

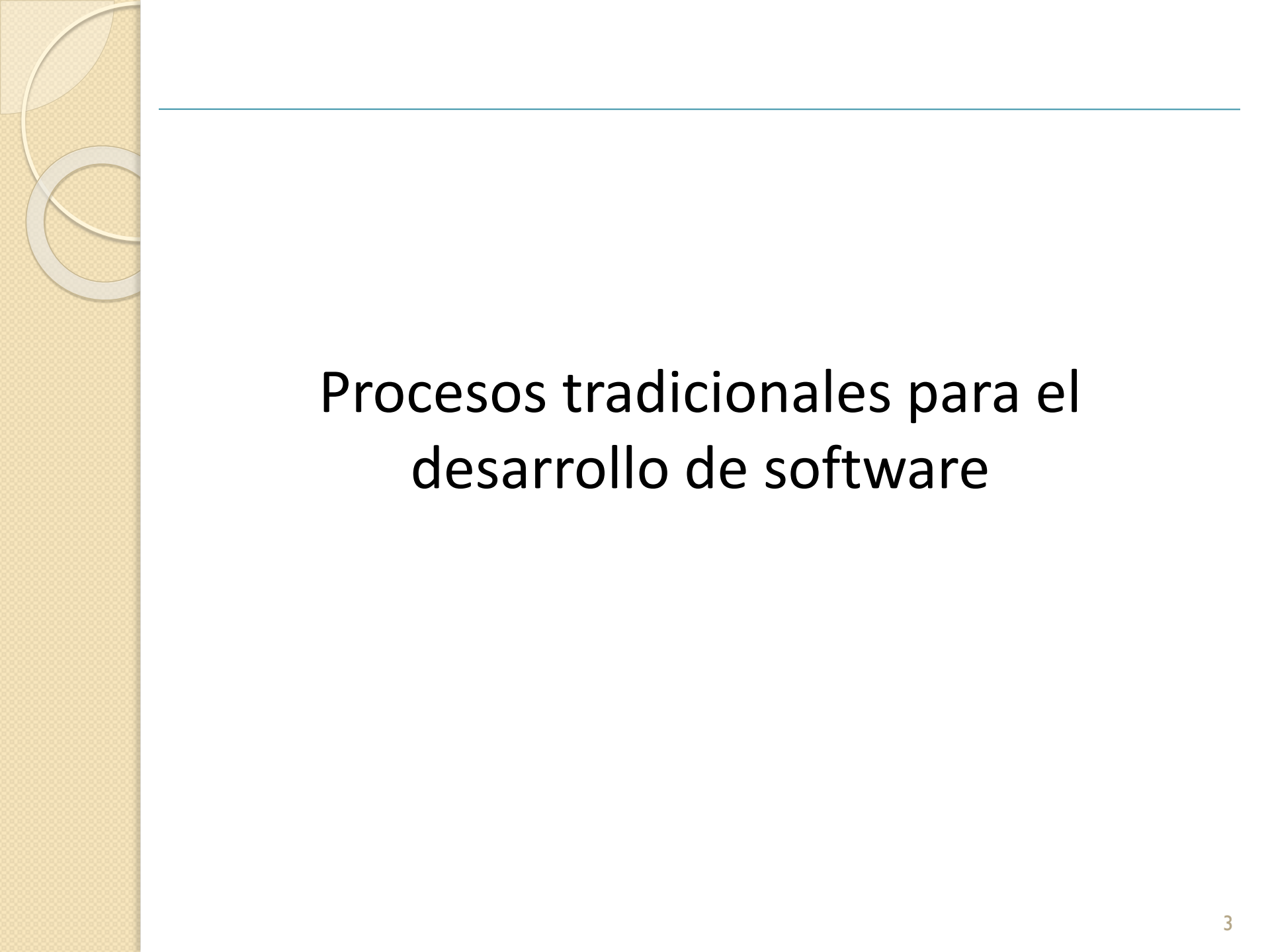
Ingeniería de software ágil 1

UT1. Procesos ágiles

Depto. de Ingeniería de Software - Facultad de Ingeniería
Universidad ORT Uruguay

Temario

- Proceso software.
- Procesos “tradicionales”: origen y características.
- El concepto de agilidad.
- Procesos “ágiles”: origen y características.
- El “Manifiesto ágil”.
- Principios del manifiesto ágil.
- Dominios de complejidad.
- Equipos auto-gestionados



Procesos tradicionales para el desarrollo de software

Procesos tradicionales

- En la década de los '80 y principios de los '90, había una opinión generalizada de que la mejor manera de crear buen software era utilizar procesos de desarrollo de software controlados y rigurosos.
- Estos procesos incluyen:
 - planificación detallada del proyecto.
 - especificación y análisis de requisitos.
 - uso de métodos de análisis y diseño respaldados por herramientas de software.
 - aseguramiento formal de la calidad.

Procesos tradicionales

- Este enfoque suele denominarse “dirigido por un plan” (*plan-driven*).
- El desarrollo dirigido por un plan implica una **sobrecarga significativa de trabajo en la planificación, el diseño y la documentación** del sistema.
- El sistema se especifica en detalle antes de que comience la implementación.
- Los errores de especificación, las omisiones y los malentendidos a menudo **se descubren** solo después de que se haya implementado una parte significativa del sistema.

Procesos tradicionales

- Para solucionar estos problemas, los desarrolladores deben rehacer el trabajo que pensaban que estaba “completo”.
- Como consecuencia, es prácticamente **imposible entregar software rápidamente** y responder de manera oportuna a las **solicitudes de cambios** en el software entregado.

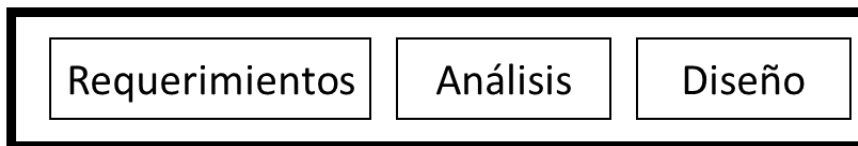
Procesos tradicionales

- Inspirados en las ingenierías clásicas.
- Basado en separar el diseño del producto de su construcción.
- El diseño detallado permite:
 - Definir qué es lo que hay que construir: componentes del producto y cómo se ensamblan.
 - Crear un plan detallado de lo que hay que construir: .
 - Transferir el diseño y el plan de construcción a quienes son responsables del desarrollo del producto.
 - Construir el producto siguiendo los lineamientos del diseño y el plan de construcción.

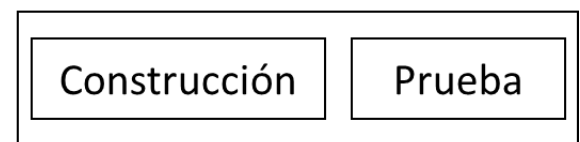
Procesos tradicionales

- La suposición fundamental es que la ingeniería de software exitosa requería mucho trabajo preparatorio antes de comenzar a escribir código.
- Por ejemplo, es importante dedicar tiempo a obtener los requisitos "correctos" y crear modelos gráficos (diagramas) del software.
- Estos modelos se crean durante el proceso de diseño del software y se utilizan para documentar el software.

Front-Loaded



Back-Loaded



Premisas de los procesos tradicionales

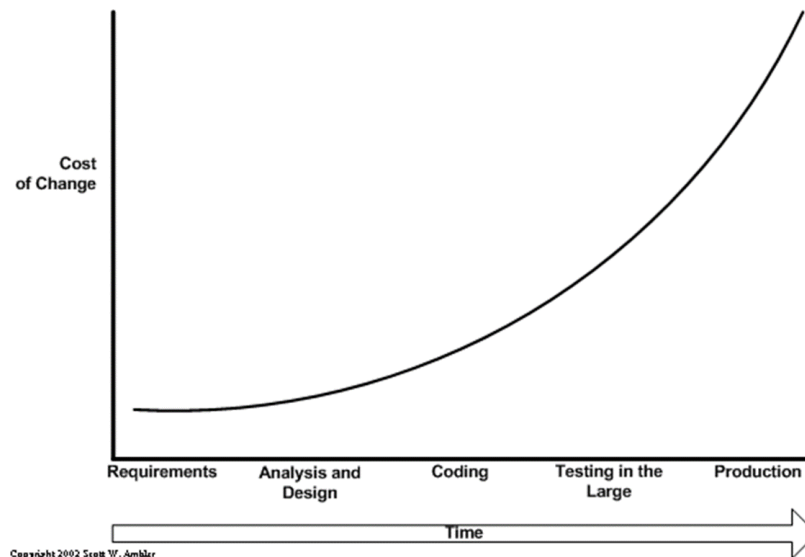
- Negociaciones contractuales:
 - Invertir esfuerzo al comienzo del proyecto para definir los requisitos que el producto debe satisfacer.
 - Al definirse “claramente” esos requisitos, se reduce la probabilidad de tener que realizar cambios durante el proyecto.
- Seguimiento de un plan:
 - Definir el alcance del proyecto (todo lo que hay que hacer).
 - Planificar todo el trabajo a realizar, considerando alcance, cronograma, presupuesto, calidad, recursos.

Premisas de los procesos tradicionales

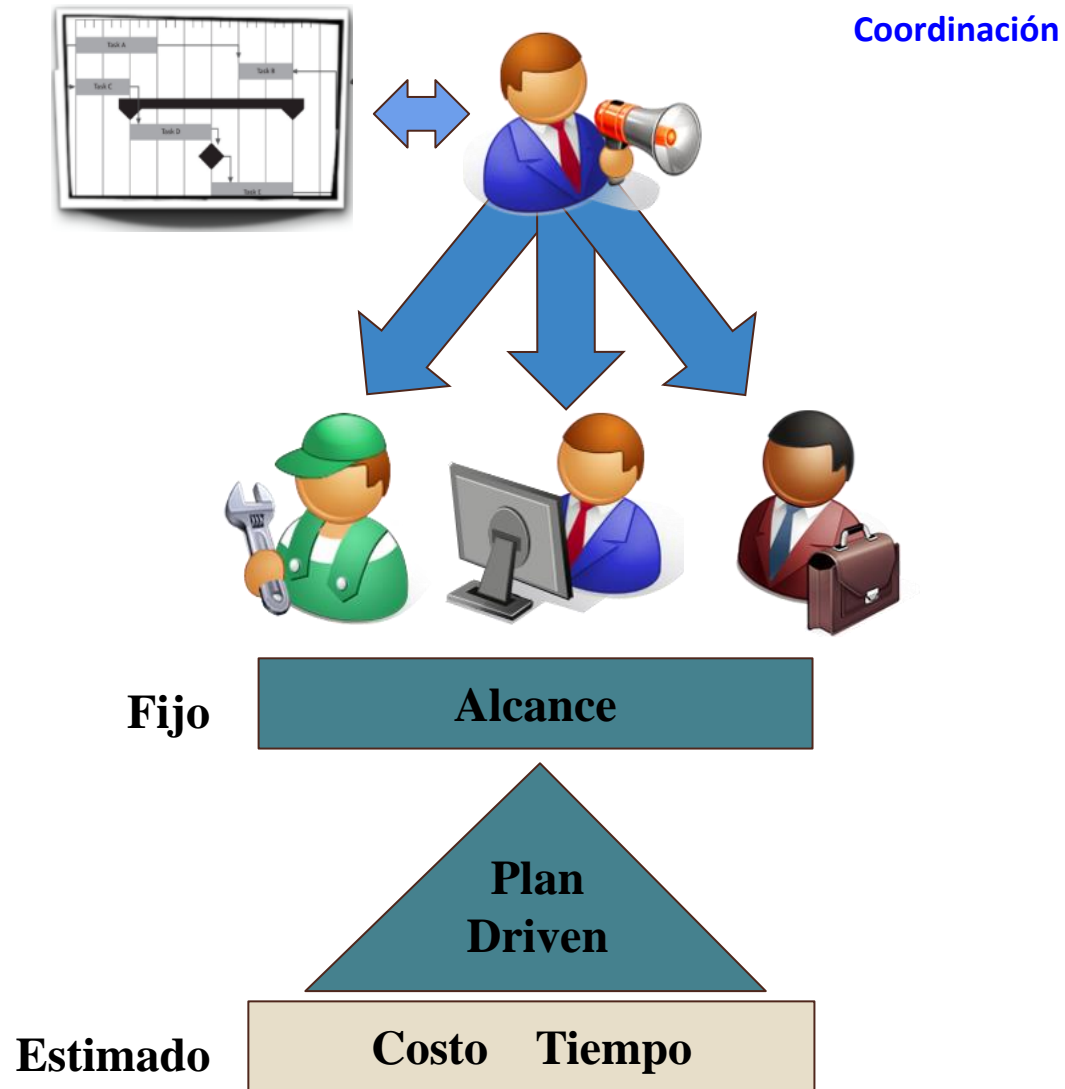
- Documentación extensiva:
 - Documentar en forma detallada los requisitos y el diseño para que los desarrolladores puedan implementar.
- Procesos y herramientas:
 - Establecer y aplicar un conjunto definido de actividades.
 - El uso repetido de esas actividades permite optimizar el proceso y mejorarlo con el tiempo....

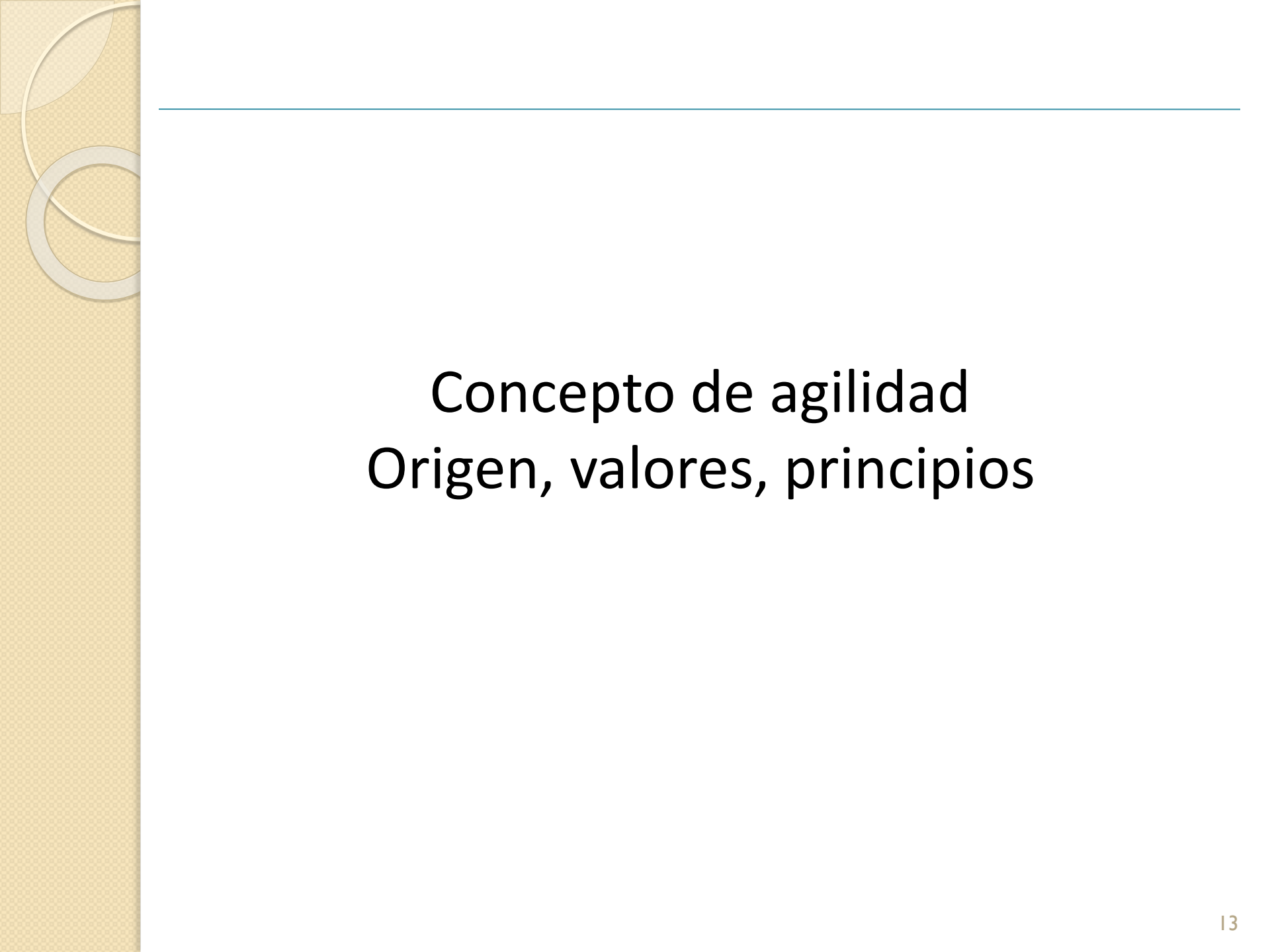
Procesos tradicionales

- El problema con la planificación “por adelantado” de un proyecto es que implica tomar decisiones detalladas sobre el software mucho antes de que comience la implementación.
- Inevitablemente, las cosas cambian:
 - Surgen nuevos requisitos, los miembros del equipo pueden cambiar, las prioridades de negocio evolucionan, etc.



Procesos tradicionales





Concepto de agilidad

Origen, valores, principios

Agilidad

- “ágil” es un término para describir un enfoque para la gestión de proyectos que se centra en:
 - la entrega temprana de valor de negocio.
 - la mejora continua del producto que se crea y de los procesos utilizados para crear el producto.
 - la flexibilidad del alcance.
 - la participación del equipo.
 - la entrega de productos bien probados que reflejan necesidades del cliente.

Origen

- Sobre finales del siglo XX, Internet estaba cambiando el mundo de la tecnología y de su uso por parte de las personas y de las organizaciones.
- Las empresas que trabajaban en la industria del software y en las “.com” estaban sometidas a una presión constante para ser los primeros en comercializar tecnologías que cambiaban rápidamente.
- Los equipos de desarrollo de software trabajaban con intensidad, luchando por entregar nuevas versiones de software antes de que los competidores hicieran obsoletas sus empresas.

Origen

- Dado el ritmo del cambio en aquella época, comenzaron a aparecer fallas e inconformidades con las prácticas convencionales de gestión de proyectos.
- El uso de metodologías tradicionales para la gestión de proyectos (*plan-driven*) y para el desarrollo de software (por ejemplo, cascada), no permitía responder en forma adecuada a la naturaleza dinámica del mercado y a los nuevos enfoques comerciales emergentes.
- Los equipos de desarrollo de productos necesitaban una nueva forma de responder rápidamente a estas demandas para mantenerse competitivos en el mercado cambiante.

Origen

- Algunos sectores de la industria y algunos “*practitioners*” comenzaron a explorar diferentes formas y alternativas a estos enfoques obsoletos de la gestión de proyectos.
- En 2001, un grupo de expertos y de profesionales se reunieron en Snowbird, Utah, para:
 - para compartir sus experiencias, ideas y prácticas para el desarrollo de software.
 - discutir la mejor manera de expresarlos.
 - sugerir formas de mejorar el mundo del desarrollo de productos y servicios basados en software.

Origen

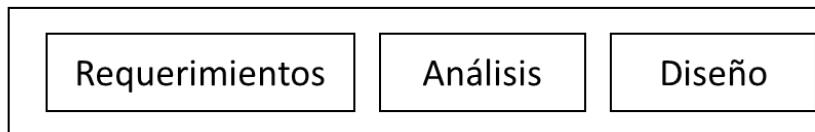
- Los resultados de esa reunión y de otras posteriores fueron:
 - El Manifiesto Ágil: una expresión intencionalmente simplificada de los valores fundamentales del desarrollo.
 - Los Principios Ágiles: un conjunto de 12 conceptos rectores que ayudan a los equipos de desarrollo de productos a ofrecer valor y mantener el rumbo.
 - La Agile Alliance: una organización enfocada en apoyar a individuos y organizaciones que aplican principios y prácticas ágiles.

Manifiesto ágil

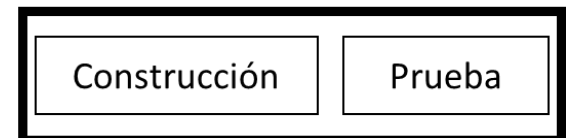
- **Individuos e interacciones** sobre procesos y herramientas.
- **Software funcionando** sobre documentación extensiva.
- **Colaboración con el cliente** sobre negociación contractual.
- **Respuesta ante el cambio** sobre seguir un plan.

Aunque se valoran los elementos de la derecha,
se valoran más los de la izquierda.

Front-Loaded



Back-Loaded



Manifiesto ágil

- **Individuos e interacciones** sobre procesos y herramientas:
 - Es más importante centrarse en las contribuciones que cada persona aporta al equipo y la confianza y la comunicación que existe dentro del equipo que en la adherencia al proceso o el uso de herramientas elaboradas.
 - Las personas son más eficaces cuando pueden hablar y trabajar juntas cara a cara.
 - Reducir las barreras que inhiben la capacidad de las personas para reunirse.

Manifiesto ágil

- **Software funcionando** sobre documentación extensiva:
 - El desarrollo de productos de software es una actividad creativa que es difícil de imaginar por completo mediante el uso de documentos estáticos.
 - Al crear un código funcional que los clientes puedan probar, se obtendrán mejores comentarios de los que se obtendrían cuando los clientes “se imaginan” usando el producto viendo solamente los documentos.
 - En lugar de intercambiar documentos como planes de prueba y especificaciones de productos, los equipos ágiles deben crear versiones de funcionalidad limitada del producto que los clientes y usuarios puedan probar antes de darles su aprobación.

Manifiesto ágil

- **Colaboración con el cliente** sobre negociación contractual:
 - Es más importante que el cliente se involucre íntimamente con el equipo de desarrollo del producto que centrarse en los términos y condiciones del proyecto.
 - Los cronogramas específicos de los proyectos suelen ser difíciles de predecir y de cumplir.
 - En cambio, un proyecto ágil debe centrarse en la interacción regular con los clientes para obtener comentarios (*feedback*) y trabajar con ellos para ajustar los plazos a medida que avanza el proyecto.

Manifiesto ágil

- **Respuesta ante el cambio** sobre seguir un plan:
 - Lo único de lo que la mayoría de los equipos pueden estar seguros cuando inician un proyecto es que algo va a cambiar.
 - Los equipos ágiles deben poder responder y adaptarse a los cambios habituales en lugar de ceñirse a un plan que se creó al comienzo de un proyecto o una fase.
 - Los requisitos del producto se priorizan al comienzo de las iteraciones en lugar de que el equipo se ponga de acuerdo con todos los requisitos del producto al comienzo de un proyecto.

Manifiesto ágil

- En resumen, mientras que los enfoques tradicionales enfatizan un plan rígido, evitan el cambio, documentan todo, y fomentan el control jerárquico, el Manifiesto se enfoca en las personas, la colaboración, el producto, el cambio.



Principios del Manifiesto ágil

- Para apoyar a los equipos que realizan la transición a enfoques ágiles, los cuatro valores anteriores del manifiesto ágil se complementan con doce principios.

12 Principios



Principios del Manifiesto ágil

1. Nuestra mayor prioridad es satisfacer al cliente mediante la entrega temprana y continua de software con valor.
2. Aceptamos que los requisitos cambien, incluso en etapas tardías del desarrollo. Los procesos ágiles aprovechan el cambio para proporcionar ventaja competitiva al cliente.
3. Entregamos software funcional frecuentemente, entre dos semanas y dos meses, con preferencia al periodo de tiempo más corto posible.
4. Los responsables de negocio y los desarrolladores trabajamos juntos de forma cotidiana durante todo el proyecto.

Principios del Manifiesto ágil

- 5. Los proyectos se desarrollan en torno a individuos motivados. Hay que darles el entorno y el apoyo que necesitan, y confiarles la ejecución del trabajo.
- 6. El método más eficiente y efectivo de comunicar información al equipo de desarrollo y entre sus miembros es la conversación cara a cara.
- 7. El software funcionando es la medida principal de progreso.

Principios del Manifiesto ágil

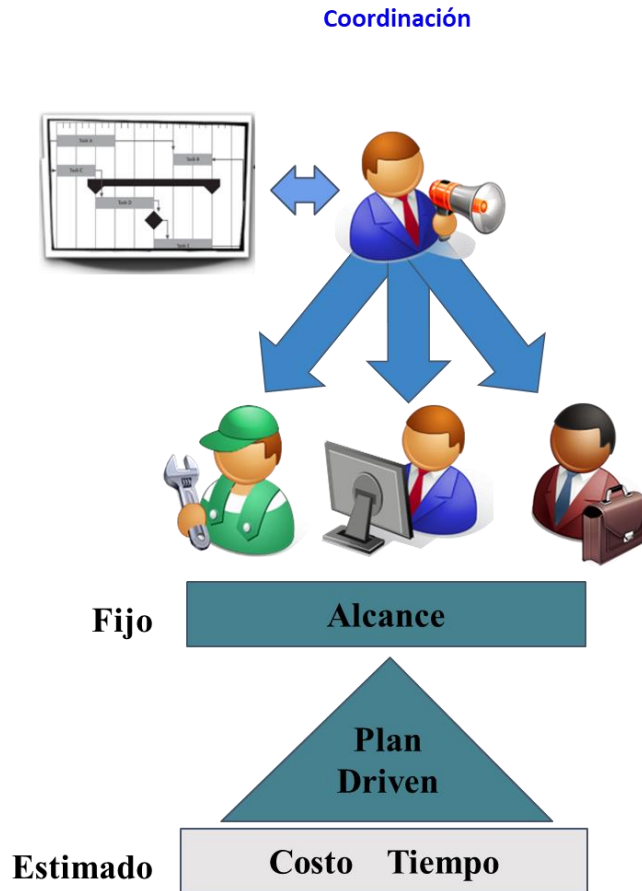
8. Los procesos ágiles promueven el desarrollo sostenible. Los promotores, desarrolladores y usuarios debemos ser capaces de mantener un ritmo constante de forma indefinida.
9. La atención continua a la excelencia técnica y al buen diseño mejora la agilidad.
10. La simplicidad, o el arte de maximizar la cantidad de trabajo no realizado, es esencial.
11. Las mejores arquitecturas, requisitos y diseños emergen de equipos auto-organizados.

Principios del Manifiesto ágil

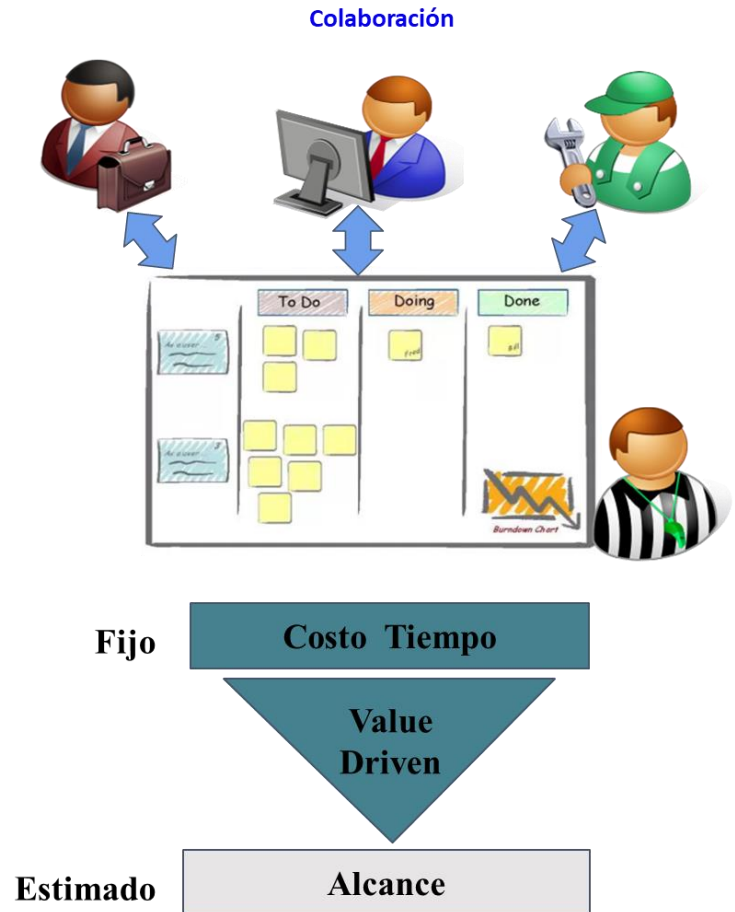
12. A intervalos regulares el equipo reflexiona sobre cómo ser más efectivo para a continuación ajustar y perfeccionar su comportamiento en consecuencia.

Enfoques “tradicional” y “ágil”

Tradicional



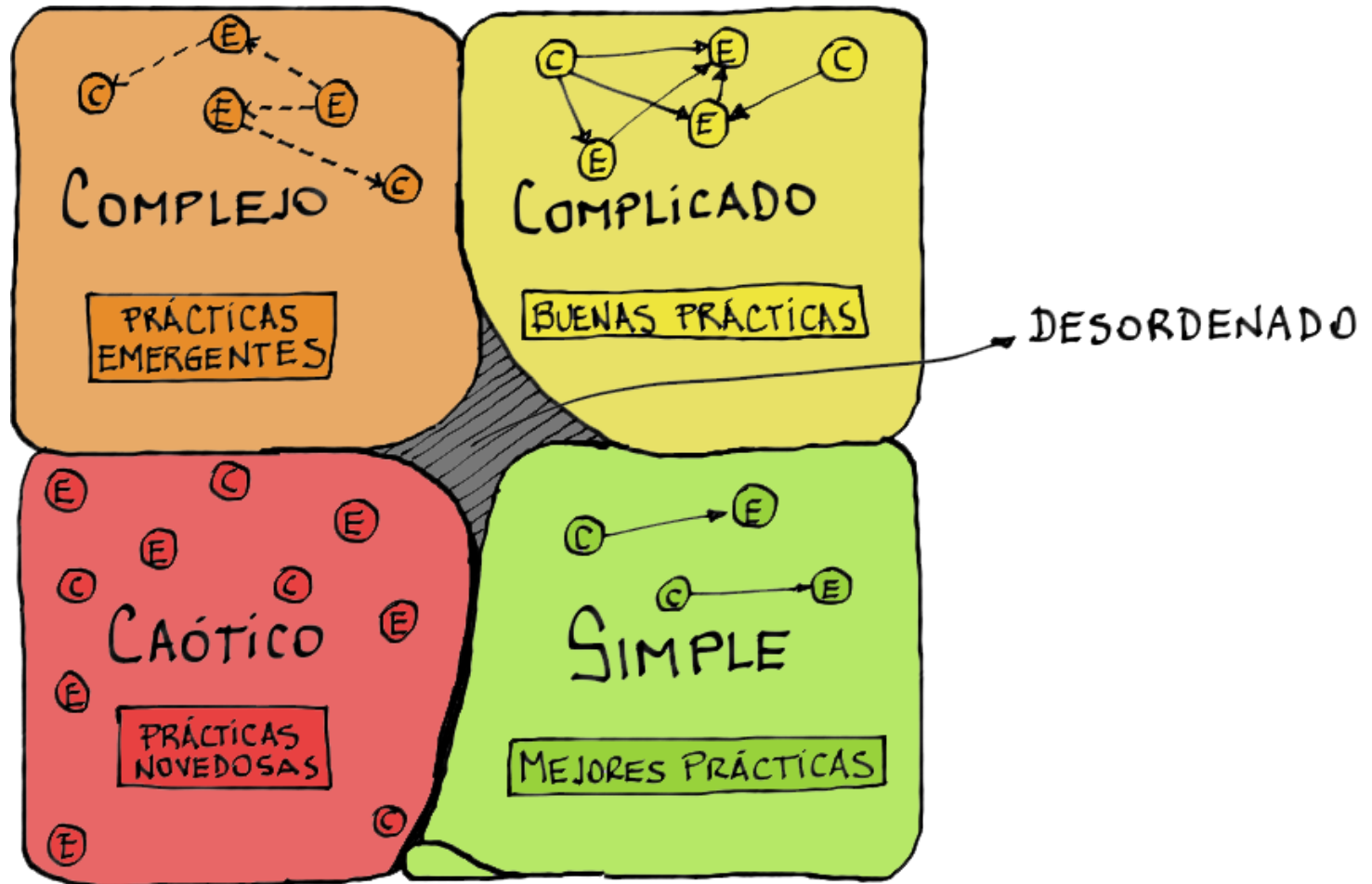
Ágil





Dominios de complejidad

Dominios de complejidad



Dominios de complejidad

- Dominio SIMPLE:
 - problemáticas simples; muy fácil identificar las causas y sus efectos.
 - existen las mejores prácticas, soluciones conocidas para problemas conocidos.
 - los procesos más eficientes en este dominio son aquellos que especifican una serie lógica de pasos y se ejecutan de manera repetitiva, una y otra vez.
 - Ejemplos de este dominio: la construcción en serie de un mismo producto, la instalación en muchos clientes de un mismo sistema.

Dominios de complejidad

- Dominio COMPLICADO:
 - problemas complejos, buenas prácticas y perfiles expertos.
 - Hay múltiples soluciones correctas para una misma problemática, pero se requiere del involucramiento de expertos para poder identificarlas.
 - Para resolver un problema funcionaría mejor un conjunto de expertos en la materia que releven la situación, investiguen diferentes alternativas y planteen la solución en base a las buenas prácticas.
 - Ejemplos de este dominio: problema de performance en un software o base de datos, mantenimiento de sistemas y soporte técnico.

Dominios de complejidad

- Dominio COMPLEJO:
 - no existen ni mejores ni buenas prácticas catalogadas para las situaciones frente a las cuales nos podemos encontrar.
 - no se sabe con anticipación si una determinada solución va a funcionar; solo se pueden examinar los resultados y adaptarse.
 - es el dominio de las prácticas emergentes. Las soluciones encontradas rara vez son replicables, con los mismos resultados, a otros problemas similares.
 - se necesita generar contextos donde haya lugar para la experimentación y donde el fallo sea de bajo impacto.
 - se requieren niveles altos de creatividad, innovación, interacción y comunicación.

Dominios de complejidad


- Dominio COMPLEJO:
 - El desarrollo de nuevos productos o la incorporación de nuevas características en productos existentes es un contexto complejo en el que Scrum se utiliza mucho para actuar, inspeccionar y adaptar las prácticas emergentes de un equipo de trabajo.

Dominios de complejidad

- Dominio CAÓTICO:
 - Los problemas caóticos requieren una respuesta inmediata.
 - Se está en una crisis y se necesita actuar de inmediato para restablecer cierto orden.
 - Se debe actuar de inmediato, alguien debe tomar el control y mover la situación fuera del caos.
 - Por ejemplo, solucionar el problema inmediatamente (sin importar la forma técnica), para luego, fuera del caos, evaluar y aplicar una solución más robusta, de ser necesario.
 - Este es el dominio de la improvisación.

Dominios de complejidad

- Dominio DESORDENADO:
 - Se está en este dominio cuando no se sabe en qué dominio se está.
 - Es una zona peligrosa, ya que no se pueden medir las situaciones ni determinar la forma de actuar.
 - Es muy típico en estas situaciones que las personas interpreten las situaciones y actúen en base a preferencias personales.
 - El gran peligro en este dominio es actuar de manera diferente a la que se necesita para resolver ciertos problemas.
 - Todo lo que se haga debe estar enfocado a salirse de este espacio hacia uno mejor identificado, para luego actuar de la manera en que dicho dominio lo requiera.

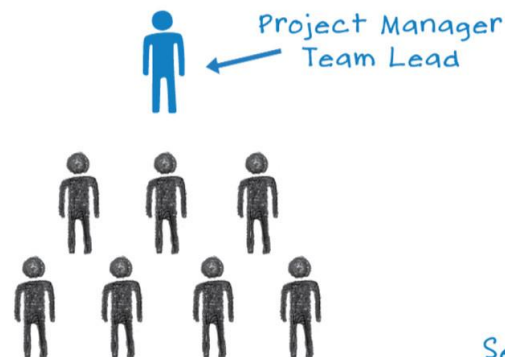


Equipos auto-gestionados o auto-organizados

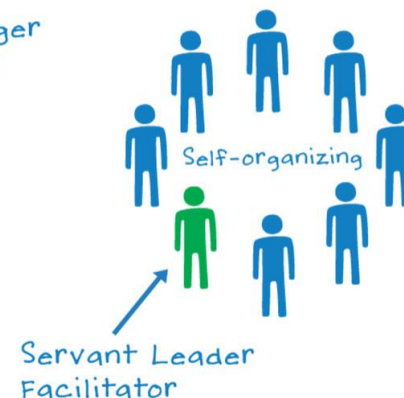
Equipos auto-gestionados

- Un principio fundamental de todos los métodos de desarrollo ágiles es que el equipo de desarrollo de software debe ser autoorganizado.
- Los equipos autoorganizados no tienen un gerente de proyecto que asigne tareas y tome decisiones por el equipo.
- Los equipos autoorganizados trabajan discutiendo problemas y tomando decisiones por consenso.

Traditional Teams



Agile Teams



Equipos auto-gestionados

- Equipo “tradicional”, el gerente del proyecto:
 - coordina el trabajo.
 - define el trabajo por hacer y asigna tareas a los miembros del equipo.
 - tiene que organizar las cosas para que el trabajo no se retrase porque un miembro del equipo está esperando que otros terminen su trabajo.
 - tiene que informar a todos los miembros del equipo sobre los problemas y otros factores que pueden retrasar el trabajo.
 - no suele animar a los miembros del equipo a asumir la responsabilidad de la coordinación y la comunicación.

Equipos auto-gestionados

- Equipo autoorganizado:
 - el propio equipo tiene que establecer formas de coordinar el trabajo y comunicar los problemas a todos los miembros del equipo.
 - debido a que el equipo, más que los individuos, asume la responsabilidad del trabajo, el equipo puede hacer frente a cambios en su integración.

Espacios de trabajo colaborativo

