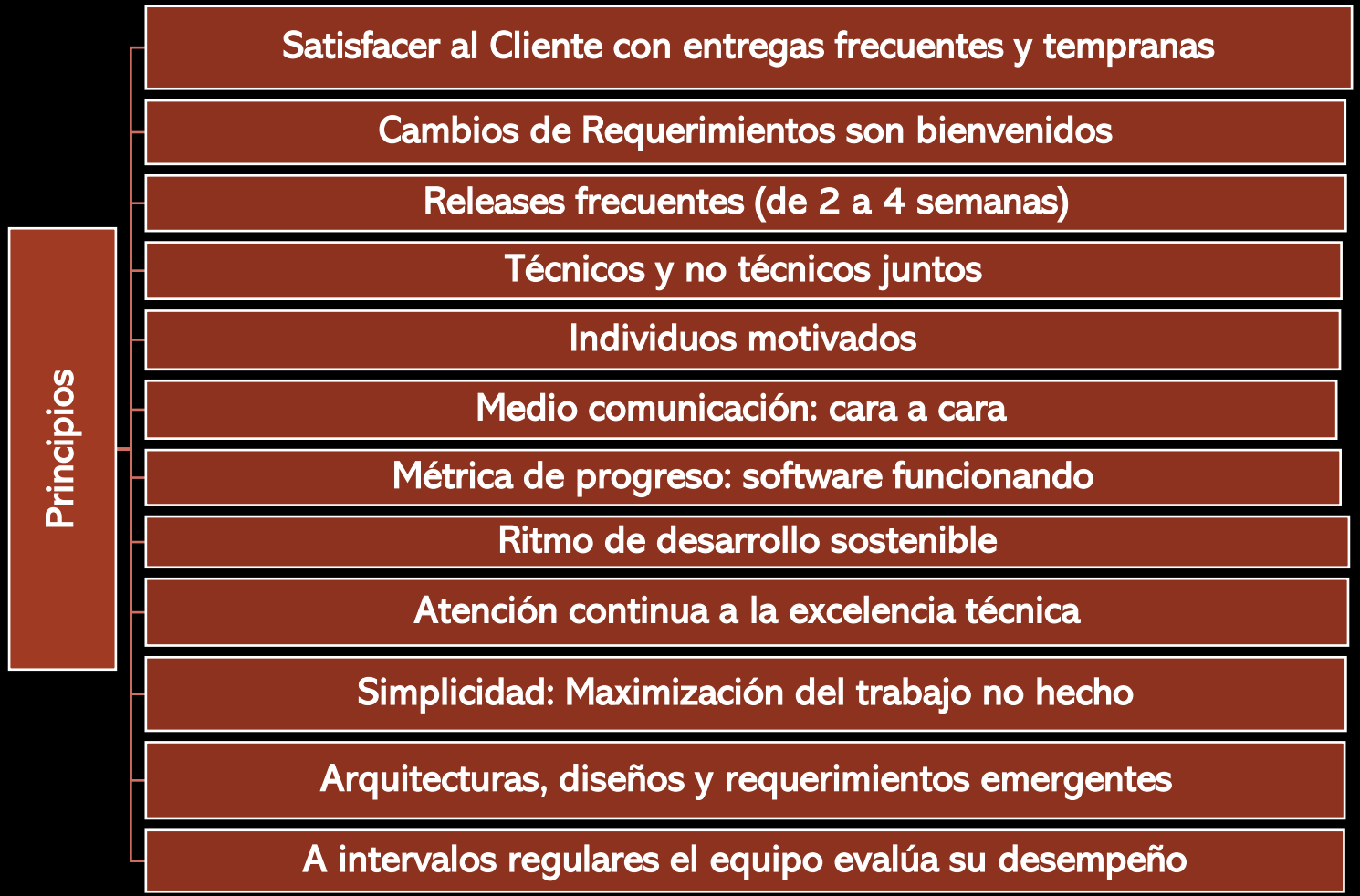
We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:

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

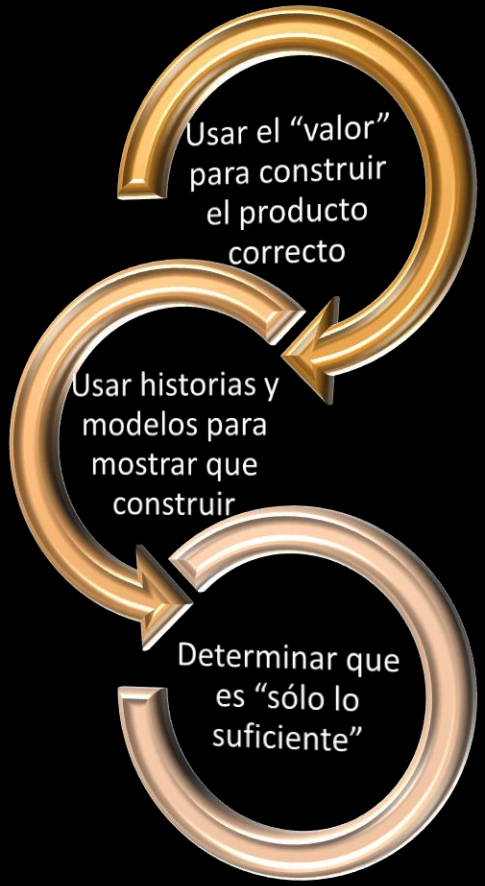
That is, while there is value in the items on the right, we value the items on the left more

Andy Martin Ron Brian Alistair JEFF Jim Robert David Kent Ken

Los 12 principios del Manifiesto Ágil

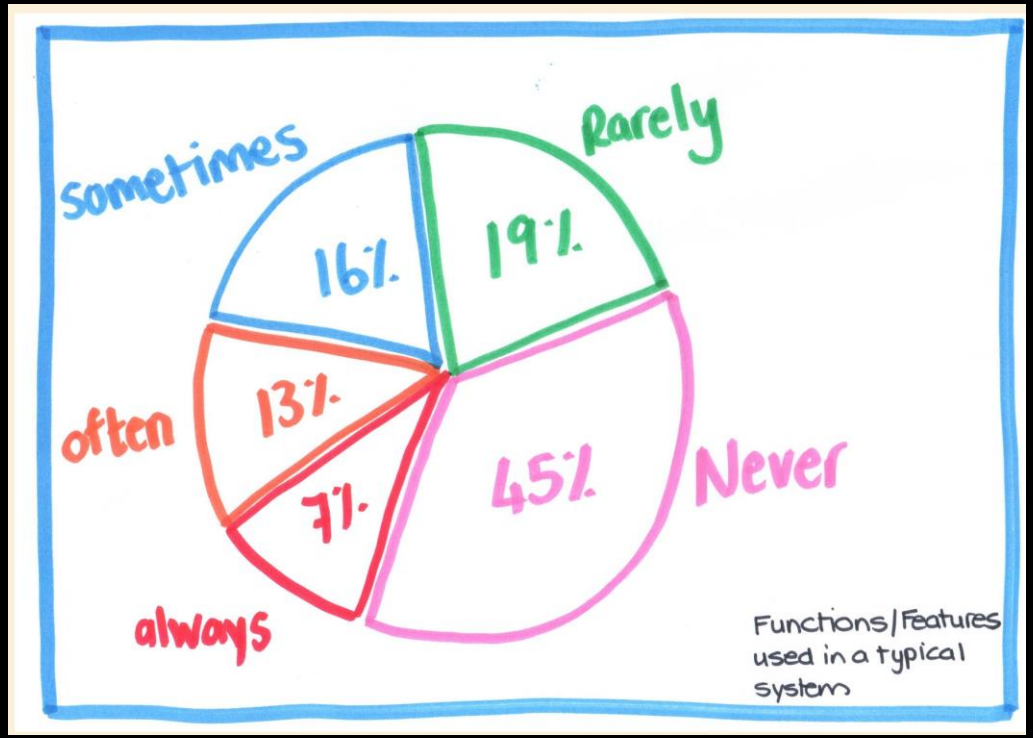


Requerimientos en Agile



El costo del tradicional BRUF

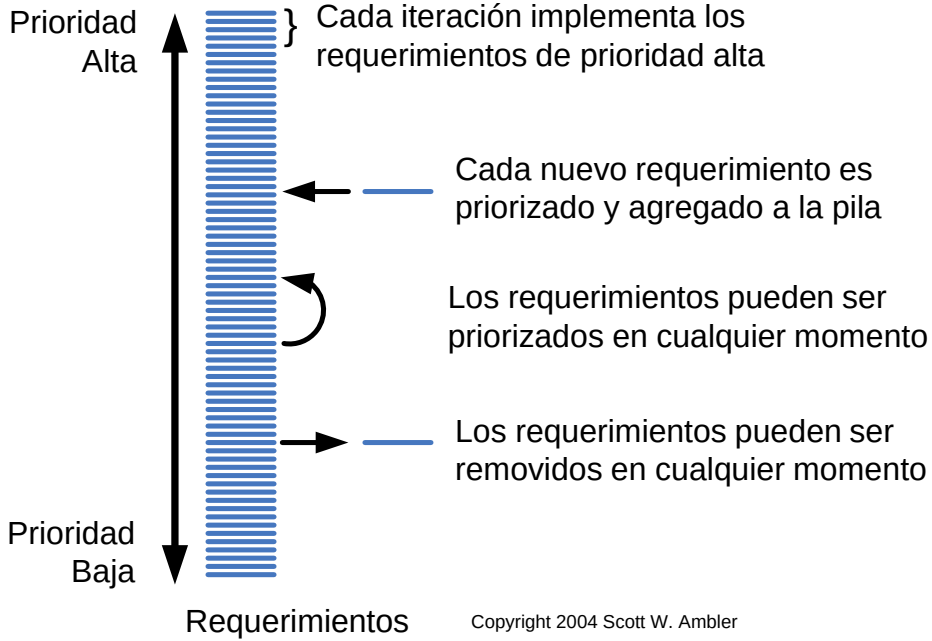
Los productos “Exitosos” también tienen un desperdicio significante



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



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.	Colaborativo con stakeholders de mayor interés (clientes, usuarios finales).
Equipo	Analista de Negocios, Project Manager y Áreas de Proceso.	Equipo multidisciplinario.
Herramientas	Entrevistas, observación y formularios.	Principalmente prototipado. Técnicas de facilitación para descubrir.
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

Tipos de Requerimientos

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

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

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.

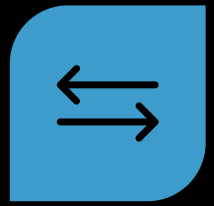
Siempre se cumple eso de que: “El usuario dice lo que quiere cuando recibe lo que pidió”.

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”.

Principios Ágiles relacionados a los Requerimientos 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

Universidad Tecnológica Nacional

Facultad Regional Córdoba

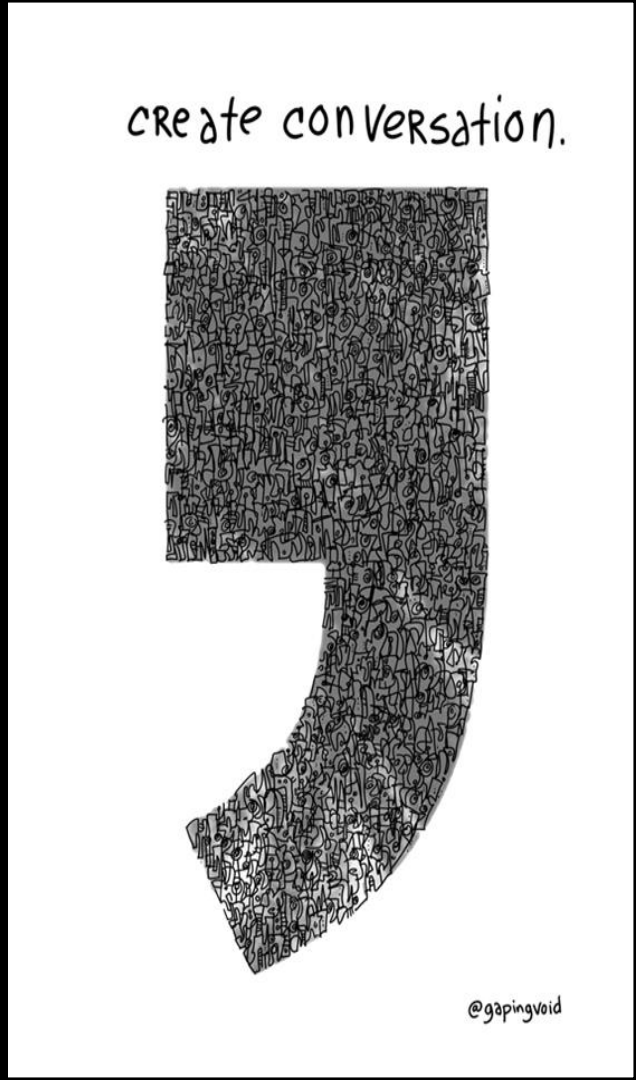
Cátedra de Ingeniería de Software

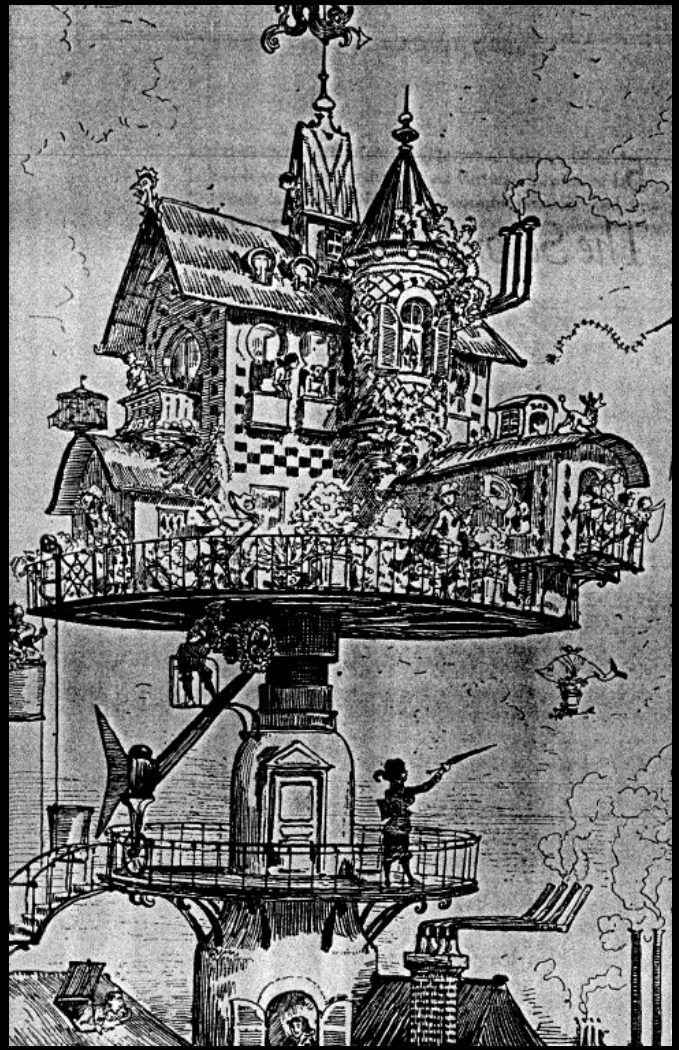
Docentes: Judith Meles – Laura Covaro

User Stories

"....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,

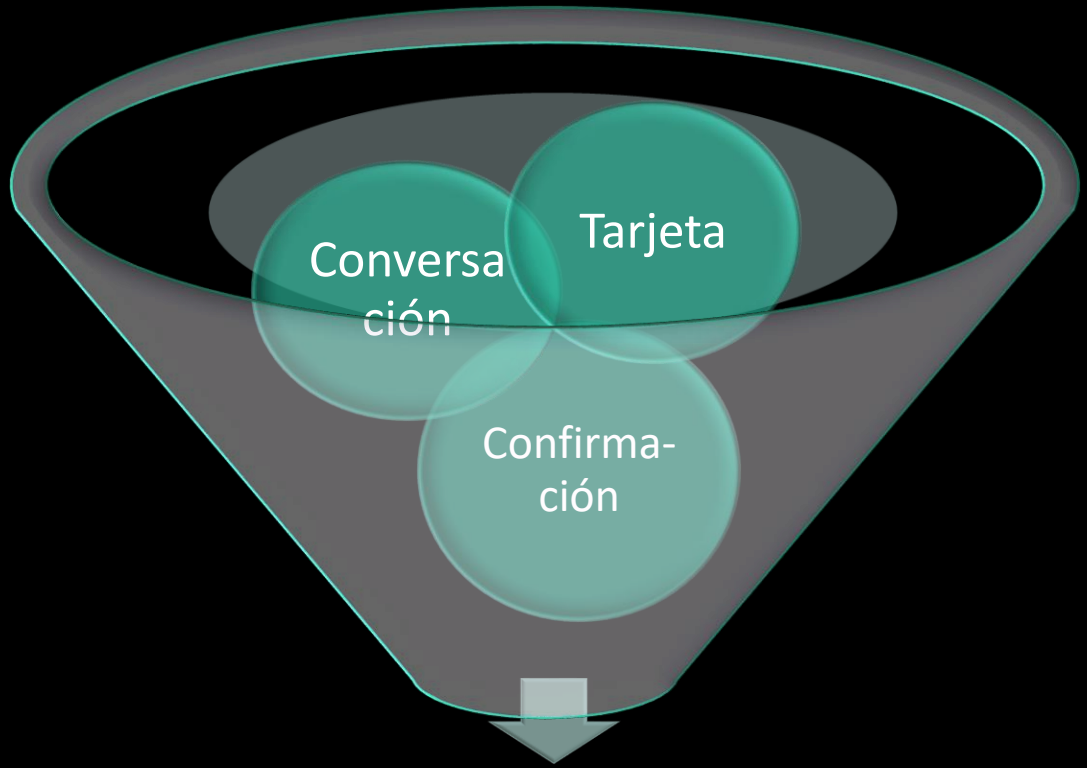




La parte más difícil de construir un sistema de software es decidir precisamente qué construir. Ninguna otra parte del trabajo conceptual es tan difícil como establecer los requerimientos técnicos detallados... Ninguna otra parte del trabajo afecta tanto el sistema resultante si se hace incorrectamente.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

¿Cuáles son las partes de una User Story?



Forma de expresar las Historias de Usuario

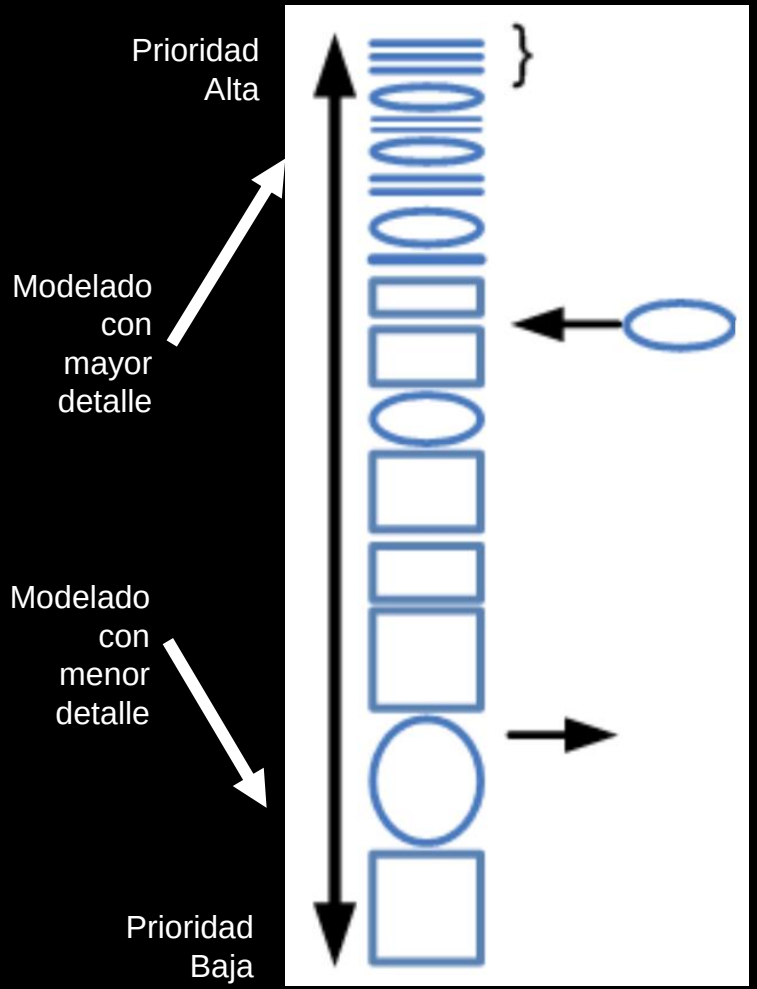
Como <nombre del rol>,
yo puedo <actividad>
de forma tal que <valor de negocio que recibo>

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

Representa la acción que realizará el sistema

Comunica porque es necesaria la actividad

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 Story: un ejemplo de tarjeta

Buscar Destino por Dirección

Como Conductor quiero buscar un destino a partir de una calle y altura para poder llegar al lugar deseado sin perderme.

Criterios de Aceptación:

- **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 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

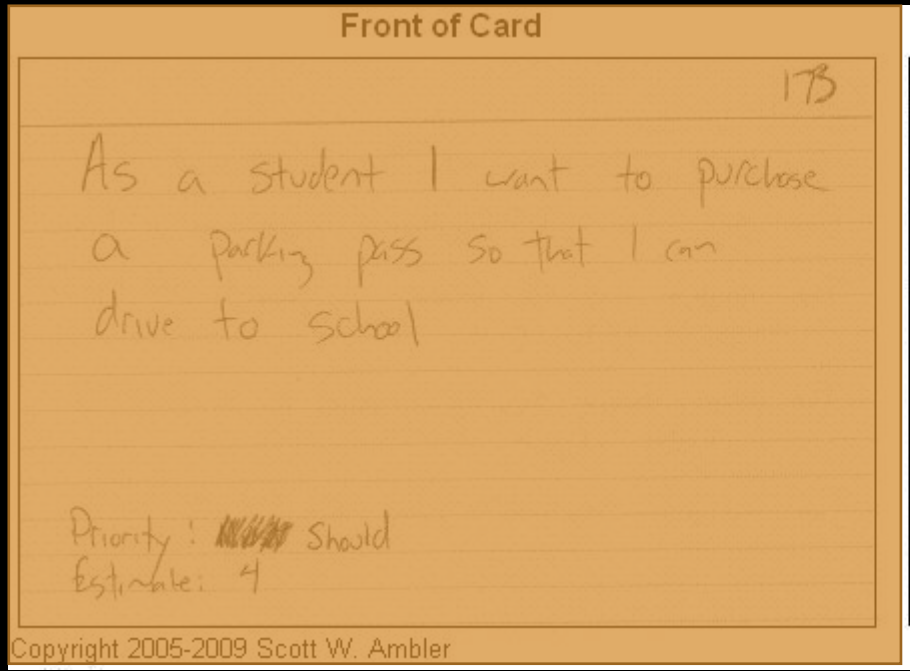
¿Cuáles son los Criterios de Aceptación buenos?

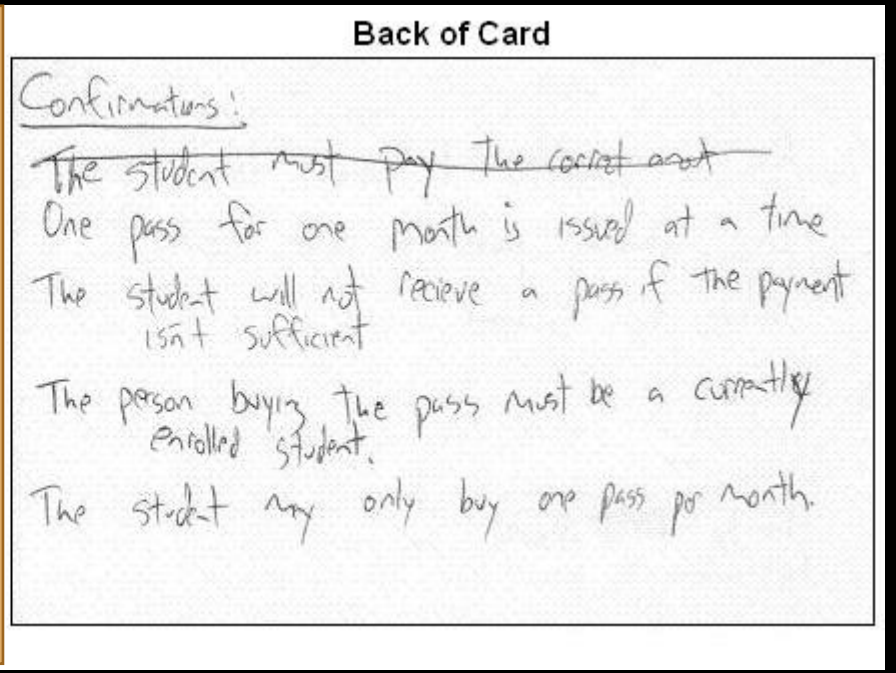
- Definen una intención, no una solución
  - Ej.: El usuario debe elegir al menos una cuenta para operar
- Son independientes de la implementación
- Relativamente de alto nivel, no es necesario que se escriba cada detalle

¿Y los detalles? ¿Dónde van?

- 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





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

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

Defínción de listo – Definition of Ready

Definición de Hecho – Definition of Done

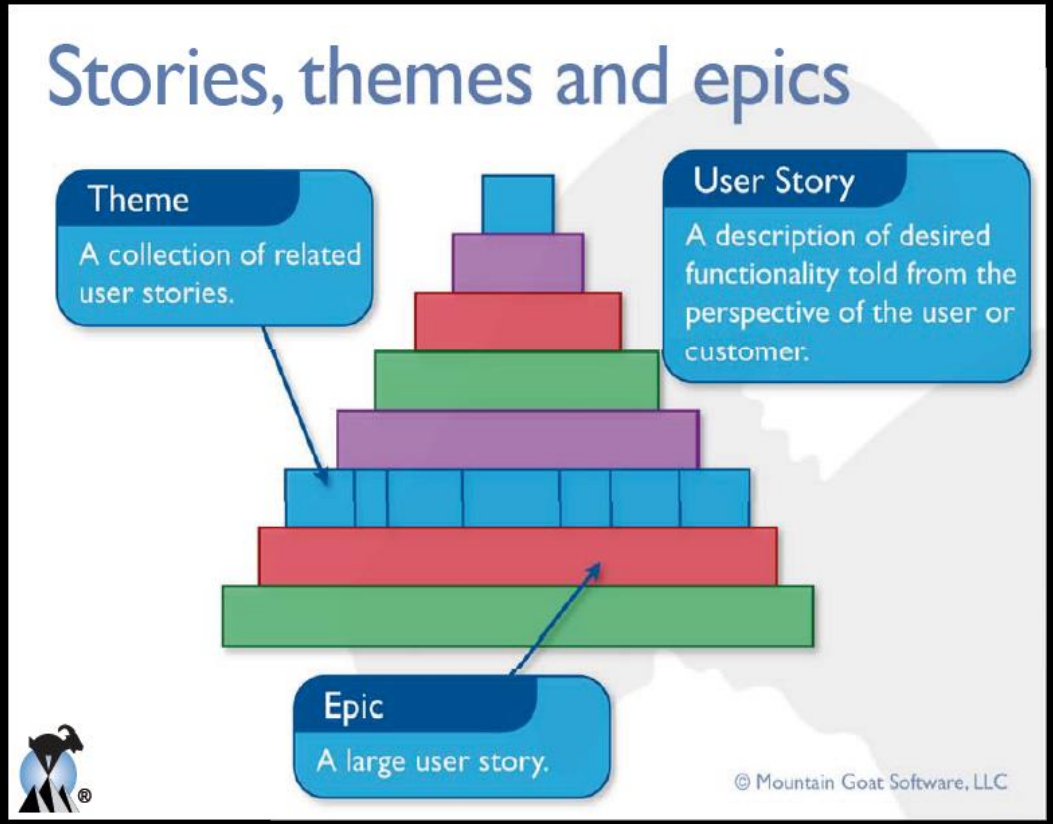
* “INVEST in Stories” – Bill Wake

http://xp123.com/xplor/xp0308/index.shtml

*INVEST Model

- **Independent** – calendarizables e implementables en cualquier orden
- **Negotiable** – el “qué” no el “cómo”
- **Valuable** – debe tener valor para el cliente
- **Estimatable** – para ayudar al cliente a armar un ranking basado en costos
- **Small** – deben ser “consumidas” en una iteración
- **Testable** – demostrar que fueron implementadas

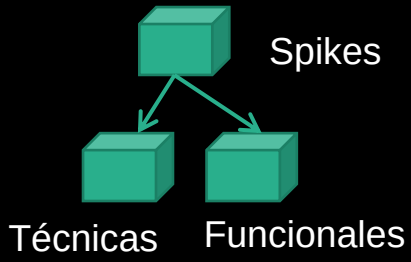
Diferentes niveles de abstracción



Spikes

- 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



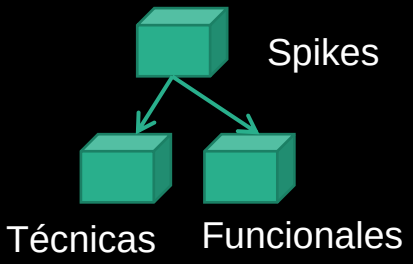
Técnicas

- 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.

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.

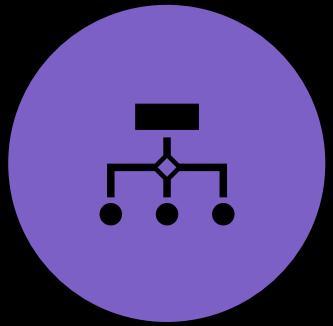
Implementar la spike en una iteración separada de las historias resultantes

- 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.

Algunas cosas para dejar en claro

DIFERIR EL ANÁLISIS DETALLADO TAN TARDE COMO SEA POSIBLE, LO QUE ES JUSTO ANTES DE QUE EL TRABAJO COMIENCE.

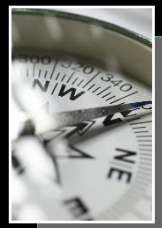
HASTA ENTONCES, SE CAPTURAN REQUERIMIENTOS EN LA FORMA DE “USER STORIES”.



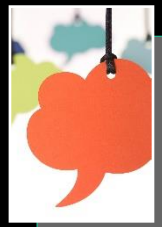
LAS USER STORIES NO SON REQUERIMIENTOS, NO NECESITAN SER DESCRIPCIONES EXHAUSTIVAS DE LA FUNCIONALIDAD DEL SISTEMA.

Tips para que las user stories sean útiles para el equipo

Un paso a la vez (evitar la palabra “Y”)



Usar palabras claras en los criterios de aceptación



No olvides la parte invisible: la conversación

Las user stories se escriben desde la perspectiva del usuario

No forzar todo para escribirlo como user stories

Material Bibliográfico de Referencia

- Libro:
  - Cohn, Mike - USER STORIES APPLIED – Editorial Addison Wesley 2004- Capítulos 1, 2 y 6
- Papers
  - Dean Leffingwell and Pete Behrens – A user story primer (2009)
- Link
  - http://www.mountaingoatsoftware.com/