

APUNTES DE INTRODUCCIÓN.

ENTORNO DE DESARROLLO



5 de octubre de 2018

profesor: joaquín delhom viana

ENRIQUE LÓPEZ MARTÍNEZ 1º DAW

1. **Define "Ciclo de vida del software".**

Es una sucesión de estados o fases por los cuales pasa un software a lo largo de su "vida".

1. **Nombra las fases principales del desarrollo de software y explica brevemente que se hace en cada una de ellas.**

**Análisis de requisitos**: Extraer los requisitos de un producto de software es la primera etapa para crearlo. Mientras que los clientes piensan que ellos saben lo que el software tiene que hacer, se requiere de habilidad y experiencia en la ingeniería de software para reconocer requisitos incompletos, ambiguos o contradictorios.

**Diseño y arquitectura**: Consiste en incorporar consideraciones de la implementación tecnológica, como el hardware, la red, etc.

**Programación**: Reducir un diseño a código puede ser la parte más obvia del trabajo de ingeniería de software, pero no es necesariamente la porción más larga.

**Pruebas**: Consiste en comprobar que el software realice correctamente las tareas indicadas en la especificación.

**Documentación**: Todo lo concerniente a la documentación del propio desarrollo del software y de la gestión del proyecto, pasando por modelaciones (UML), diagramas, pruebas, manuales de usuario, manuales técnicos, etc

**Mantenimiento**: Mantener y mejorar el software para enfrentar errores descubiertos y nuevos requisitos. Esto puede llevar más tiempo incluso que el desarrollo inicial del software.

1. **Explica brevemente en qué consiste el modelo en cascada cuando hablamos de desarrollo de software.**

Es un proceso de desarrollo secuencial, en el que el desarrollo de software se concibe como un conjunto de etapas que se ejecutan una tras otra.

1. **Ventajas e inconvenientes del modelo en cascada.**

Ventajas

* Realiza un buen funcionamiento en equipos débiles y productos maduros, por lo que se requiere de menos capital y herramientas para hacerlo funcionar de manera óptima.
* Es un modelo fácil de implementar y entender.
* Es un modelo conocido y utilizado con frecuencia.

Desventajas

* El proceso de creación del *software* tarda mucho tiempo ya que debe pasar por el proceso de prueba y hasta que el *software* no esté completo no se opera. Esto es la base para que funcione bien.
* Cualquier error de diseño detectado en la etapa de prueba conduce necesariamente al rediseño y nueva programación del código afectado, aumentando los costos del desarrollo.
* Una etapa determinada del proyecto no se puede llevar a cabo a menos de que se haya culminado la etapa anterior.

1. **¿Qué se entiende por verificación? ¿Y por validación?**

Verificación: Se comprueba que el software cumple los requisitos funcionales y no funcionales de su especificación.

Validación: Comprueba que el software cumple las expectativas que el cliente espera.

1. **Explica cómo funciona el modelo de desarrollo mediante creación de prototipos.**

Se centra en una representación de aquellos aspectos del software que serán visibles para el cliente o el usuario final. Este diseño conduce a la construcción de un prototipo, el cual es evaluado por el cliente para una retroalimentación; gracias a ésta se refinan los requisitos del software que se desarrollará. La interacción ocurre cuando el prototipo se ajusta para satisfacer las necesidades del cliente. Esto permite que al mismo tiempo el desarrollador entienda mejor lo que se debe hacer y el cliente vea resultados a corto plazo.

1. **Explica cómo funciona el modelo espiral cuando se aplica al desarrollo orientado a objetos.**

* En el modelo espiral, el software se desarrolla en una serie de versiones incrementales. Durante las primeras iteraciones la versión incremental podría ser un modelo en papel o un prototipo, durante las últimas iteraciones se producen versiones cada vez más completas del sistema diseñado.
* EL modelo en espiral se divide en un número de actividades de marco de trabajo, también llamadas **REGIONES DE TAREAS**, Cada una de las regiones están compuestas por un conjunto de tareas del trabajo llamado **CONJUNTO DE TAREAS** que se adaptan a las características del proyecto que va a emprenderse en todos los casos se aplican actividades de protección.

1. **¿Qué cuatro principios rigen el desarrollo ágil expresados en el Manifiesto Ágil?**

### Valorar más a los individuos y sus interacciones que a los procesos y las herramientas

### Valorar más el software funcionando que la documentación exhaustiva

### Valorar más la colaboración con el cliente que la negociación contractual

### Valorar más la respuesta ante el cambio que seguir un plan

1. **¿Qué es una historia de usuario? Consulta el siguiente enlace:**

Es una representación de un requisito escrito en una o dos frases utilizando el lenguaje común del usuario.

1. **Haz un resumen sobre que se entiende por lean el software y qué principios lo rigen. Consulta el siguiente enlace:**

Es una traducción de los principios y las prácticas de la forma de producir lean, hacia el área del desarrollo de software.

Los principios que lo rigen son:

**Eliminar los desperdicios**

Todo lo que no añade valor al cliente se considera un desperdicio:

• Código y funcionalidades innecesarias.

• Retraso en el proceso de desarrollo de software.

• Burocracia.

Con el fin de poder eliminar los desperdicios deberíamos ser capaces de reconocerlos y encontrarlos. Los procesos y funcionalidades extra que no son usados por el cliente son desperdicios.

**Amplificar el Aprendizaje**

La acumulación de defectos debe evitarse ejecutando las pruebas tan pronto como el código

está escrito en lugar de añadir más documentación o planificación detallada.El proceso de recopilación de requisitos de usuarios podría simplificarse mediante la presentación de las pantallas de los usuarios finales para que estos puedan hacer sus aportes.

**Decidir lo más tarde posible**

Cuanto más complejo es un proyecto, más capacidad para el cambio debe incluirse en éste, así que debe permitirse el retraso de los compromisos importantes y cruciales. Un enfoque de desarrollo de software ágil puede llevarles opciones rápidamente a los clientes, lo que implica,

retrasar algunas decisiones cruciales hasta que los clientes hayan reconocido mejor sus necesidades.

**Entregar tan rápido como sea posible**

Cuanto antes se entrega el producto final sin defectos considerables más pronto se pueden recibir comentarios y se incorporan en la siguiente iteración.

La velocidad asegura el cumplimiento de las necesidades actuales del cliente y no lo que éste requería para ayer. Esto les da la oportunidad de demorarse pensando lo que realmente necesitan, hasta que adquieran un mejor conocimiento.

**Capacitar al equipo**

una creencia tradicional en la mayoría de las empresas acerca de la toma de decisiones en la organización: los administradores dicen a los trabajadores cómo hacer su propio trabajo. En una técnica llamada Work-Out, los roles cambian: a los directivos se les enseña a escuchar a los desarrolladores, de manera que éstos puedan explicar mejor qué acciones podrían tomarse, así como ofrecer sugerencias para mejoras.

**Construir integridad intrínseca**

Integridad Conceptual significa que los componentes separados del sistema funcionan bien juntos, como en un todo, logrando equilibrio entre la flexibilidad, sostenibilidad, eficiencia y capacidad de respuesta. Esto podría lograrse mediante la comprensión del dominio del problema y resolviéndolo al mismo tiempo, no secuencialmente. La información necesaria es

recibida por los pequeños lotes, no en una vasta cantidad y con una preferible comunicación cara a cara, sin ninguna documentación por escrito. El flujo de información debe ser constante en ambas direcciones, a partir del cliente a los desarrolladores y viceversa, evitando así la gran y estresante cantidad de información después de un largo periodo de desarrollo en el aislamiento.

**Véase todo el conjunto**

Los defectos en el software tienden a acumularse durante el proceso de desarrollo por medio de la descomposición de las grandes tareas en pequeñas tareas y de la normalización de las diferentes etapas de desarrollo. Las causas reales de los defectos deben ser encontradas y eliminadas.

"Piensa en grande, actúa en pequeño, equivócate rápido; aprende con rapidez" estas consignas resumen la importancia de comprender el terreno y la idoneidad de implementar los principios Lean a lo largo del proceso de desarrollo de software.

1. **KANBAN. Estudia las ventajas e inconvenientes de tener una pizarra web digital para la metodología Kanban. Puedes consultar los siguientes enlaces:**

**Ventajas**

* Funciones de colaboración adicionales, como integración de correo electrónico.
* Informes y métricas más sofisticados que toman mucho menos tiempo para generar que los informes de la junta física.
* La capacidad de integrarse con otros sistemas empresariales, como la gestión de la cartera de proyectos o las plataformas de HelpDesk, para eliminar la entrada de datos duplicados.

**Desventajas**

* Para equipos que tienen una ubicación conjunta del 100% no les interesa ya que cada miembro del equipo pueden interactuar individualmente con la pizarra física con la junta directiva y no tienen que depender de colegas en el sitio para actualizar sus tarjetas por poder.
* Cara

1. **KANBAN. Haz un resumen de la metodología Kanban e indica sus diferencias**

**frente a SCRUM. Puedes consultar el siguiente enlace:**

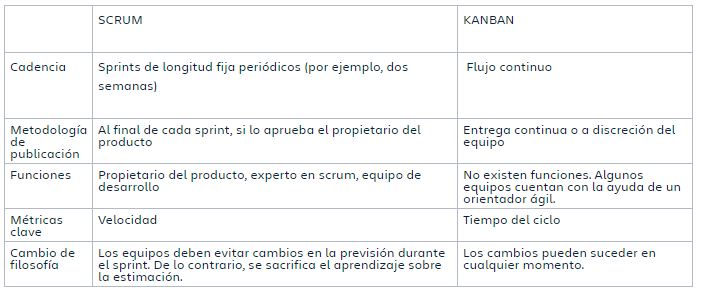
Kanban funciona con una serie de principios básicos con el fin de obtener el máximo rendimiento de su flujo de trabajo:

**Visualice lo que hace (su flujo de trabajo):** una visualización de todas sus tareas y elementos en una tabla contribuirá a que todos los miembros de su equipo se mantengan al corriente con su trabajo.

**Limite la cantidad de Trabajo en Proceso (límites del TEP):** establezca metas asequibles. Mantenga el equilibrio de su flujo de trabajo mediante la limitación de los trabajos en proceso para prevenir el exceso de compromiso en la cantidad de tareas que será incapaz de terminar.

**Realice un seguimiento de su tiempo:** El seguimiento del tiempo confluye con la metodologia Kanban. Realice un seguimiento de su tiempo de forma contínua y evalúe su trabajo con precisión.  
**Lectura fácil de indicadores visuales:** conozca lo que está ocurriendo de un solo vistazo. Utilice tarjetas de colores para distinguir los Tipos de trabajo, Prioridades, Etiquetas, Fechas límite y más.

**SCRUM VS KANBAN**



**Identifique los cuellos de botella y elimine lo que resulta descartable:** aproveche al máximo los plazos y ciclos de ejecución, del Flujo Acumulativo y de los informes de tiempo. Estos criterios le permitirán evaluar su rendimiento, detectar los problemas y ajustar el flujo de trabajo en consecuencia.

1. **SCRUM. Explica como funciona Scrum. Consulta los siguientes enlaces:**

Scrum es un proceso en el que se aplican de manera regular un conjunto de

buenas prácticas para trabajar colaborativamente, en equipo, y obtener el mejor resultado posible de un proyecto. Scrum está especialmente indicado para proyectos en entornos complejos, donde se necesita obtener resultados pronto, donde los requisitos son cambiantes o poco definidos, donde la innovación, la competitividad, la flexibilidad y la productividad son fundamentales.

**PROCESO**

En Scrum un proyecto se ejecuta en ciclos temporales cortos y de duración fija

(iteraciones)que normalmente son de 2 semanas, aunque en algunos equipos son de 3 y hasta 4 semanas, límite máximo de feedback de producto real y reflexión).

Cada iteración tiene que proporcionar un resultado completo, un incremento de producto final que sea susceptible de ser entregado con el mínimo esfuerzo al cliente cuando lo solicite.

El proceso parte de la lista de objetivos/requisitos priorizada del producto, que

actúa como plan del proyecto. En esta lista el cliente (Product Owner) prioriza los objetivos balanceando el valor que le aportan respecto a su coste.

1. **SCRUM. Define los siguientes términos:**

Product backlog: es un listado de todas las tareas que se pretenden hacer durante el desarrollo de un proyecto.

Sprint backlog: es el listado de tareas en el que subdivide las historias de usuario que describen las funcionalidades que componen un proyecto.

1. **SCRUM. En la terminología Scrum qué terminos se utilizan como sinónimo de:**

* Jefe de Proyecto: Scrum Master
* Cliente: Stakeholders
* Equipo de desarrollo: Development Team

1. **SCRUM. Haz un resumen de los requisitos para poder utilizar Scrum. Consulta el siguiente enlace:**

Los requisitos para poder utilizar scrum son:

Cultura de empresa

Debe fomentar:

• El trabajo en equipo y la colaboración entre todas las personas implicadas en un proyecto.

• Equipos autogestionados a los que se ha delegado la responsabilidad y autoridad para hacer su trabajo, en contraposición a la dirección y control de personas que ejercería un jefe de proyecto tradicional.

• La creatividad del equipo.

Scrum sistematiza la identificación de obstáculos que pueden impedir el correcto progreso del proyecto.

**Compromiso del Cliente**

Scrum exige del cliente una alta implicación y una dedicación regular:

• El cliente tiene la responsabilidad de dirigir los resultados del producto o proyecto:

• El cliente debe disponer de una visión de alto nivel del producto o proyecto y tener reflejadas sus expectativas en forma de lista de requisitos priorizada donde ha indicado el valor que le aportará cada uno.

• El cliente replanifica el proyecto en cada iteración para maximizar este ROI de manera continua.

**Compromiso de la Dirección**

La Dirección debe estar comprometida y apoyar el uso de Scrum:

• Se harán muy evidentes los obstáculos ya existentes y por venir que impiden el correcto desarrollo de los proyectos (a nivel de expectativas del cliente, productividad, calidad, etc.)

• Será necesario tomar decisiones, realizar cambios organizativos, alinear a personas y proporcionar recursos para hacer la transición.

▪ Un proyecto ya no consistirá en que cada Departamento/Área/Unidad realice su parte del trabajo y se la pase al siguiente. Será necesario formar equipos

autogestionados y multidisciplinares capaces de conseguir un objetivo por ellos

mismos.

▪ Se tendrá que gestionar aquellas conductas personales que no permiten que otraspersonas puedan aportar ideas sobre el qué y el cómo de un proyecto, que defienden a toda costa su parcela de responsabilidad, que les cuesta mucho cederla al equipo y dejar de controlarlo.

**Compromiso del equipo**

• Scrum se basa en el compromiso conjunto y la colaboración entre los miembros del equipo. La transparencia entre todos es fundamental para poder inspeccionar la situación real del proyecto y así poder hacer las mejores adaptaciones que permitan conseguir el objetivo común. será difícil trabajar utilizando Scrum para las personas que:

• No confían en los demás, no permiten que otras personas puedan aportar ideas sobre el qué y el cómo, no son capaces de colaborar en la resolución de problemas ni de delegar tareas.

• Su modo de relación se basa en la generación de conflicto o bien evitan entrar en conflictos sanos en que ambas partes ganen (ganar/ganar), con lo que los problemas realmente no se resuelven sino que crean conversaciones veladas, de manera que estas personas no acaban de adquirir un compromiso real con el equipo.

• No son capaces de cambiar sus hábitos y salir de su zona de confort, tienen miedo al cambio, prefieren que se les diga qué tienen que hacer o quieren decir a los demás qué tienen que hacer.

**Relación entre proveedor y cliente**

La relación entre el cliente y el proveedor del proyecto debe estar basada en el principio de ganar – ganar.

En lugar de estar ligados por un contrato férreo de alcance, tiempo y coste, las dos partes asumen que habrá cambios para que cliente pueda obtener lo que realmente necesita, no lo que está escrito en un documento inicial o seguir un plan inicial que vaya perdiendo su sentido.

Debe existir transparencia en la ejecución del proyecto para facilitar esta relación.

**Facilidad para realizar cambios en el proyecto**

Para poder utilizar Scrum se debe poder ir incorporando requisitos de manera incremental en el producto del proyecto. Para ello es necesario:

• Disponer de técnicas y herramientas que faciliten el crecimiento incremental y la introducción de cambios.

• Mantener la simplicidad y calidad interna del producto que se está creando. Para cubrirlos requisitos actuales del cliente no hay que realizar más esfuerzo del que sea necesario y, a la vez, se debe vigilar no hacer nada en contra de futuros requisitos.

**Tamaño del equipo**

El tamaño de un equipo está entre 5 y 9 personas. Por debajo de 5 personas cualquier imprevisto o interrupción sobre un miembro del equipo compromete seriamente el compromiso que han adquirido.

De cualquier manera, se puede hacer Scrum con 3 personas y se ha utilizado en proyectos con 250 personas en varios equipos. Cuando es necesario que más de un equipo trabaje de manera ágil en un mismo proyecto, existen diferentes técnicas que permiten esta colaboración, desde el Scrum de Scrums hasta equipos de integración que dedican parte de su tiempo a trabajar con los equipos de desarrollo.

**Equipo Trabajando en un mismo espacio común**

Todos los miembros del equipo trabajan en la misma localización física, para poder maximizar la comunicación entre ellos mediante conversaciones cara a cara, diagramas en pizarras blancas, tarjetas en el tablón de tareas, etc.

De esta manera se minimizan otros canales de comunicación menos eficientes (llamadas telefónicas, correos electrónicos, documentos, etc.), que hacen que las tareas se transformen en un “pasa pelota” o que hacen perder el tiempo en el establecimiento de la comunicación .

**Dedicación del equipo a tiempo completo**

Los miembros del equipo dedicarse al proyecto a tiempo completo para de esta manera:

• Evitar dañar su productividad, que se vería afectada si tuviesen que ir cambiando de tarea para diferentes proyectos o duplicando el número de reuniones para estos diferentes proyectos.

• Facilitar la gestión de recursos humanos de la organización. Esta gestión se

simplifica si en la organización las personas se reservan a un proyecto por iteraciones completas.

**Estabilidad del equipo**

El equipo debe ser estable durante el proyecto, sus miembros deben cambiar lo mínimo posible, para poder aprovechar el esfuerzo que les ha costado construir sus relaciones interpersonales, engranarse y establecer su organización del trabajo.

1. **Explica los 5 valores de la Programación Extrema.**

### **Simplicidad**: Los programas deben ser los más sencillos posibles y tener la funcionalidad necesaria que se indican en los requisitos.

**Comunicación**: Muy importante. La XP ayuda mediante sus prácticas a la comunicación entre los integrantes del grupo de trabajo: jefes de proyecto, clientes y desarrolladores.

### **Retroalimentación (feedback**): Las pruebas que se le realizan al software nos mantiene informados del grado de fiabilidad del sistema.

**Coraje o valentía**: Esto es un esfuerzo para evitar empantanarse en el diseño y requerir demasiado tiempo y trabajo para implementar el resto del proyecto. La valentía le permite a los desarrolladores que se sientan cómodos con reconstruir su código cuando sea necesario.

**Humildad**: Con la compartición de código, la refactorización y el trabajo de equipo tan estrecho una buena dosis de humildad siempre es de agradecer.

1. **¿Cuáles son las características distintivas de XP frente a otras metodologías ágiles? Explícalas. Puedes consultar el siguiente enlace:**

Estas son las características que las diferencia:

* El Juego de la Planificación: Es un permanente diálogo entre las partes empresarial (deseable) y técnica (posible).
* Diseño Sencillo: Cuando implementamos nuevas características en nuestros programas nos planteamos la manera de hacerlo lo mas simple posible, después de implementar esta característica, nos preguntamos como hacer el programa mas simple sin perder funcionalidad, este proceso se le denomina recodificar o refactorizar (refactoring)
* Pruebas: No debe existir ninguna característica en el programa que no haya sido probada, los programadores escriben pruebas para chequear el correcto funcionamiento del programa
* Refactorización: Implementar nuevas características en nuestros programas para que en un futuro acepte nuevos cambios y pueda albergar nuevas características.