FSID

Fundamentos de los Sistemas de Informacion Digitales

Metodologías Ágiles

Año 2025

TEMARIO

- ¿Que es?
- 12 Principios del manifiesto de la Agilidad
- Conceptos generales
- Ceremonias
- Roles

Teoria de SCRUM

> Se basa en el empirismo y el pensamiento Lean

- El empirismo afirma que el conocimiento proviene de la experiencia y de la toma de decisiones con base en lo observado.
- El pensamiento Lean reduce el desperdicio y se enfoca en lo esencial.
- Scrum emplea un enfoque iterativo e Incremental para optimizar la previsibilidad y controlar el riesgo.
- Scrum combina cuatro eventos formales para inspección y adaptación dentro de un evento contenedor, el Sprint.

Pilares de las Metodologias Agiles

- Valorar más a los 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.

Principios

1 Nuestra principal prioridad es satisfacer al cliente a través de la entrega temprana y continua de software de valor.

2 Son bienvenidos los requisitos cambiantes, incluso si llegan tarde al desarrollo.Los procesos ágiles se doblegan al cambio como ventaja competitiva para el cliente.

3 Entregar con frecuencia software que funcione, en periodos de un par de semanas hasta un par de meses, con preferencia en los períodos breves.

4 Las personas del negocio y los desarrolladores deben trabajar juntos de forma cotidiana en el proyecto.

5 Construcción de proyectos en torno a individuos motivados, dándoles la oportunidad y el respaldo que necesitan, y procurando la confianza para que realicen la tarea.

6 La forma más eficiente y efectiva de comunicar información de ida y vuelta dentro de un equipo de desarrollo es mediante la conversación cara a cara.

7 El software que funciona es la principal medida del progreso.
8 Los procesos ágiles promueven el desarrollo sostenido. Los patrocinadores , desarrolladores y usuarios deben mantener un ritmo constante de forma indefinida.

9 La atención continua a la excelencia técnica enaltece la agilidad.

10 Es esencial la simplicidad como arte de maximizar la cantidad de trabajo que se hace.

11 Las mejores arquitecturas, requisitos y diseños emergen de equipos que se autoorganizan.

12 En intervalos regulares, el equipo reflexiona sobre la forma de ser más efectivo y ajusta su conducta en consecuencia.

Programacion extrema (XP)

Es un método ágil ampliamente adoptado que engloba un conjunto de prácticas de programación efectivas.

Esta metodología se destaca por:

- Su enfoque en las liberaciones frecuentes de software
- La mejora continua
- La participación activa del cliente en el equipo de desarrollo.

Se centra en potenciar las relaciones interpersonales como un factor clave para el éxito,
promoviendo el trabajo en equipo,
fomentando el aprendizaje de los desarrolladores y
creando un ambiente de trabajo positivo

Programacion extrema (XP)

Valores de la programación extrema

Simplicidad: Todo el código debe ser lo más simple posible para agilizar el desarrollo.

Comunicación: Es fundamental, dentro del equipo de trabajo y con el cliente, hay que crear un ambiente de cooperación y unidad.

Retroalimentación: Concreta y frecuente con el cliente, del equipo y de los usuarios finales.

Coraje: La valentía permite a los programadores sentirse cómodos con reconstruir su código cuando sea necesario.

Respeto: debe haber respeto y aprecio entre el equipo.

Programacion extrema (XP)

Prácticas de la programación extrema

Planeación del incremento: Los requerimientos se registran en tarjetas de historia. Las historias que se van a incluir en la próxima liberación se determinan por el tiempo disponible y prioridad relativa.

Liberaciones pequeñas: Al principio, se desarrolla el conjunto mínimo de funcionalidad útil que ofrece valor para el negocio. Las liberaciones del sistema son frecuentes y agregan incrementalmente funcionalidad a la primera liberación.

Diseño simple: Se realiza un diseño solo suficiente para cubrir aquellos requerimientos actuales.

Desarrollo de la primera prueba: Las pruebas de unidad (para probar el funcionamiento del módulo de modo aislado del resto) son frecuentes y continuas. Se aconseja escribir el código de la prueba antes de la codificación de modo que sirva como un marco de referencia de prueba de unidad automatizada.

Programacion extrema (XP)

Refactorización: Se espera que todos los desarrolladores refactoricen de manera continua el código, tan pronto como sea posible y se encuentren mejoras de éste. Lo anterior conserva el código simple y mantenible.

Programación en pares: Los desarrolladores trabajan en pares, y cada uno comprueba el trabajo del otro; además, ofrecen apoyo para que se realice siempre un buen trabajo.

Propiedad colectiva: Los desarrolladores en pares laboran en todas las áreas del sistema. Todos los programadores se responsabilizan por el código, de modo que no se generen “islas de experiencia”. Cualquiera puede cambiar cualquier función.

Integración continua: Tan pronto como esté completa una tarea, se integra en todo el sistema. Después de tal integración, deben probarse todas las pruebas de unidad y de integración en el sistema.

Ritmo sustentable: Grandes cantidades de tiempo extra no se consideran aceptables, pues el efecto neto de este tiempo libre con frecuencia es reducir la calidad del código y la productividad de término medio.

Cliente en sitio: Un representante del usuario final del sistema (el cliente) tiene que disponer de tiempo completo para formar parte del equipo XP. En un proceso de programación extrema, el cliente es miembro del equipo de desarrollo y responsable de llevar los requerimientos del sistema al grupo para su implementación.

Roles en Programacion extrema (XP)

• Programador: Es quien escribe las pruebas unitarias y produce el código del sistema. Debe existir una comunicación y coordinación adecuada entre los programadores y otros miembros del equipo.

• Cliente: Encargado de escribir las historias de usuario y las pruebas funcionales para validar su implementación. Además, es quien asigna la prioridad a las historias de usuario y decide cuáles se implementan en cada iteración centrándose en aportar mayor valor al negocio. El cliente es solo uno dentro del proyecto, pero puede corresponder a un interlocutor que está representando a varias personas que se verán afectadas por el sistema.

• Encargado de pruebas (Tester): Ayuda al cliente a diseñar las pruebas funcionales, de ejecutarlas y de brindar los resultados al resto del equipo.

Roles en Programacion extrema (XP)

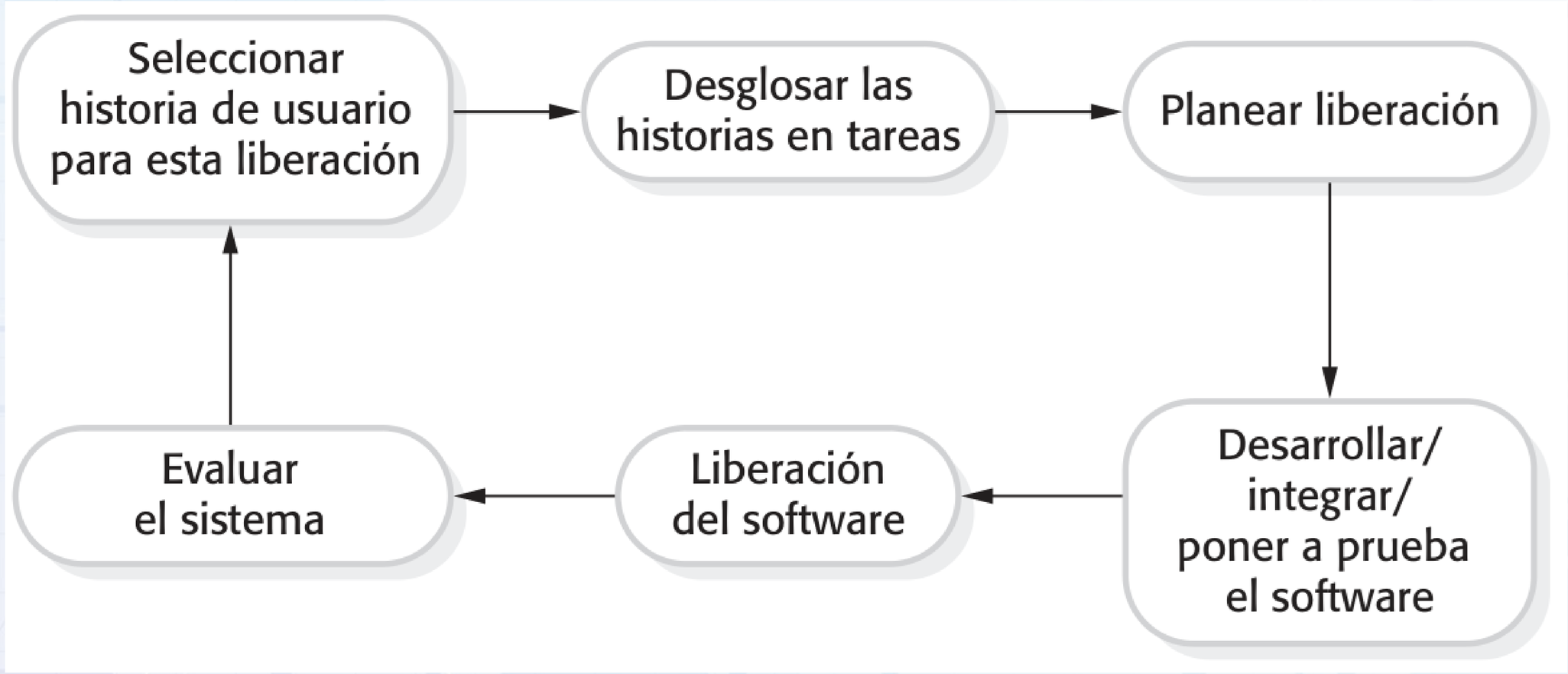
• Encargado de seguimiento (Tracker). Es el responsable de que se cumplan las estimaciones realizadas y de ver si los objetivos trazados en cada iteración fueron alcanzados.

• Entrenador (Coach). Es la persona que conoce a fondo el proceso XP y el responsable del proceso global. Debe proveer guía a todos los miembros del equipo para que el proceso sea satisfactorio.

• Consultor. Es un miembro externo del equipo con un conocimiento específico en algún tema necesario para el proyecto. Guía al equipo para resolver un problema específico.

• Gestor (Big Boss). Es quien coordina el vínculo entre clientes y programadores, el que ayuda a que las condiciones de trabajo sean las adecuadas.

Ciclo de desarrollo de Programación extrema (XP)



Historias de usuario

Básicamente, una historia de usuario es un requerimiento que tiene un determinado usuario y que lo expresa, de modo simple, desde su perspectiva. Suelen tener el siguiente formato:

COMO <rol>
QUIERO <descripción de eventos que queremos que ocurra>
PARA <descripción de funcionalidad>

Ejemplo en un banco

COMO Oficial de atención al cliente
QUIERO Presionar una tecla y acceder a los productos que el cliente tiene contratados
PARA Ofrecer nuevos productos

También se clasifican las historias de usuario por su complejidad para poder determinar la cantidad de tiempo que llevará la implementación de dicha HU

Modelo Scrum

El modelo Scrum es el modelo más aplicado entre las organizaciones que utilizan desarrollo ágil.

A diferencia de otros modelos, este no refiere solamente a un modelo de construcción de software,

sino la metodología ágil enfocada en la gestión general de proyectos.

El proceso Scrum

El proceso de Scrum se compone de tres fases bien definidas. La primera fase es el planeamiento del sprint (“sprint planning”, en inglés), en la cual se establecen los objetivos generales del proyecto y se diseña la arquitectura de software. Esta etapa sienta las bases para el desarrollo posterior del sistema.

La segunda fase es conocida como los Sprints, que consisten en una serie de ciclos de trabajo. En cada sprint, el equipo se enfoca en desarrollar un incremento del sistema. Durante la planificación del sprint, se seleccionan las características o funcionalidades específicas a desarrollar y se realiza la implementación del software correspondiente. Al finalizar cada sprint, se entrega la funcionalidad completa a los participantes.

La tercera y última fase es la Revisión del Sprint, donde el equipo realiza una evaluación exhaustiva del sprint finalizado. Se identifican áreas de mejora y se definen acciones para el próximo ciclo. Este proceso de revisión y mejora continua asegura que el equipo se adapte y aprenda de cada iteración, optimizando así su desempeño.

Una vez concluida la revisión del sprint, el equipo reinicia el proceso, comenzando un nuevo ciclo con una nueva planificacion del bosquejo. De esta manera, Scrum permite un enfoque iterativo e incremental en el desarrollo de software, promoviendo la entrega continua de valor y la mejora constante del equipo.

Caracteristicas del proceso Scrum

1. Los sprints tienen longitud fija, por lo general de dos a cuatro semanas. Una vez definida la duración ideal del sprint, en función de las características del desarrollo, la misma no debe cambiarse ya que, en cierto modo, pasan a conformar la medida del ritmo de trabajo.

2. El punto de partida para la planeación es el Product Backlog, que es la lista de trabajo pendiente de realizar en el proyecto. Los nuevos requerimientos se van incorporando a esta lista. Al comenzar un sprint se revisan las tareas pendientes, se priorizan y se deciden cuál o cuáles de ellas serán desarrolladas en ese sprint.

3. La fase de selección de tareas incluye a todo el equipo del proyecto que trabaja con el cliente, con la finalidad de priorizar y elegir las características y la funcionalidad a desarrollar.

4. Una vez seleccionadas las tareas, el equipo se organiza para desarrollar el software. Con el objetivo de revisar el progreso y, si es necesario, volver a asignar prioridades al trabajo, se realizan reuniones diarias breves con todos los miembros del equipo. Estas reuniones diarias habitualmente se realizan sin sillas para que sean cortas y efectivas. Durante esta etapa, el equipo se aísla del cliente y la organización, y todas las comunicaciones se canalizan a través del llamado “Scrum Master”. El papel de este último es proteger al equipo de desarrollo de distracciones externas y atender las necesidades que los miembros plantean en la reunión diaria.

5. La forma exacta en que el trabajo se realiza puede variar y depende del problema y del equipo.

6. Al final del sprint, el trabajo hecho se revisa y se presenta a los diferentes interesados, continuándose con el siguiente sprint.

Roles en Scrum

1. Roles comprometidos directamente con el proyecto: Estos roles son los que obligatoriamente se requieren para producir el producto o del proyecto. Dichos roles son los responsables del éxito de cada sprint y del proyecto en sí:

- Product Owner es la persona responsable de lograr el máximo valor empresarial para el proyecto. También es responsable de la articulación de requisitos del cliente. El Product Owner representa la voz del cliente dentro del proyecto.

- Equipo Scrum es el grupo o equipo de personas responsables de la comprensión de los requisitos especificados por el Product Owner, y de la creación de los entregables del proyecto.

Scrum Master es un facilitador que asegura que el equipo Scrum esté dotado de un ambiente propicio para completar el proyecto con éxito. El Scrum Master guía, facilita y les enseña las prácticas de Scrum a todos los involucrados en el proyecto; elimina los impedimentos que encuentra el equipo; y asegura que se estén siguiendo los procesos de Scrum.

2. Otros Roles involucrados con el proyecto: Son aquellos roles que, si bien son importantes, no participan directamente del proceso Scrum. No siempre están presentes y no son obligatorios, aunque en muchos proyectos desempeñan un papel muy importante y deben ser tenidos en especialmente en cuenta. Como ejemplo, podemos nombrar a los Stakeholder(s) (cliente, usuarios, accionistas).

Ceremonias en Scrum

1. **Sprint Planning o planificación inicial**, con la participación de todo el equipo de desarrollo, Es donde el Product Owner presenta la versión actualizada y priorizada del Backlog, para que el equipo pueda estimar que tareas podrán ser incluidas en el próximo sprint. Se define entonces
que se va a hacer y cómo se realizará.

2. **Daily Meeting o reuniones rápidas diarias**, son encuentros breves para que cada miembro del equipo informe que tarea realizará ese día. También informará al Scrum Master si tiene algún impedimento para que éste pueda tratar de solucionarlo.

3. **Sprint Review,** es la reunión final al terminar el sprint, donde el product owner y el equipo de desarrollo presentan a los stakeholders el incremento terminado, para que lo inspeccionen y realicen las observaciones pertinentes. No se trata de una demo, sino que se presenta el software funcionando en producción. De esta reunión pueden surgir nuevas necesidades que pasan a actualizar el Product Backlog.

4. **Sprint Retrospective**. Esta reunión, que puede realizarse en conjunto con lo anterior, consiste en reflexionar sobre el sprint que ha finalizado, identificando posibles cosas que puedan mejorarse. Es común analizar que salió bien, que ha fallado y que cosas podrían hacerse de un modo más eficiente.

5. **Refinamiento del Product Backlog**. Esta reunión adicional permite depurar el listado de pendiente y completar, refinar o aclarar ciertas historias de usuario que pudieron quedar pendientes durante el Sprint Planning.

Tablero Scrum

Es una herramienta muy útil en la metodología Scrum, ya que permite visualizar y gestionar el flujo de trabajo de manera efectiva. Cada columna del tablero representa una etapa del proceso, y las tarjetas representan las historias de usuario o tareas a realizar.

En un tablero Scrum, las tarjetas se mueven a través de diferentes columnas que representan las etapas del flujo de trabajo. Comúnmente, se utilizan las siguientes columnas: "Por hacer", "En progreso", "Testeo" y "Terminadas". Estas columnas reflejan el ciclo típico de trabajo en Scrum.

En la columna "Por hacer", se encuentran las tareas que aún no han sido iniciadas. A medida que un miembro del equipo comienza a trabajar en una tarea, esta se mueve a la columna "En progreso". Una vez que la tarea está completa, se traslada a la columna "Testeo" para realizar las pruebas necesarias. Finalmente, cuando la tarea ha pasado exitosamente las pruebas, se coloca en la columna "Terminadas".

Es importante destacar que el tablero Scrum puede adaptarse a las necesidades específicas del equipo y el proyecto. Se pueden agregar columnas adicionales según las etapas o actividades que sean relevantes para el flujo de trabajo en particular. Por ejemplo, se pueden incluir columnas como "En revisión", "En espera" o "Bloqueadas" para identificar y abordar posibles cuellos de botella o problemas que requieren atención adicional.

La personalización del tablero Scrum permite que el equipo tenga una representación visual clara y específica de su flujo de trabajo, lo que facilita la identificación de áreas de mejora, la gestión de tareas y la resolución de problemas de manera más efectiva.

Tablero Scrum

El tablero Scrum es una herramienta fundamental que proporciona una visualización clara del progreso de cada tarea y facilita la sincronización diaria en las reuniones del equipo Scrum. Permite identificar de manera rápida y sencilla las tareas que están en curso, las que están pendientes y las que ya han sido completadas. Esto ayuda a mantener a todos los miembros del equipo informados y al tanto del estado del proyecto. Esto facilita la colaboración y la toma de decisiones conjuntas, ya que todos tienen acceso a la misma información actualizada.

Además, el tablero Scrum ayuda a distribuir el trabajo de manera equitativa entre los miembros del equipo. Cuando una persona completa una tarea, puede pasar a ocuparse de la siguiente tarea pendiente, lo que ayuda a mantener un flujo constante y evita la acumulación de trabajo en una sola persona. Esto fomenta la colaboración y la capacidad del equipo para responder de manera ágil a los cambios y desafíos que puedan surgir durante el proyecto.

Es importante destacar que existen herramientas informáticas que permiten crear y gestionar tableros Scrum en línea, lo que facilita la colaboración y el seguimiento del progreso del proyecto por parte de todos los miembros del equipo, incluso si están trabajando de forma remota

Tablero Scrum de ejemplo

FSID

GRACIAS

Año 2025