fundación esplai
ciudadanía comprometida

C1 - Introducción a la Programación y al Diseño de Software

4 - Metodologías de Desarrollo

4.1 - Panorama de las metodologías Ágil

fundaciónesplai
ciudadanía comprometida

Proceso ‘clásico’ de desarrollo de software

Ese proceso consta habitualmente de las siguientes fases:

1. Captura de requisitos y conceptualización
   >> Entender lo que se busca
2. Análisis y Descripción funcional
   >> Listar características y funciones disponibles
3. Diseño >> Arquitectura, i.e. cómo vamos a organizar los elementos
4. Codificación >> La programación y ejecución de los trabajos
5. Pruebas >> Eliminación de fallos y comprobaciones de calidad
6. Distribución >> Instalación en el sistema del cliente

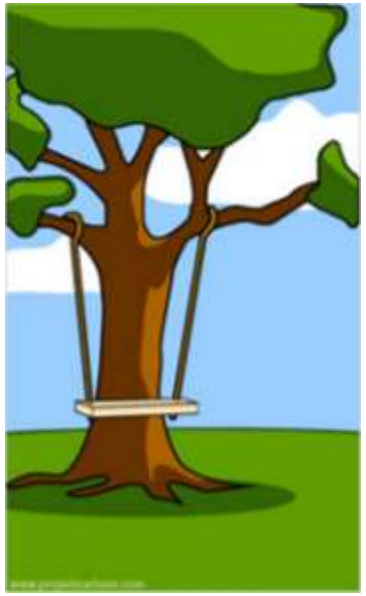
¿Esto se corresponde con la realidad de un proyecto?

Toma de requisitos

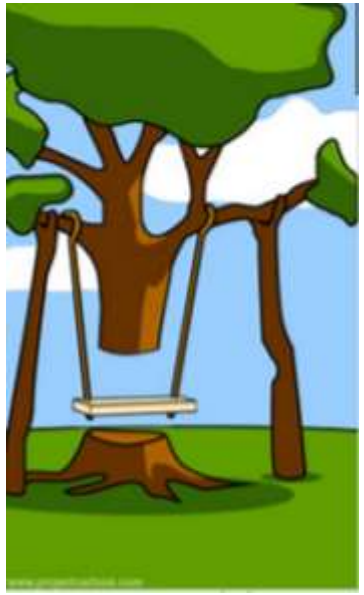
Como el usuario ha expresado sus necesidades

La descripción del comercial

Ejecución



Como el jefe del proyecto lo entendió

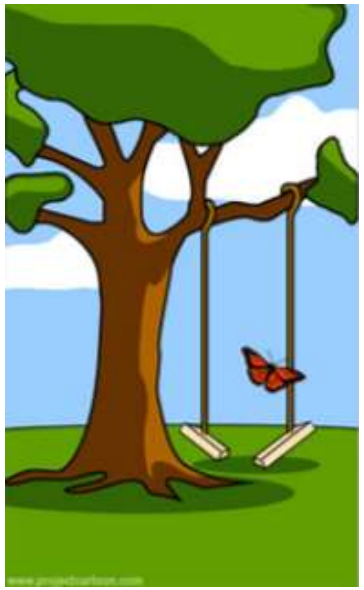


Como ha sido diseñado

Como ha sido desarrollado

Resultado producido

Lo que los usuarios han recibido

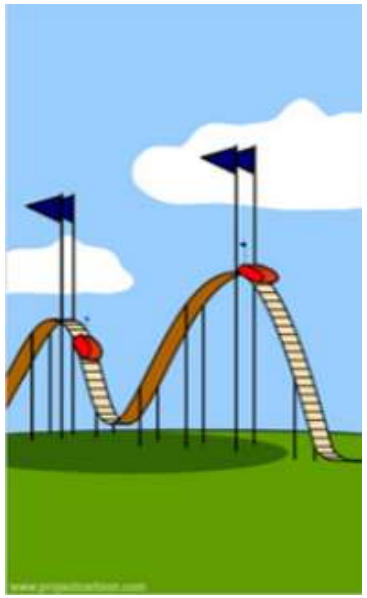


Como se ha comportado a la puesta en marcha

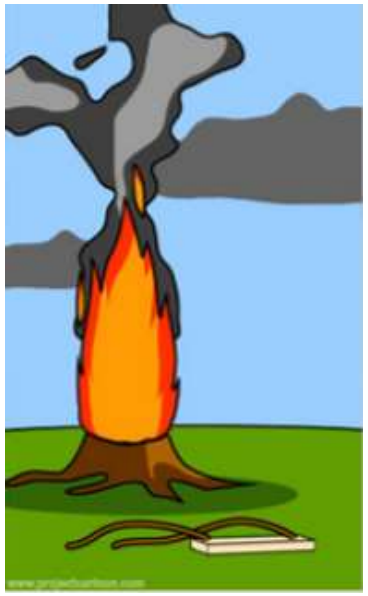
Como se aportaron las correcciones

Gestión del proyecto

Cuando ha sido entregado

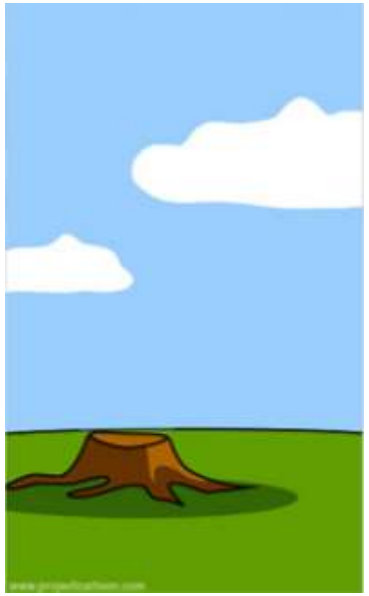


El presupuesto del proyecto

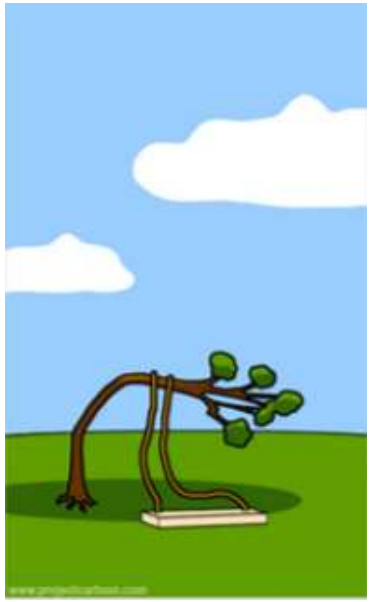


El impacto sobre la organización

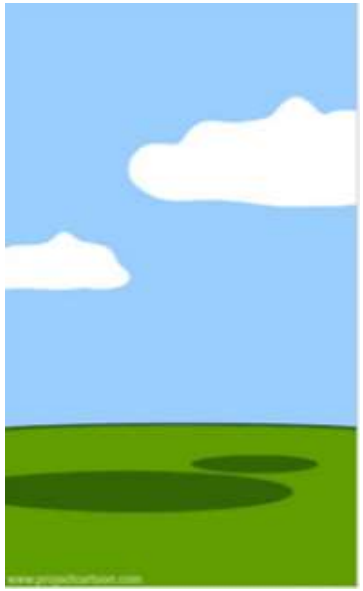
Post-proyecto



La organización del soporte técnico



El plan de contingencias



La documentación del proyecto

Y al final...

¡Lo que el usuario realmente necesitaba!

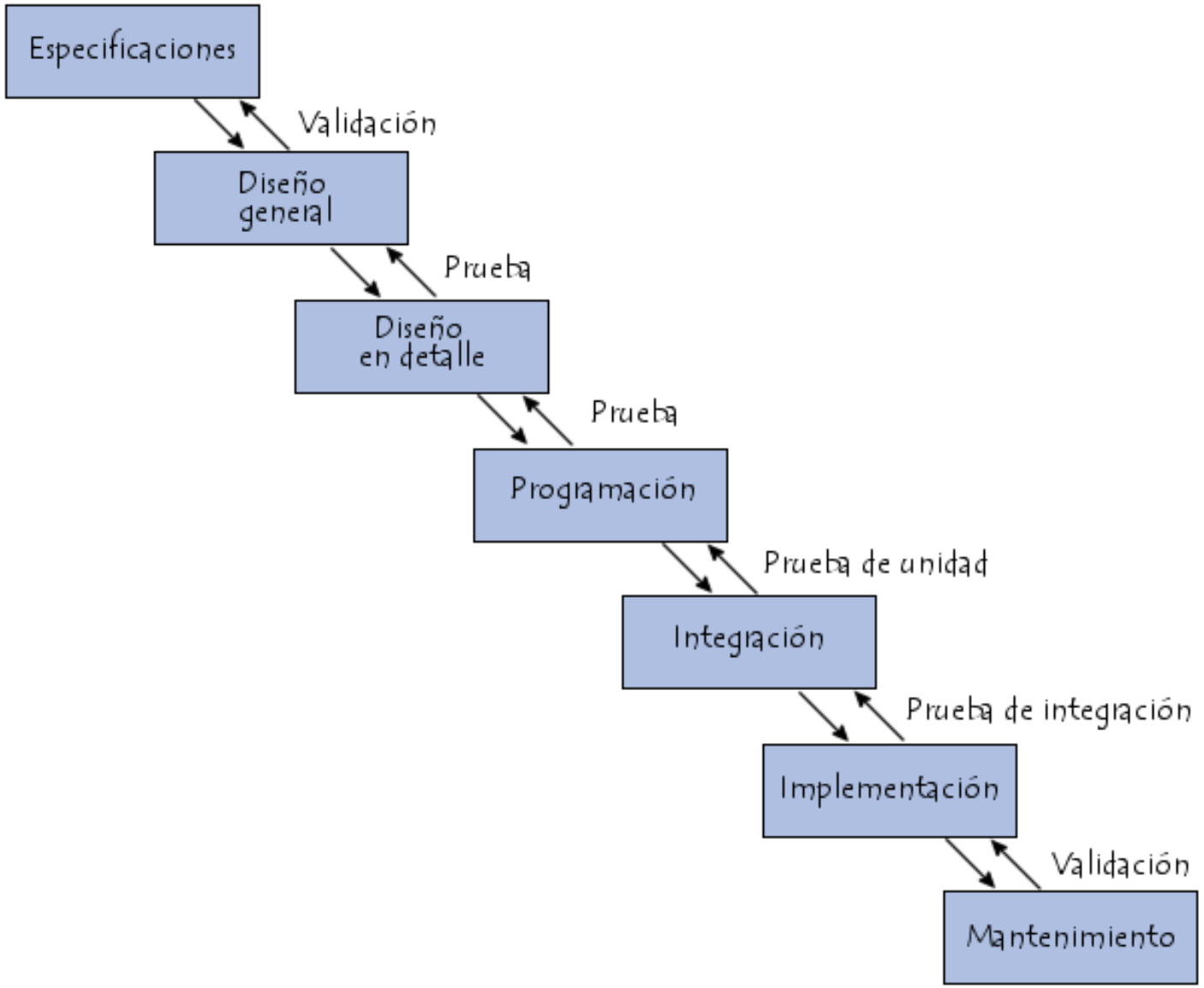
- La comunicación y la transparencia son clave en todo el proceso

- Todas las metodologías Agil hacen hincapíe en estos aspectos

Modelo en cascada

Es el módelo más ampliamente seguido por las organizaciones: se estima que el 90% de los sistemas han sido desarrollados así.

Se hace llamar también “Waterfall”



Características

- Este modelo admite la posibilidad de hacer iteraciones
  - Durante el mantenimiento se puede ver la necesidad de cambiar algo en el diseño.
  - Eso significa que se harán los cambios necesarios en la codificación y se tendrán que realizar de nuevo las pruebas.
  - Es decir, si se tiene que volver a una de las etapas anteriores al mantenimiento hay que recorrer de nuevo el resto de las etapas.
- Después de cada etapa se realiza una revisión para comprobar si se puede pasar a la siguiente
- Trabaja en base a documentos: la entrada y la salida de cada fase es documentación
- Idealmente, cada fase podría hacerla un equipo diferente gracias a la documentación generada entre las fases.

Ventajas

- La planificación es sencilla
- La calidad del producto resultante es alta
- Permite trabajar con personal poco cualificado, ya que la programación se hace en base a los documentos descriptivos

Inconvenientes

- Hay que tener todos los requisitos al principio. Y lo normal es que el cliente no tenga perfectamente definidas las especificaciones del sistema, o que surjan necesidades imprevistas.
- Si se han cometido errores en una fase es difícil volver atrás.
- No se tiene el producto hasta el final, esto quiere decir que:
  - Si se comete un error en la fase de análisis no lo descubrimos hasta la entrega, con el consiguiente gasto inútil de recursos.
  - El cliente no verá resultados hasta el final, con lo que puede impacientarse .
  - Es comparativamente más lento que los demás y el coste es mayor también.

fundaciónesplai
ciudadanía comprometida

Tipos de proyectos para los que es adecuado

- Proyectos para los que se dispone de todas las especificaciones desde el principio
  - Proyectos de reingeniería: ya tengo un sistema viejo que hace X, con lo cual puede fácilmente comprobar las funciones que realiza para reproducirlas
- Proyecto que no es novedoso (como un énisimo programa de contabilidad)
- Proyecto simple que se entiende bien desde el principio

¿El proyecto ideal?

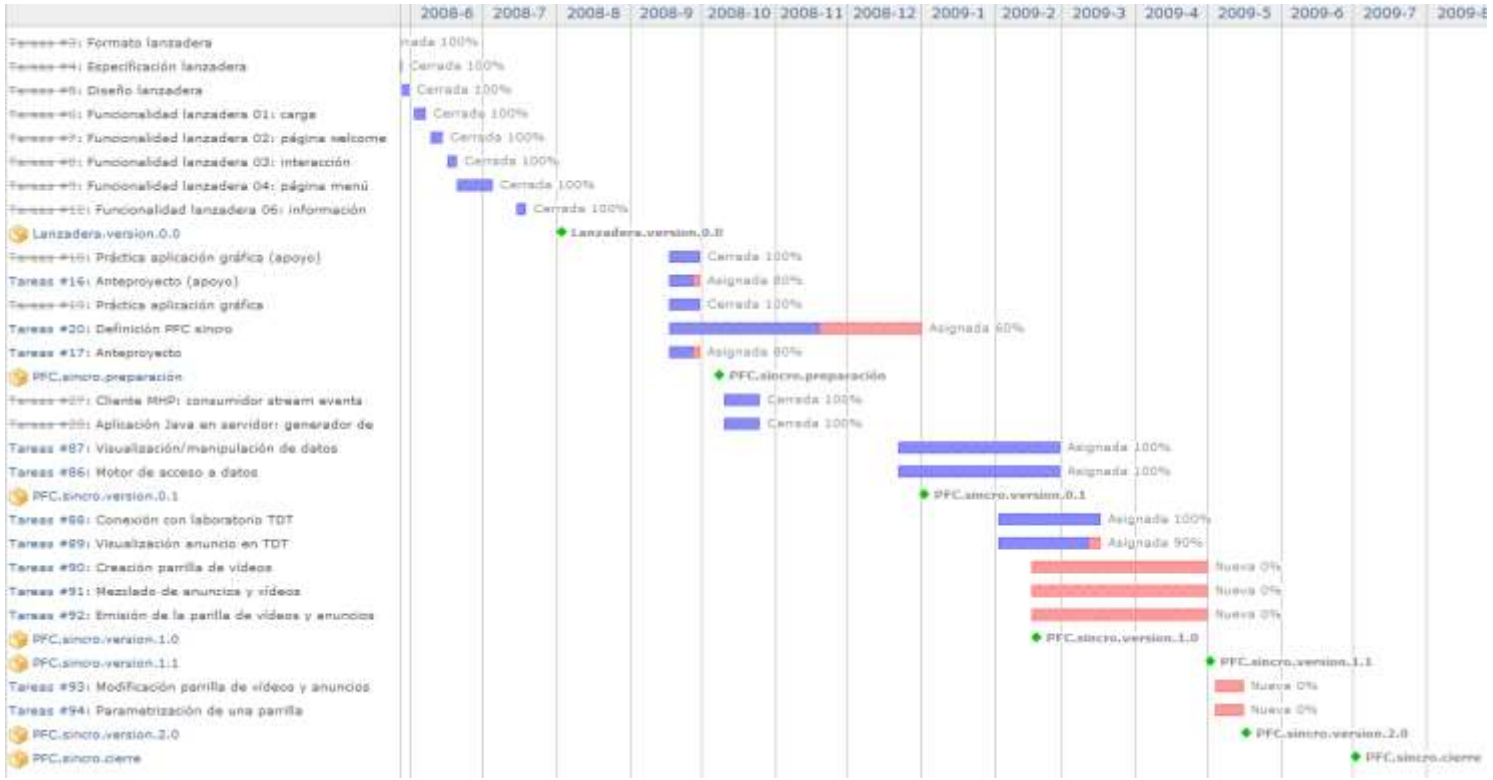


Diagrama de Gantt, muy frecuente para planificar proyectos con sus tareas y en qué orden se tienen que ejecutar

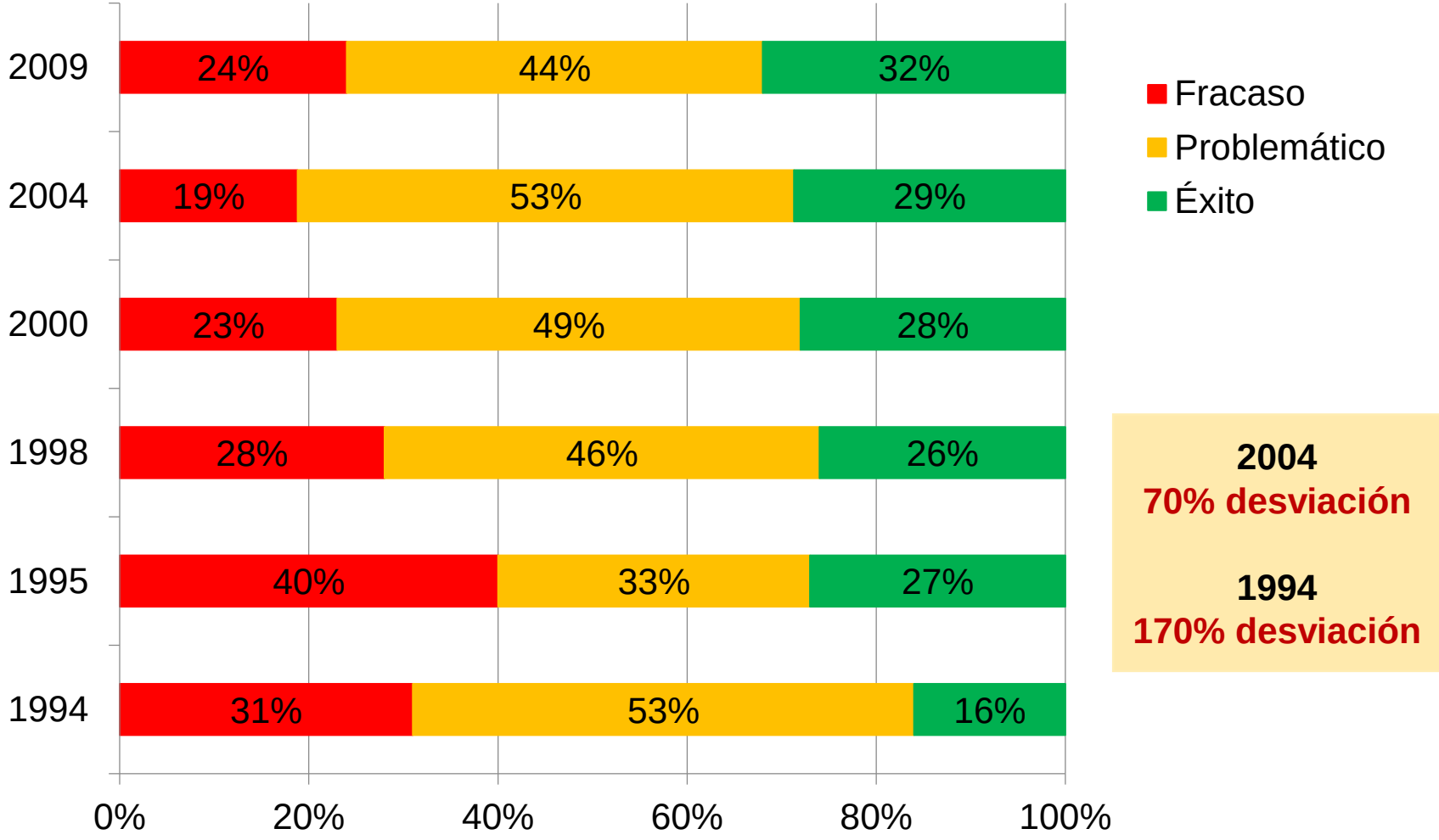
Pocas veces…

A menudo no se consigue respetar los plazos o la organización por los cambios que hay que gestionar

Éxito y fracaso de proyectos

‘Éxito’ significa que un proyecto se ha acabado en los plazos y con el coste previsto

La situación ha ido mejorando, gracias a la incorporación de nuevas metodologías



2004
70% desviación

1994
170% desviación

fundaciónesplai
ciudadanía comprometida

¿Por qué fracasan los proyectos de software?

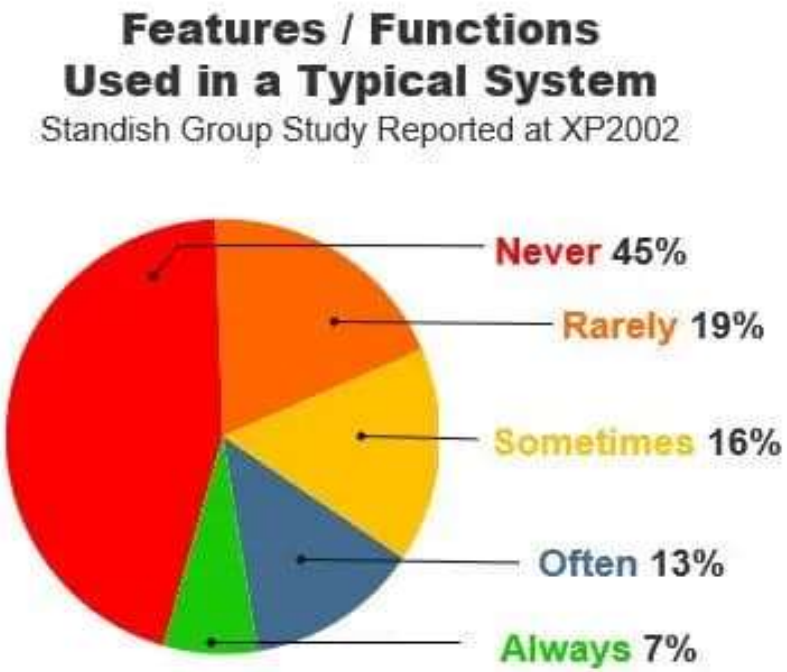
- Retrasos y desviaciones en la planificación
- Cancelación por coste o demora
- Defectos y fallos >> mala calidad, ejemplo de Lidl
- Requisitos mal comprendidos >> ejemplo del columpio…
- Cambios de negocio >> ejemplo de la cadena Blockbusters
- Falsa riqueza de características >> muchas funciones, pocas útiles
- Cambios de personal >> turnover = impacto en los equipos

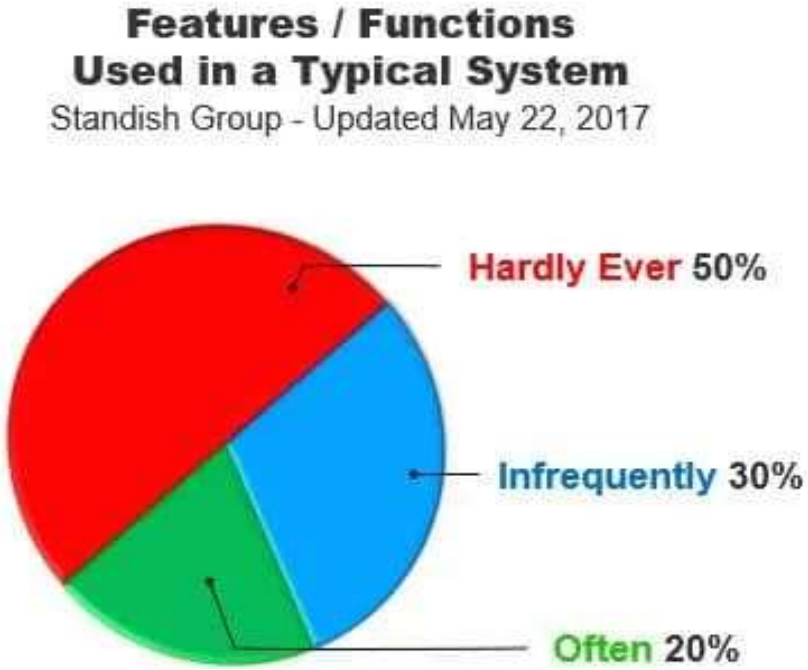
Falsa riqueza de características

Más de la mitad de las funciones desarrolladas sirven poco o núnca.

Esto significa que para cada hora útil, se ha dedicado otra hora en pura perdida.

El proyecto se hubiera podido realizar ¡en la mitad de tiempo y por un coste muy inferior!





Triangulo de Hierro

Existe una relación rígida entre:

- El Alcance de un proyecto (lo que incluye y abarca)
- Su Coste
- El Tiempo que se tardará en realizarlo

No se puede modificar un parámetro sin que afecte a los demás:

- ¿El cliente lo quiere antes? Será más caro, o habrá que reducir el alcance
- ¿Lo quiere más barato? Habrá que reducir el alcance, o si es posible, alargar el tiempo (par aprovechar otras oportunidades/huecos)
- ¿Quiere que incluya más cosas? Será más caro Y tardará más

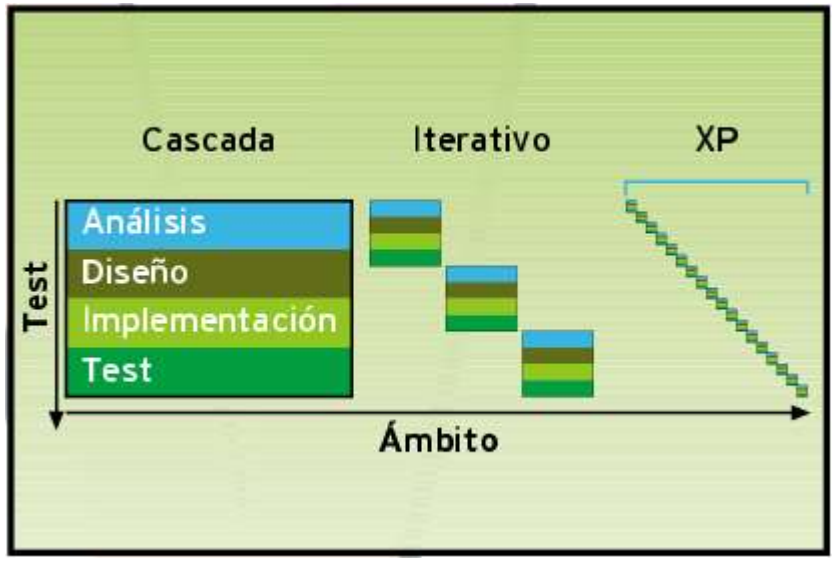
- Todo esto a calidad igual. Si bajamos la calidad, posiblemente podamos rebajar algo los demás criterios también.

¿Como soluciona Agile estos problemas?

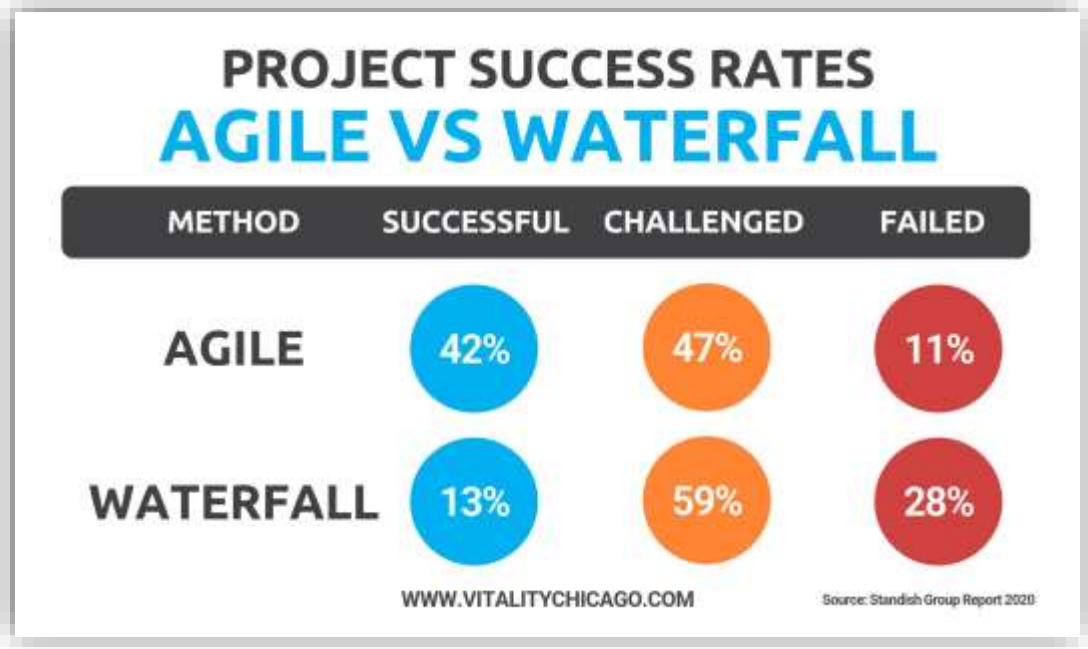
Usando iteraciones, con entrega al final de cada iteración.

Concretamente, así solucionamos los diversos problemas:

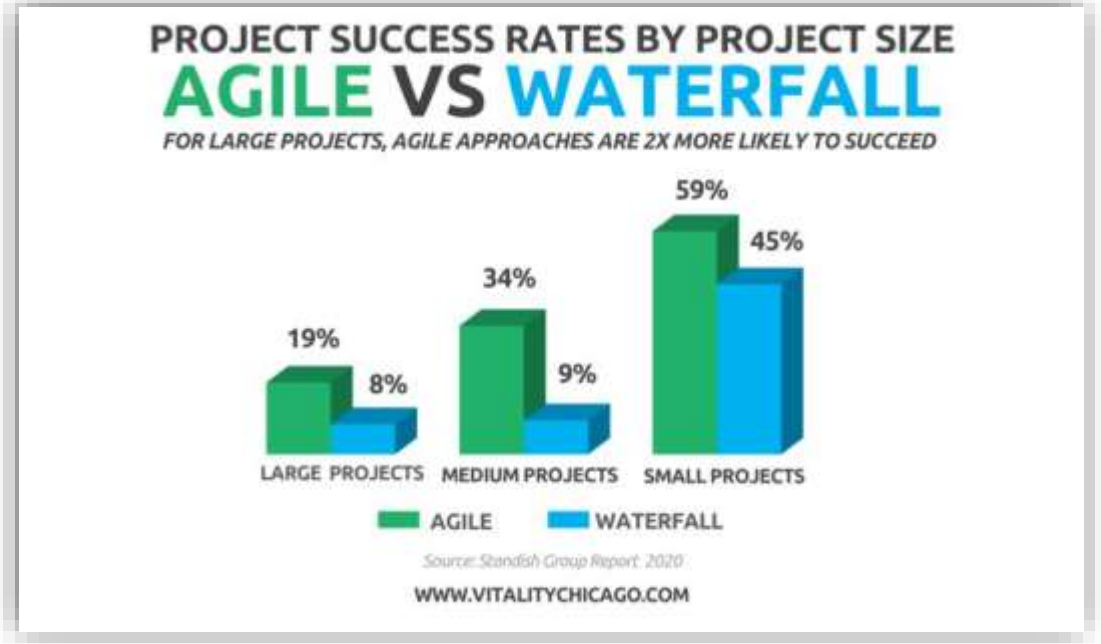
- Retrasos y desviaciones: ciclos cortos
- Cancelan el proyecto: entregas periódicas
- Defectos y fallos: pruebas continuas
- Requisitos mal comprendidos: meter el cliente dentro del equipo
- Cambios de negocio: versiones cortas
- Falsa riqueza de características: priorizar las tareas
- Cambios de personal: comunicación e integración



¿Porqué pasar a la agilidad ?



Según este estudio de 2020, las tasas de éxito de los proyectos ágiles en comparación a los proyectos clásicos, hablan por si solas…



Beneficios del cambio

En el estudio anual realizado en 2015 por VersionOne, se pueden ver los aspectos que sufren una mejora a consecuencia de la adopción de un método ágil.

Las mejoras son considerables, entre el 60% y 90%, mientras que los aspectos que habrían empeorado adoptando la agilidad son marginales, con una tasa de rechazo del 1% al 3%.

BENEFIT	GOT BETTER	NO CHANGE	GOT WORSE	DON'T KNOW
Ability to manage changing priorities	87%	3%	1%	9%
Increased team productivity	85%	3%	1%	11%
Improved project visibility	84%	3%	1%	12%
Increased team morale/motivation	81%	5%	3%	11%
Better delivery predictability	81%	6%	2%	11%
Faster time to market	80%	7%	1%	13%
Enhanced software quality	79%	6%	2%	14%
Reduced project risk	78%	6%	1%	15%
Improved business/IT alignment	77%	6%	1%	16%
Improved engineering discipline	73%	7%	2%	19%
Enhanced software maintainability	70%	8%	2%	21%
Better manage distributed teams	62%	11%	2%	25%

*Respondents were able to make multiple selections.


backlog
team
sprint
Scrum
product
items
Sprint
complete
project
Product
work
owner
framework
Items
artifacts
teams
meets
value
roles
completed
prioritized
priority
set
processes
broken
one
three
struggles
chart
Three
ceremonies
progress
tasks
scrum
share
look
values
retrospectives
productive
understand
time
terminology
ways
dynamic
right
form
done
refined
agile
Team
deliver
Manifesto
negotiation
planning
choose
Successful
burndown
embrace
comprehensive
day

Agilidad, un marco de trabajo

Gestión de proyectos (complejos, sujetos al cambio)

Métodos empíricos (experiencia)

Iteración, Adaptación, Entrega

Agilidad, un marco de trabajo

- La agilidad suministra un Marco de trabajo para la gestión de proyectos, esencialmente para la concepción de softwares, pero no únicamente (también se usa en indústria)
- Es un marco particularmente adaptado a la gestión de proyectos complejos y sujetos al cambio: es el caso a menudo en desarrollo de software
- Los métodos ágiles se basan en el empirismo, es decir, son el fruto de la experiencia

El MARCO de trabajo ofrecido por los métodos ágiles es:

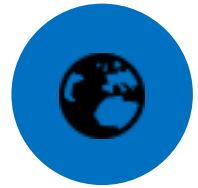
- **Iterativo**: se basa en ciclos cortos,
- **Adaptativo**: preparado para antender las necesidades de cambio y adaptación,
- **Incremental**: entrega regular, durante todo el proyecto, de versiones funcionables y utilizables de todo o una parte del software (los llamados incrementos)

fundación esplai
ciudadanía comprometida

Manifiesto Ágil

Años 80

2001



agilemanifesto.org

4

valores centrales

12

mandamientos

4 valores centrales (prioridades)

Individuos e interacciones, más que procesos y herramientas

Software operacional, más que documentación exhaustiva

Colaboración con los clientes, más que negociación contractual

Adaptación al cambio, más que seguimiento de un plan

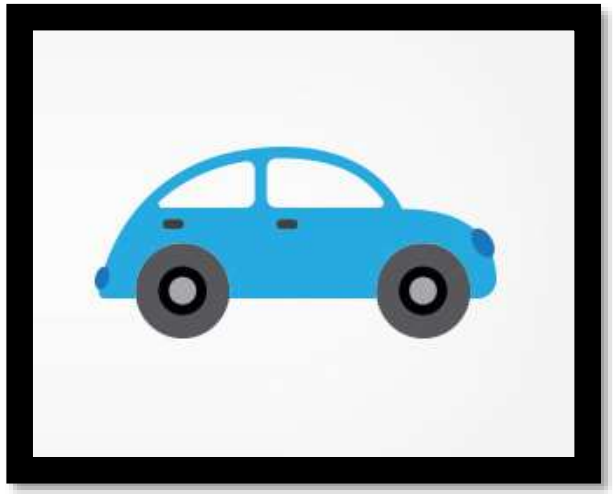
Iterativo, adaptativo, incremental

Proyecto: Crear un vehículo
Para ayudar al cliente en sus desplazamientos, comprar, llevar niños al colegios, etc.

En un proyecto clásico (en cascada) las fases podrían ser:

- Diseño del vehículo,
- Creación del chasis, de la carrocería, actualización del motor,
- Concepción del interior,
- Ensamblaje final,
- Prueba en carretera,
- Entrega al cliente

Durante el desarrollo el cliente sólo habría visto dibujos y maquetas.



Algunas adaptaciones necesarias serian detectadas únicamente al final:

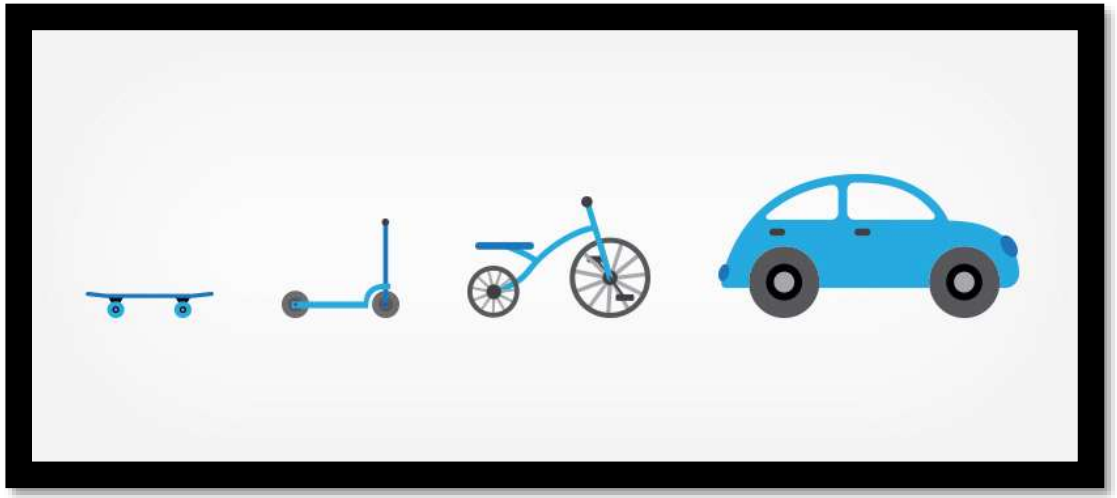
- Un color erroneo porque el cliente es daltónico,
- Asientos inadaptados debido a la medida de sus piernas…

Iterativo, adaptativo, incremental

Proyecto de creación de un vehículo (cont.)

En Agile las fases podrían ser:

- Crear un skateboard: ofrece parte de la solución, al permitir al cliente desplazarse más rápido
- Luego, añadir un manillar, y convertirlo en patinete: más estabilidad, y posibilidad de colgar una bolsa.
- Añadir un sillín y ruedas más grande: más comodidad, posibilidad de llevar algún paquete



Y luego:

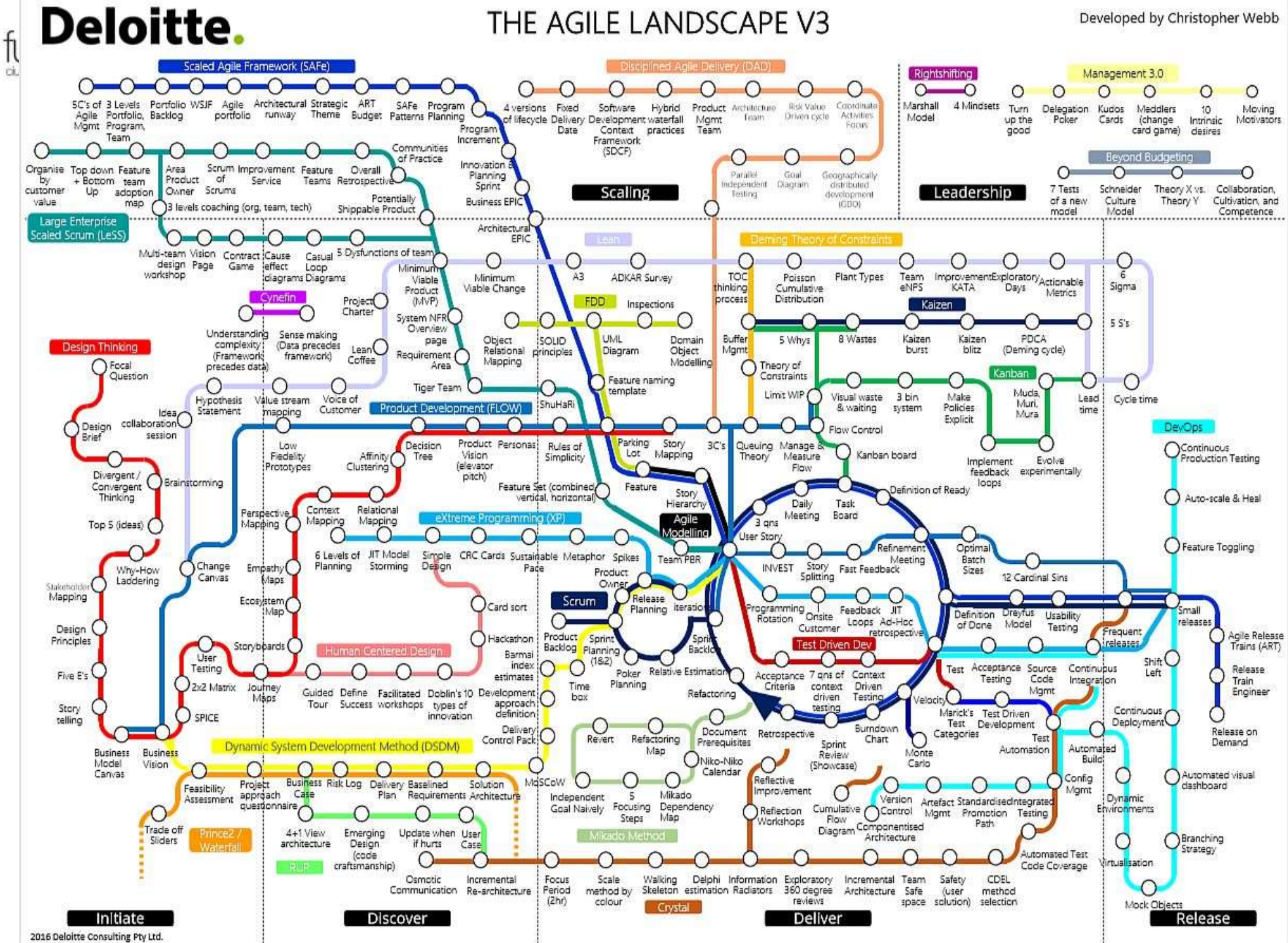
- Cambiar de 2 a 4 ruedas, con un motor,
- Añadir caroceria,
- Añadir más asientos….

Trabajando en modo Ágil, entregamos valor al cliente a cada etapa, y nos damos cuenta de los problemas mucho antes.

fundación esplai
ciudadanía comprometida

Metodologías ágiles

- Métodos o filosofías ?
- Herramientas con combinaciones múltiples
- Inspiración mutua


Deloitte.
THE AGILE LANDSCAPE V3
Developed by Christopher Webb
Scaled Agile Framework (SAFe)
Disciplined Agile Delivery (DAD)
Rightshifting
Management 3.0
Beyond Budgeting
Scaling
Leadership
Large Enterprise Scaled Scrum (LeSS)
Lean
Deming Theory of Constraints
Cynefin
Kaizen
Design Thinking
Product Development (FLOW)
Kanban
DevOps
eXtreme Programming (XP)
Agile Modelling
Scrum
Human Centered Design
Test Driven Dev
Dynamic System Development Method (DSDM)
Mikado Method
Prince2 / Waterfall
RUP
Crystal
Initiate
Discover
Deliver
Release
2016 Deloitte Consulting Pty Ltd.

Principios fundamentales

Transparencia

Inspección

Adaptación

fundaciónesplai
ciudadanía comprometida

Algunas metodologías Agile

- JIT : *Just In Time* – resolver problemas enseguida
- Lean : valor añadido, reducir las pérdidas
- Kanban : tarjeta de señalización, sistema visual, tareas en curso (*Work In Progress*)
- FDD : *Feature-Driven Development*
  Desarrollo basado en las funcionalidades
- UP : *Unified Process – Rationnal/UML*
  Basado en *uses cases*
- XP : eXtreme Programming
  Programación extrema


kanban
flow
tool
signal
management
model
alternative
project
manage
japan
trigger
evolutionary
leadership
delivery
principles
process
reminder
just
theories
limit
leaders
signaling
chain
Lean
roles
efficiency
tasklist
kaizen
explicit
inventory
control
jit
Manufacturing
transportation
plan
method
development
feedback
visualize
supply
improve
change
approach
production
scrum
software
collaborate
system
sheduling
board
task
japanese

Lean

- **Manufactura Lean**: este concepto abarca el producir más y más con menos esfuerzo humano, menos equipo, menos tiempo, menos espacio para así darles a los clientes lo que quieren exactamente: *Toyota / Taiichi Ohno*.

  Es una adaptación de producción en masa en donde los trabajadores y las áreas de trabajo son hechas mas flexibles y mas eficientes adoptando métodos que reducen los desperdicios.

“We get brilliant results from average people managing brilliant processes (…) our competitors (…) get average (…) results from brilliant people managing broken processes.”

Toyota

” …American car makers (…) admit that Toyota’s real advantage was its ability to harness the intellect of ‘ordinary’ employees.”

Harvard Business Review

Lean – **4 principios**

- La metodología Lean está basada en cuatro principios:
  - Minimizar desperdicios
  - Calidad a la primera
  - Sistema de producción flexible
  - Mejora continua


kanban
work
process
current
change
incremental
continuous
improve
flow
board
system
starting
implement
organization
understanding
evolutionary
people
practices
principles
progress
method
small
policies
organizations
respect
found
prioritization
forward
time
environmental
place
floor
feedback
disruptive
loops
visualize
reduce
implemented
flows
learning
product
explicit
areas
encourages
wip
expectations
focus
limits
make
great
evolve
benefit
learned
encourage
understand
approach
waste
flowing
move
environment

看板

kanban

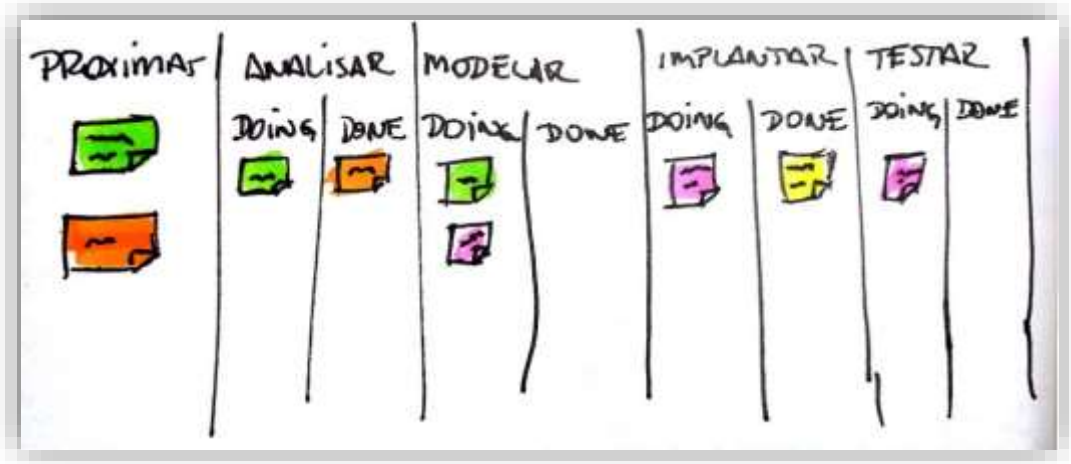
Kan = visual

Ban = tarjeta, tablero

(valla publicitaria, etiqueta…)

Kanban

- Panel de trabajo donde se dividen las tareas en tres o mas columnas con los pasos del proceso
- Implementación más sencilla: tarjetas que se pegan en un tablero que puede ser visualizado por todos los empleados. Las tarjetas actúan de testigo de los diferentes procesos.
- En la actualidad se utiliza para todo tipo de trabajos, desde fabricas industriales hasta oficinas.



Kanban - **Beneficios**

- Visualización del flujo de trabajo y sus limitaciones (WIP): permite balancear la demanda con la capacidad
- Ajuste de cada proceso y flujo de valor a medida
- Reglas simples que permiten optimizar el trabajo en función del tipo de tarea
- Mejor manejo del riesgo
- Mejora de la calidad del trabajo
- Permite identificar y resolver problemas, de forma colaborativa e inmediata

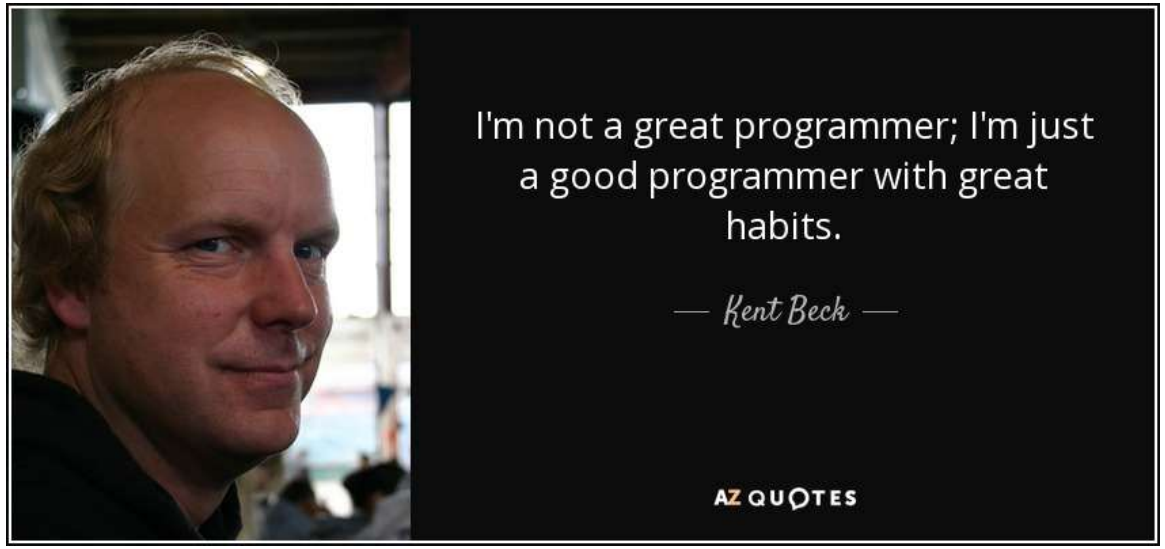
Kanban – **Resultados**

- Transparencia: permite visualizar los efectos de sus in/acciones
- Personas más inclinadas a colaborar cuando ven los procesos…
- …Incremento de colaboración y confianza entre todos los miembros del equipo
- Equipos auto-organizados
- Flujo similar de todos los procesos: ritmo de trabajo sostenido y sostenible


XP
consequences
estimates
practice
communication
programmers
development
constraints
tasks
skills
development
system
plan
people
make
features
relationships
values
edition
manage
Extreme Programming
productivity
changing
clear
time
technique
communication
release
lightweight
tasks
failure
calls
excellence
software
team
excellence
today
feel
practice
release
improve
technical
productive
manage
consequences
people
clear
today
control
calls
weeks
software
iteration
rapid
feedback
feedback
change
cycles
changing
protect
social
functionality
requirements
customer
early
success
goals
change
time
plan
relationships
technique
functionality
progress
project
productivity
features
members
values
style
project
priority
including
members
progress
success
technical
style
protect
edition
team
iteration
programmers
process
estimates
requirements
process
create
methodology
customer

XP - Definición

“Un proceso ligero, de bajo riesgo, flexible, predecible, científico y divertido de desarrollar software”



XP - eXtreme Programming

Es una metodología ágil:

- centrada en potenciar las relaciones interpersonales como clave para el éxito en desarrollo de software,
- Promoviendo el trabajo en equipo,
- Preocupándose por el aprendizaje de los desarrolladores,
- Propiciando un buen clima de trabajo.

fundaciónesplai
ciudadanía comprometida

XP – 12 prácticas

- **Equipo completo:** Forman parte del equipo todas las personas que tienen algo que ver con el proyecto, incluido el cliente y el responsable del proyecto.
- **Planificación:** Se hacen las historias de usuario y se planifica en qué orden se van a hacer y las mini-versiones. La planificación se revisa continuamente.
- **Test del cliente:** El cliente, con la ayuda de los desarrolladores, propone sus propias pruebas para validar las mini-versiones.
- **Versiones pequeñas:** Las mini-versiones deben ser lo suficientemente pequeñas como para poder hacer una cada pocas semanas. Deben ser versiones que ofrezcan algo útil al usuario final, y no trozos de código que no pueda ver funcionando.
- **Diseño simple:** Hacer siempre lo mínimo imprescindible de la forma más sencilla posible. Mantener siempre sencillo el código.
- **Pareja de programadores:** Los programadores trabajan por parejas (dos delante del mismo ordenador) y se intercambian las parejas con frecuencia (un cambio diario).

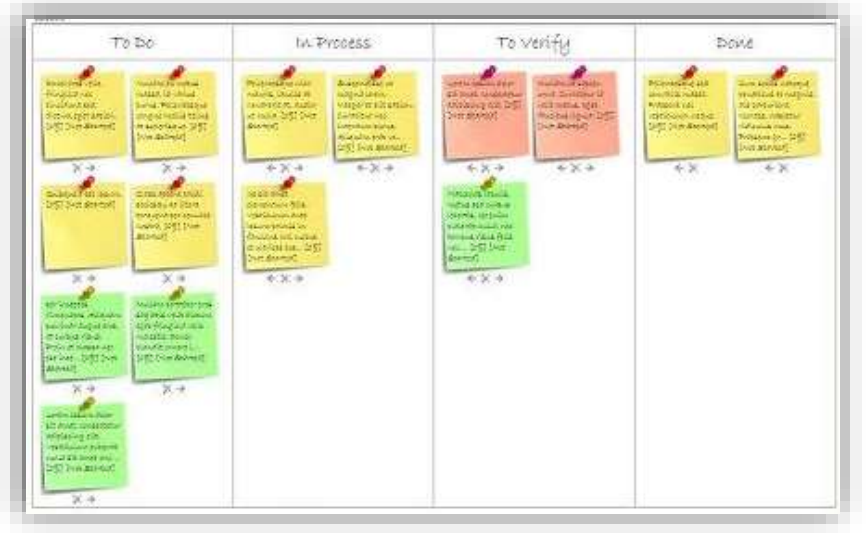
fundaciónesplai
ciudadanía comprometida

XP – 12 prácticas

- **Desarrollo guiado por pruebas (TDD):** Se deben realizar programas de prueba automática y deben ejecutarse con mucha frecuencia. Cuantas más pruebas, mejor.
- **Integración continua:** Deben tenerse siempre un ejecutable del proyecto que funcione y en cuanto se tenga una nueva pequeña funcionalidad, debe recompilarse y probarse. Es un error mantener una versión congelada dos meses mientras se hacen mejoras y luego integrarlas todas de golpe. Cuando falle algo, no se sabe qué es lo que falla de todo lo que hemos metido.
- **El código es de todos:** Cualquiera puede y debe tocar y conocer cualquier parte del código.
- **Normas de codificación:** Estilo común de codificación (no importa cual), de forma que parezca que ha sido realizado por una única persona.
- **Metáforas:** Hay que buscar unas frases o nombres que definan cómo funcionan las distintas partes del programa, de forma que sólo con los nombres se pueda uno hacer una idea de qué es lo que hace cada parte del programa: "recolector de basura" de java. Ayuda a que todos los programadores (y el cliente) sepan de qué estamos hablando y que no haya mal entendidos.
- **Ritmo sostenible:** Trabajar a un ritmo que se pueda mantener indefinidamente. No debe haber días muertos ni un exceso de horas otros días. Al tener claro semana a semana lo que debe hacerse, hay que trabajar duro en ello para conseguir el objetivo cercano de terminar una historia de usuario o mini-versión.

Iniciación a Scrum

El método ágil más usado



Equipo Scrum

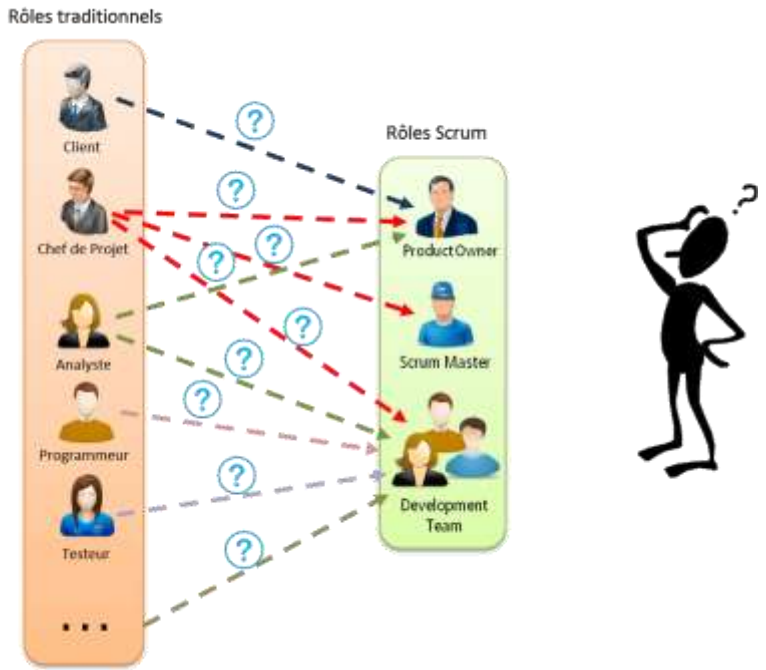
- Product Owner
- Scrum Master
- Equipo de desarrollo : 3 a 9

Equipo Scrum

Dev Team :

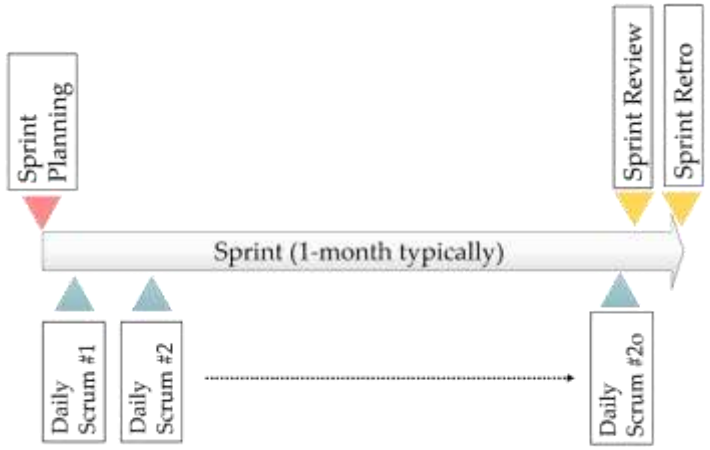
- Auto-organizado : sin jefe de proyecto
- Multifuncional : con todas las competencias en interno

Roles tradicionales



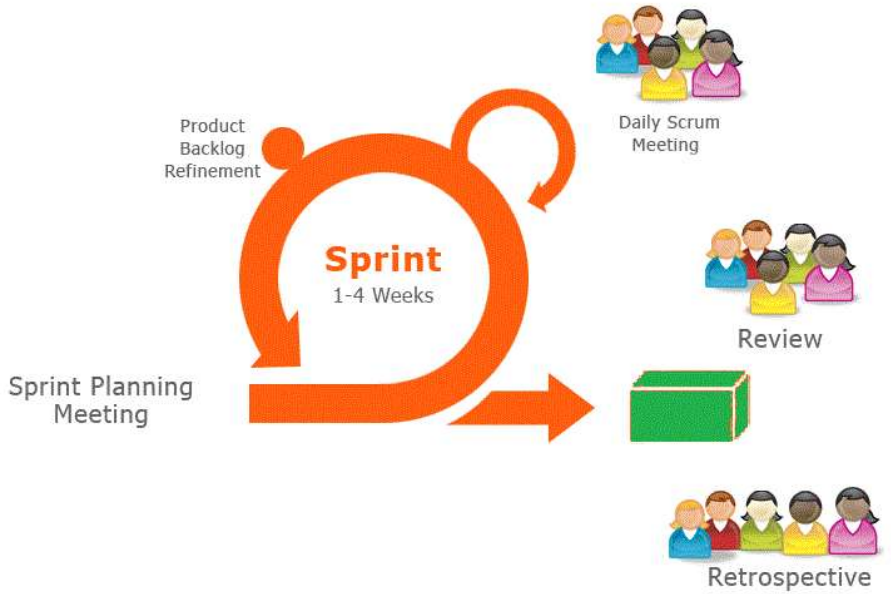
Eventos

- Para crear regularidad
- Reducir la necesidad de reuniones no definidas
- Time-box : duración limitada



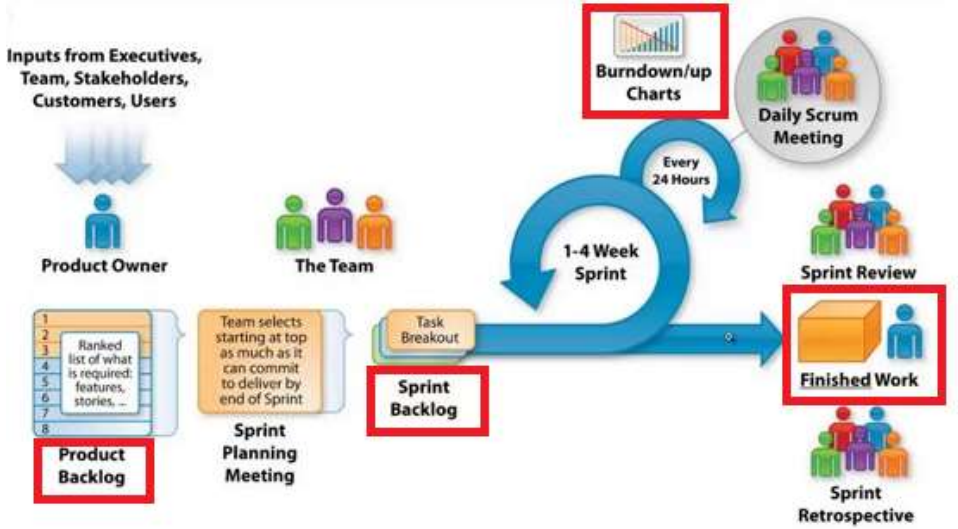
Eventos

- Sprint
- Sprint planning
- Daily Scrum
- Sprint Review
- Sprint Retrospective



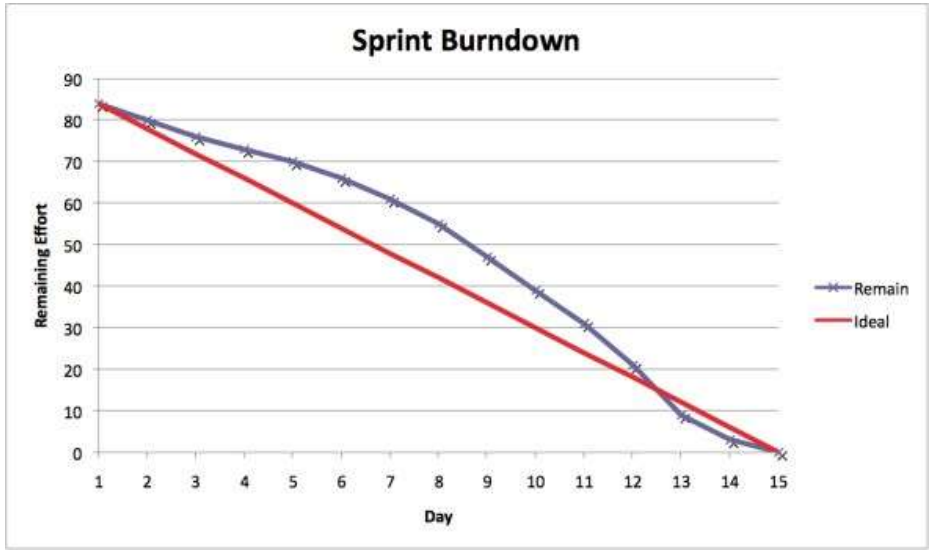
Artefactos

- Product Backlog
- Sprint Backlog
- Product increment



Artefactos

- Burndown chart
- Impediment list



fundación esplai
ciudadanía comprometida

Iniciación a Scrum

Equipo Scrum : sus miembros y sus respectivos roles

El Product Owner

- El cliente : propietario del producto, encargado de maximizar su valor y el del trabajo del equipo de desarrollo.

- Técnico + Business
- Orientado al cliente

El equipo de desarrollo

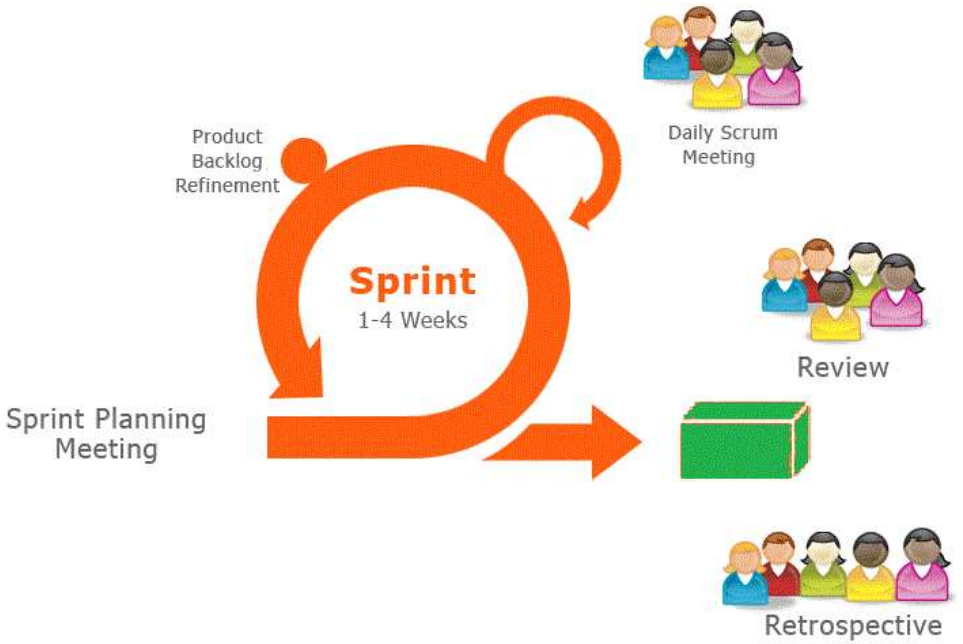
- Auto-organizado
- 3 a 9 miembros
- Pluridisciplinario
- Responsabilidad global del equipo de desarrollo
- Sin sub-equipos dedicados a un dominio en particular

fundación esplai
ciudadanía comprometida

Scrum Master

- Facilitador – Coach Ágil
- Ayuda el equipo
  - a ponerse de acuerdo sobre un objetivo accesible, en un lapso de tiempo determinado (sprint)
  - … durante la reunión diaria
  - a estar concentrado y seguir las reglas adoptadas
- Elimina los obstáculos que impiden al equipo avanzar
- Evita al equipo las distracciones exteriores

Sprints


Daily Scrum
Meeting
Product
Backlog
Refinement
Sprint
1-4 Weeks
Review
Sprint Planning
Meeting
Retrospective

fundación esplai
ciudadanía comprometida

Sprint Planning

- 8h para un Sprint de un mes
- SM + DT + PO

- 2 fases :
  - Qué >> Sprint Backlog
  - Cómo >> Detalles de las tareas, estimación, asignación, arquitectura

Daily Scrum

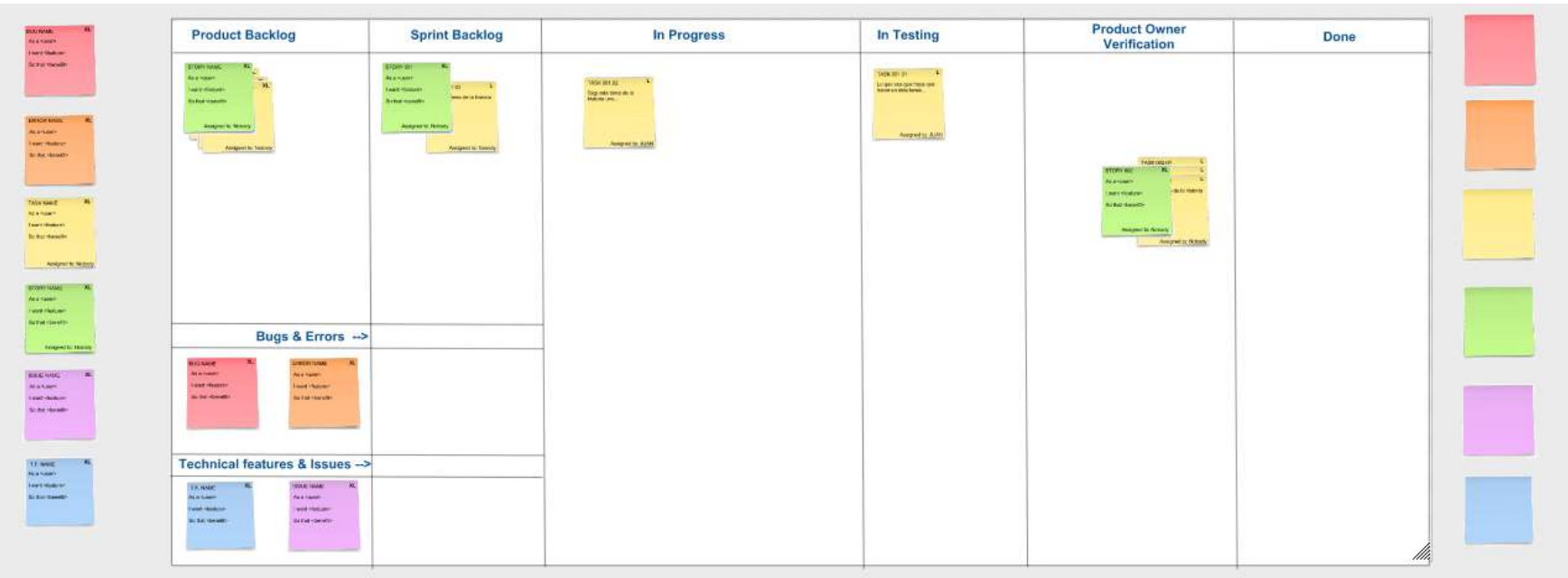
- La melé… un trabajo en equipo
- SM + DT + PO
- 15 minutos, de pie

- ¿Qué se ha hecho?
- ¿Qué se hará?

- Identificar los obstáculos
- Concentrarse sobre el objetivo del sprint
- Comunicar objetivamente sobre el avance

fundación esplai
ciudadanía comprometida

Scrum Board



Sprint Review

- 4h por un sprint de un mes
- SM + DT + PO + invitados
- Objectivo : validar la entrega

- Demostración
- Validación de las funcionalidades
- Cálculo de la velocidad
- Ordenación del PB
- Re-planificación de la entrega final

fundación esplai
ciudadanía comprometida

Sprint Retrospective

- 3h para un sprint de un mes
- SM + DT + PO
- Objetivo: mejorar la forma de hacer

- Evaluación de los modos de funcionamiento usados
- Identificación de las posibles mejoras
- Elección de las mejoras a aplicar al Sprint siguiente

fundaciónesplai
ciudadanía comprometida

Definición de ‘Acabado’ (Done)

Por ejemplo:

- Codificado siguiendo un estándar preestablecido
- Testado unitariamente
- Tests reales pasados con éxito
- Versión disponible en varios idiomas
- Documentación del usuario redactada

Pila de producto (Product Backlog)

- Lista de elementos
- Características del producto
- Orientada al cliente
- Priorizada
- Valor oficio

fundación esplai
ciudadanía comprometida

DEEP

- El Backlog debe estar :
  - **D**etallada de forma apropiada para ser entendida por los desarrolladores
  - **E**stimado, tarea por tarea (o por user story)
  - **E**mergente (o **E**volutivo) : actualizado a cada Sprint
  - **P**riorizado : los desarrolladores se ocuparan primero de los casos que tengan más importancia

fundación esplai
ciudadanía comprometida

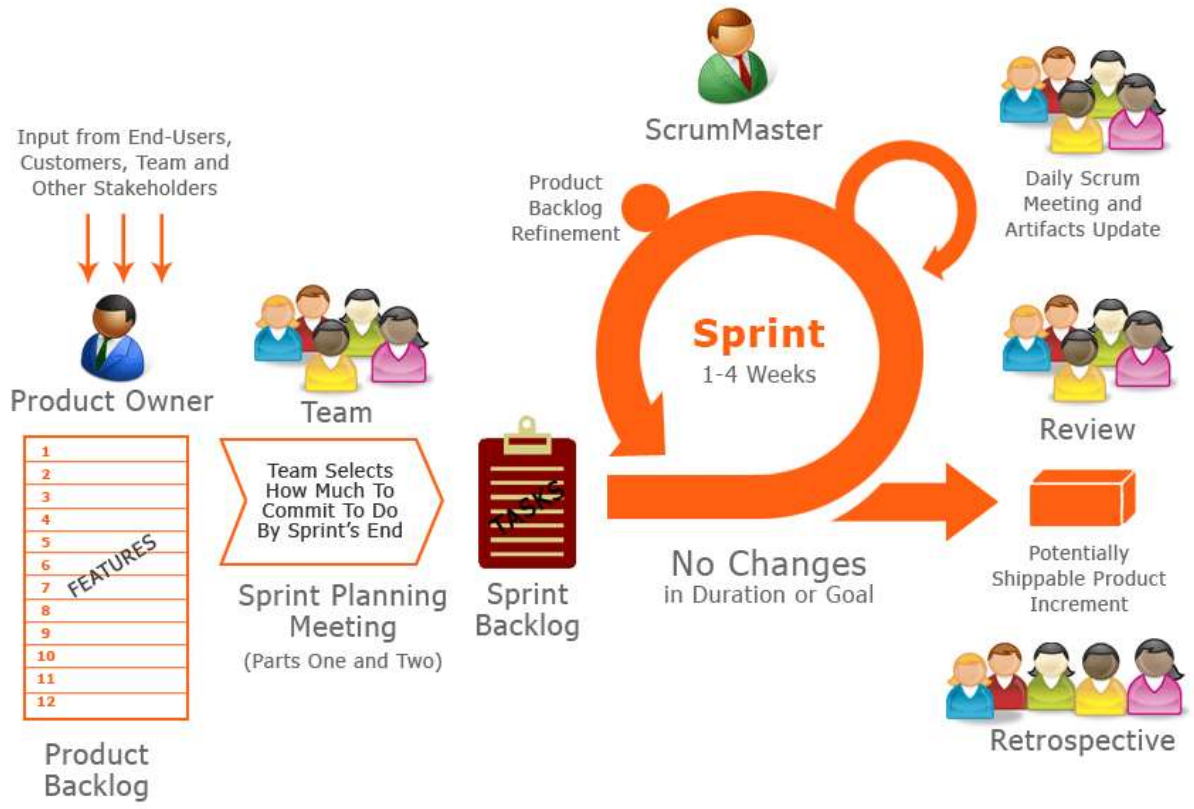
Tareas y evaluación : el planning poker

Tareas y evaluación : el planning poker

- **Expresar libremente su punto de vista** sobre la evaluación de una tarea sin prejuicios
- **Debate** entre los miembros del equipo con el fin de estimar lo mejor posible una funcionalidad
- Aspecto **lúdico**
- Puesta en común de los conocimientos y las dificultades potenciales
- Noción de **complexidad relativa**
- Story points (puntos de historia) o Días Ideales

- Seguir la velocidad al curso de los sprints

Scrum - Síntesis



Scrum - Vocabulario

- *Scrum* = melé
- *Sprint* = iteración

- *Product Backlog* = pila de producto
- *Sprint Backlog* = pila del sprint
- *Release* = incremento del producto

- *Daily Scrum* = melé diaria
- *Sprint Planning* = reunión de planificación
- *Sprint Review* = revisión del sprint
- *Sprint Retrospective* = retrospectiva del sprint

- *Grooming* = Refinado
- *Burndown Chart* = gráfico de combustión

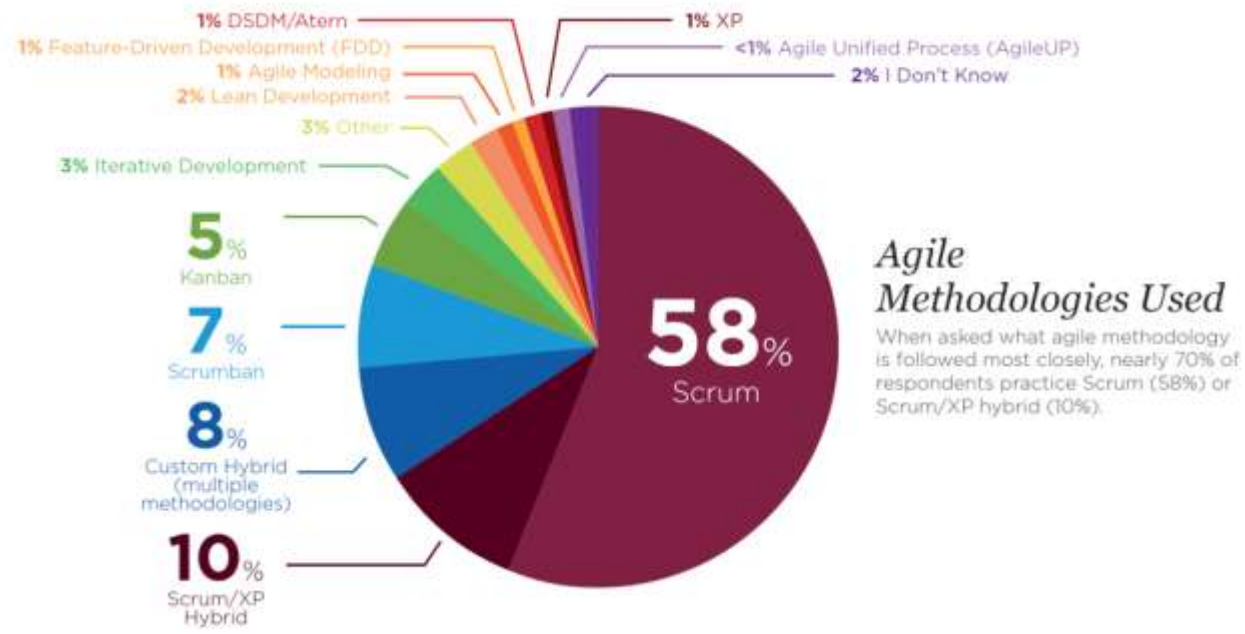
- *Scrum Master* = maestro de melé
- *Product Owner* = propietario del producto
- *Dev. Team* = equipo de desarrollo

- *Definition of Done* = definición de acabado
- *Scrum Board* = Tablero Scrum

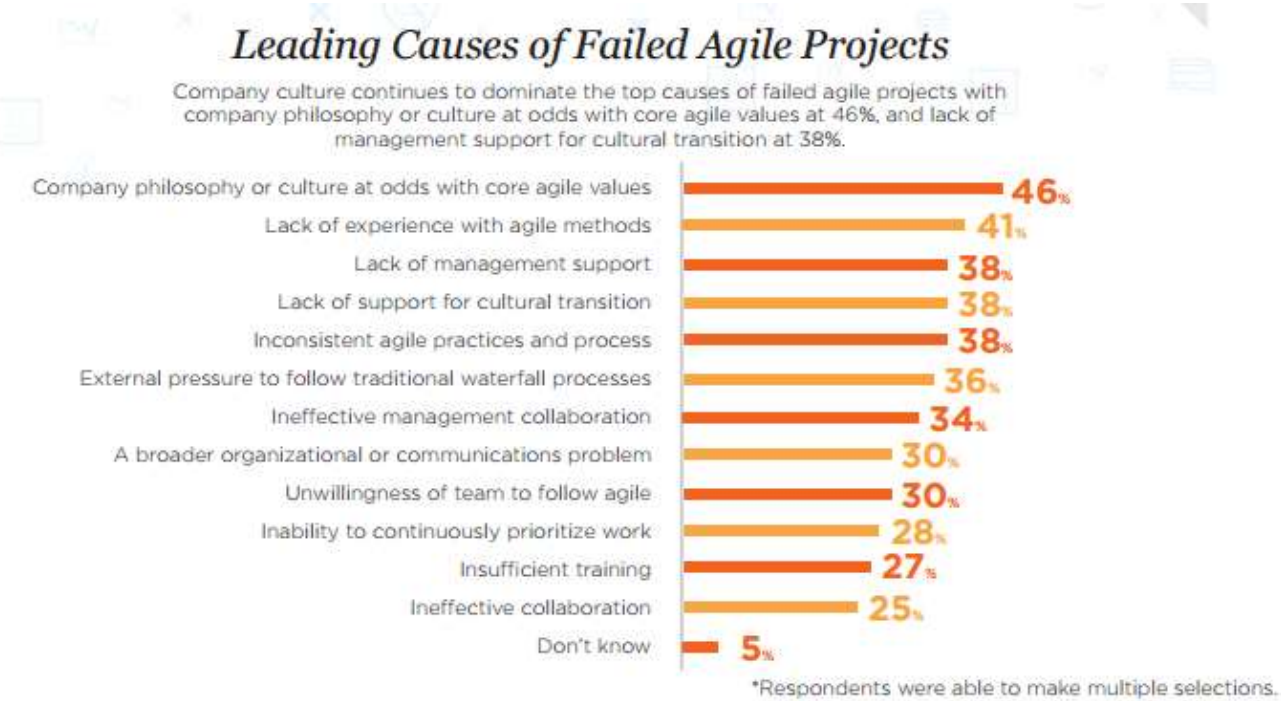
- *Epic* = funcionalidad a descomponer en User stories
- *User story* = Historia del usuario

Scrum - El marco más usado

- Estudio VersionOne de 2015



Motivos de los fracasos observados



fundación esplai
ciudadanía comprometida