Tarea 3
Ciclo de vida del software (Relación 2)

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

Los cuatro principios se pueden resumir de la siguiente manera:

1. **Individuos e interacciones por encima de procesos y herramientas**. Es decir, prima más la comunicación con el cliente y el equipo de desarrollo que en metodologías anteriores, como la de cascada.
2. **Software funcionando por encima de documentación exhaustiva**. En las metodologías ágiles se pretende tener software funcionando desde el primer día, y luego irlo mejorando paso a paso.
3. **Colaboración con el cliente por encima de negociación contractual**. Lo que es lo mismo, el cliente es parte del proyecto y puede ir solicitando cambios y mejoras, aún con el proyecto ya en marcha.
4. **Respuesta ante el cambio por encima de seguir un plan**. Esto es gracias al uso de los sprints, donde al final de cada uno (normalmente cada 2-4 semanas) tiene lugar una reunión en la que se exponen las nuevas implementaciones y se sopesan cambios y mejoras en el software.

2.- ¿Qué es una historia de usuario? Consulta el siguiente enlace:

Una historia de usuario es una representación de un requisito escrito en una o dos frases. Dentro de las metodologías ágiles se usan para especificar requisitos.

Las historias de usuario cumplen con ciertas características como por ejemplo:

– Deben ser independientes unas de otras.

– Valoradas por los clientes y los usuarios, para asegurar la viabilidad de lo que se pretende implementar con la historia.

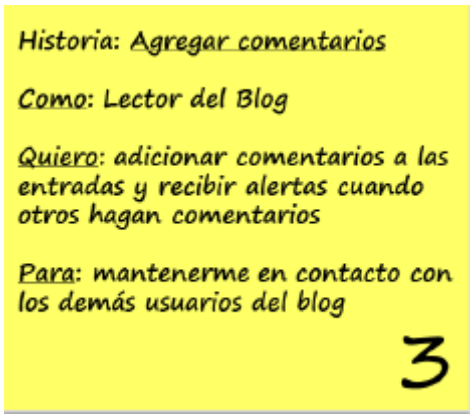
– Estimables, para tener una idea del tiempo que puede tomar realizar la tarea.

Entre otras varias.

Por último, se establece que sin importar la naturaleza de la historia, por lo general, debe responder a estas preguntas: ¿Quién se beneficia?, ¿qué se quiere? y ¿cuál es el beneficio? Tanto es así que incluso existe un “canon” no oficial de cómo se deben redactar las historias, dice así:

Como (rol) quiero (algo) para poder (beneficio).

Como *Vendedor*, quiero *registrar los productos y cantidades que me solicita un cliente* para c*rear un pedido de venta*.



https://es.wikipedia.org/wiki/Historias_de_usuario

3.- Haz un resumen sobre qué se entiende por Lean software y qué principios lo rigen. Consulta el siguiente enlace:

La metodología de desarrollo de software lean es una traducción de los principios y las prácticas del lean manufacturing, un modelo de gestión japonés que se enfoca en minimizar la pérdidas al mismo tiempo que maximiza la creación de valor para el sistema final, aplicado al desarrollo de software. Cabe destacar que lean es un término en inglés que se puede traducir como “eficiente” en este contexto.

El desarrollo de software lean se puede resumir en siete principios, así como el lean manufacturing:

– **Eliminar los desperdicios**: Todo lo que no añade valor al cliente se considera un desperdicio (código y funcionalidades innecesarias, requisitos poco claros, burocracia, etc).

– **Amplificar el aprendizaje**: Mediante reuniones cortas con los clientes se ayuda a determinar la fase actual del desarrollo y se ajustan los esfuerzos para introducir mejoras en el futuro.

– **Decidir lo más tarde posible**: 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 decisiones.

– **Entregar tan rápido como sea posible**: Sin velocidad, las decisiones no pueden ser postergadas. La velocidad asegura el cumplimiento de las necesidades actuales del cliente y no lo que éste requería para ayer.

– **Capacitar al equipo**: Las personas necesitan motivación y un propósito superior para el cual trabajar; un objetivo alcanzable dentro de la realidad con la garantía de que el equipo puede elegir sus propios compromisos.

– **Construir integridad intrínseca**: La información necesaria es recibida por pequeños lotes, con una preferible comunicación cara a cara. Además, el flujo de de información debe ser constante, a partir del cliente a los desarrolladores y viceversa.

– **Véase todo el conjunto**: Los sistemas de software hoy en día no son simplemente la suma de sus partes, sino también el producto de sus interacciones. Sólo cuando todos los principios de Lean se aplican al mismo tiempo hay una base para el éxito en el desarrollo de software.

https://es.wikipedia.org/wiki/Lean_software_development

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

El tener una pizarra web digital en Kanban tiene una serie de **ventajas**:

– **No es alterable por terceros**: Las tarjetas no se pueden desplazar o mover de sitio por error.

– **Independiente del espacio disponible**: No necesita espacios dedicados a la colocación del tablero físico.

– **Mejor colaboración con equipos distribuidos**: Cuando una parte del equipo modifica el estado del ítem, el resto del equipo puede verlo, independientemente de donde estén.

– **Generación de métricas automáticas**: Lo que es un ahorro de tiempo del equipo y un gran aspecto a favor.

– **Confidencialidad**: Si bien la transparencia es deseable, la opacidad puede tener un valor estratégico en ciertos casos.

No obstante, también existen ciertas **desventajas**:

– **Peor visualización**: La pizarra digital está limitada al tamaño de la pantalla, y lo más normal es que haya que hacer scroll para ver muchas de las tarjetas.

– **Menor interacción, colaboración y participación**: Los miembros miran la pantalla, sentados y como mucho, desplazan una tarjeta con el ratón. Esto afecta a la colaboración.

– **Mayor coste de elevación**.

- https://leankit.com/learn/kanban/kanban-board/

- https://trello.com/es
- https://taiga.io/
- https://kanbantool.com/es/

5. KANBAN. Haz un resumen de la metodología Kanban e indica sus diferencias frente a SCRUM. Puedes consultar el siguiente enlace: https://es.atlassian.com/agile/kanban

Kanban es una de las metodologías ágiles más usadas. Requiere una comunicación en tiempo real y total transparencia de trabajo, por otro lado, los elementos de trabajo se representan visualmente en un tablero Kanban (físico o virtual) que permite a los miembros del equipo ver el estado de cada uno en cualquier momento.

Pero, ¿qué es un tablero Kanban?

Un tablero Kanban (una palabra japonesa que significa “señal visual") es una herramienta ágil de gestión de proyectos diseñada para ayudar a visualizar el trabajo, limitar el trabajo en curso y maximizar la eficiencia. Este se divide de cinco componentes:

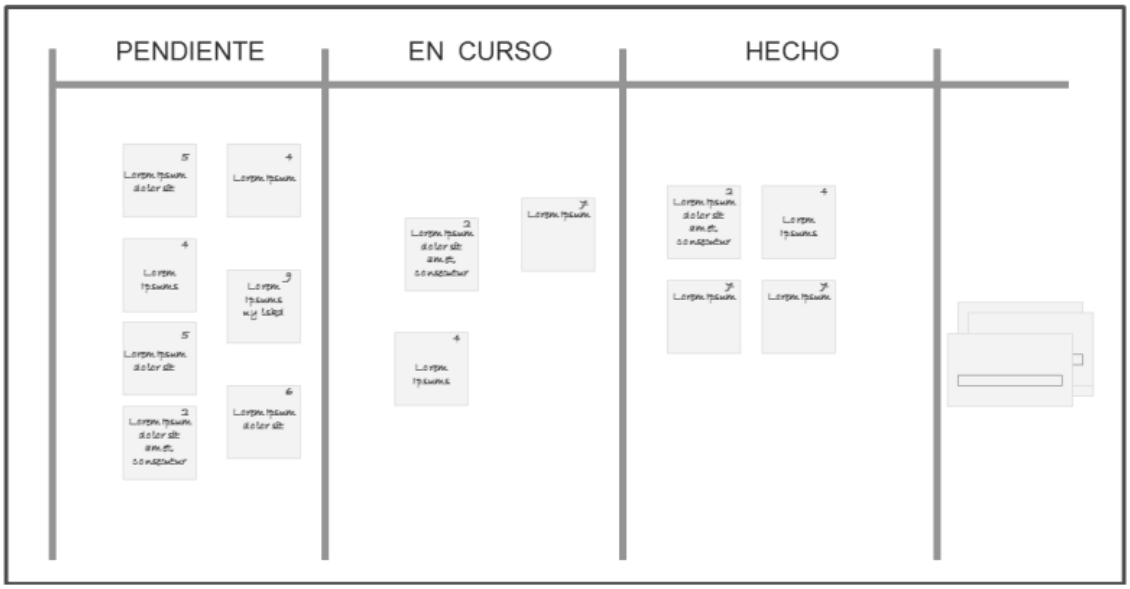
– **Señales visuales**: Tarjetas visuales (adhesivos, tickets, etc), donde se puede encapsular un historia de usuario.

– **Columnas**: Representan los denominados flujos de trabajo, que pueden ser algo como “Por hacer”, “En curso”, o algo más complejo.

– **Límites del trabajo en curso**: Número máximo de tarjetas que puede haber en una columna.

– **Un punto de compromiso**: El momento en que el equipo adopta una idea y comienza a trabajar en el proyecto.

– **Un punto de entrega**: El momento en el que el producto está en manos del cliente.



En lo que concierne a las diferencias de esta metodología con la Scrum, podemos listar algunas de ellas:

– **Su origen**: La metodología Kanban es derivada de la llamada producción lean y adaptada al desarrollo de software. Por su parte, la metodología Scrum fue específicamente ideada para el desarrollo de software.

– **La cadencia**: La metodología Kanban defiende el flujo continuo de trabajo, mientras que en Scrum se trabaja por “Sprints”, de una longitud fija (normalmente entre 2 y 4 semanas).

– **Funciones**: En Kanban no hay miembros con funciones específicas, al contrario que en Scrum, donde sabemos que algunos miembros del proyecto desempeñan las funciones “Product Owner” o “Scrum Master”.

6. SCRUM. Explica cómo 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. Estas prácticas se apoyan unas a otras y su selección tiene origen en un estadio de la manera de trabajar de equipos altamente productivos. Además, está especialmente indicado para proyectos en entornos complejos, donde se necesita obtener resultados pronto o donde los requisitos son cambiantes o pocos definidos.

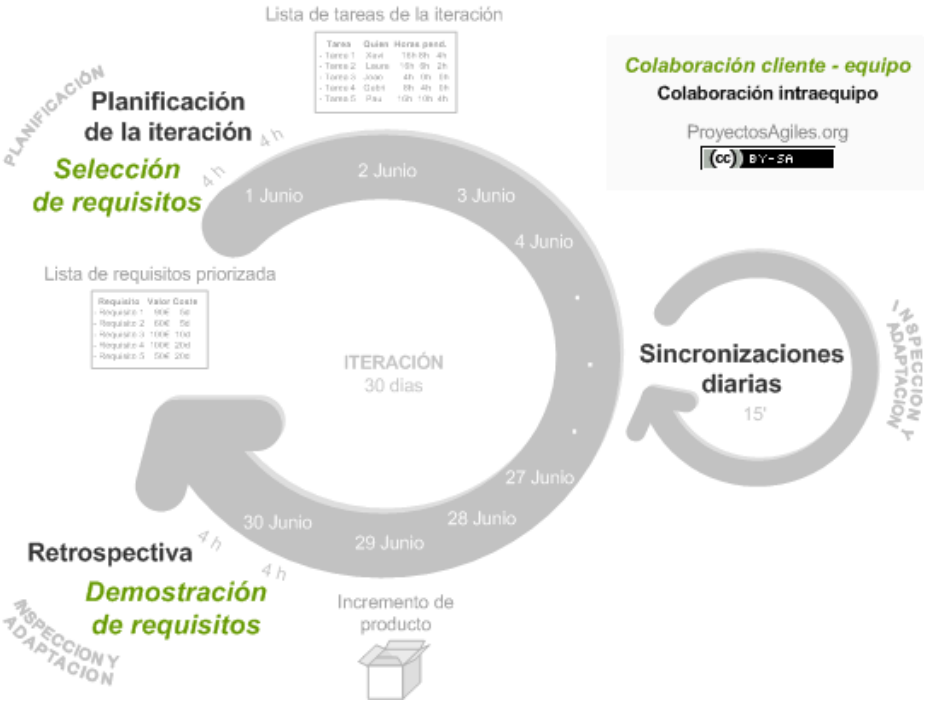
En Scrum un proyecto se ejecuta en ciclos temporales cortos y de duración fija (iteraciones de entre 2 y 4 semanas). Cada iteración tiene que proporcionar un resultado completo, un incremento de producto final que sea susceptible de ser entregado con el mínimo esfuerzo al cliente cuando lo solicite.

El primer día de la iteración se realiza la reunión de planificación de la iteración. Tiene dos partes:

– **Selección de requisitos**: El cliente presenta al equipo la lista de requisitos priorizada del producto o proyecto.

– **Planificación de la iteración**: El equipo elabora la lista de tareas de la iteración necesarias para desarrollar los requisitos seleccionados.

Además, cada día el equipo realiza una reunión de sincronización (15 minutos), normalmente delante de un tablero físico o pizarra (Scrum Taskboard). El equipo inspecciona el trabajo que el resto está realizando para poder hacer las adaptaciones necesarias que permitan cumplir con la previsión de objetivos a mostrar al final de la iteración.



- https://proyectosagiles.org/que-es-scrum/
- https://proyectosagiles.org/como-funciona-scrum/

7. SCRUM. Define los siguientes términos:

- **Product backlog**.

  Product backlog, o lista priorizada de objetivos/requisitos, representa la visión y expectativas del cliente respecto a los objetivos y entregas del producto o proyecto.

  – Los objetivos/requisitos se suelen expresar en forma de historias de usuario. Para cada uno de estos se indica el valor que aporta al cliente y el costo estimado de completarlo.

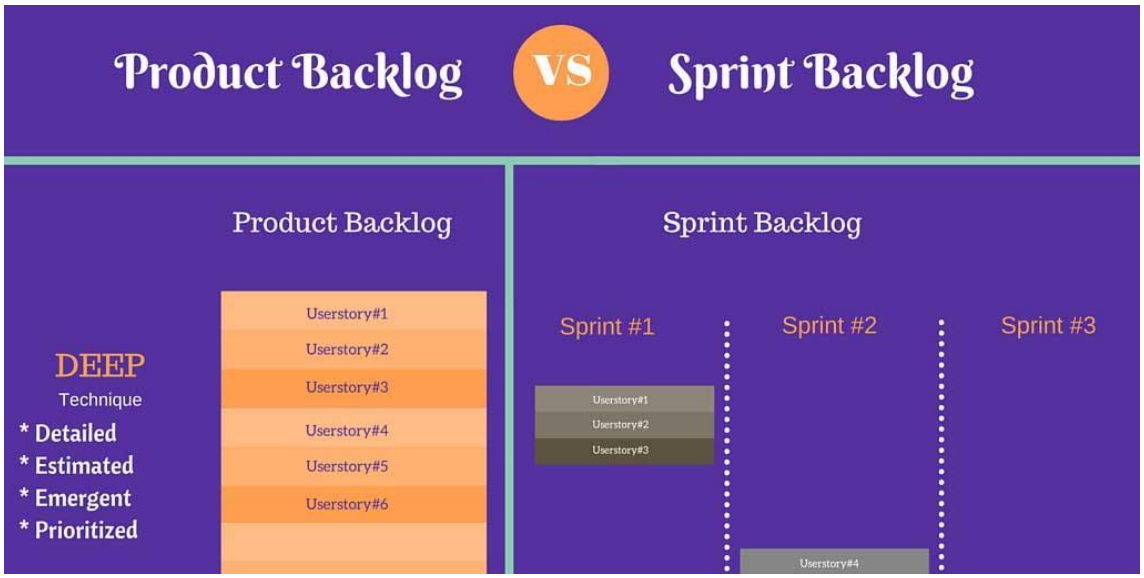
  – En la lista se indican las posibles y las entregas esperadas por el cliente, en función de la velocidad de desarrollo del equipo que trabajará en el proyecto.

– También tiene que considerar los riesgos del proyecto e incluir los requisitos o tareas necesarios para mitigarlos.

- **Sprint backlog**.

  La lista de tareas de la iteración (Sprint Backlog) es un subconjunto de objetivos/requisitos del Product Backlog seleccionado para la iteración actual y su plan de tareas de desarrollo. Con eso, podemos decir que el Sprint Backlog es una planificación táctica del trabajo a realizar en la iteración actual.

  Esta lista permite ver las tareas donde el equipo está teniendo problemas y no avanza, con lo que le permite tomar decisiones al respecto.



8. SCRUM. En la terminología Scrum qué términos se utilizan como sinónimo de:

- **Jefe de proyecto**.

  En Scrum, se refiere al jefe de proyecto como Scrum Master. Este se encarga de conseguir que el equipo conozca y sienta los principios y valores Agile, así como

la teoría y prácticas de Scrum. De este modo, es el coach y líder al servicio del equipo.

- **Cliente**.

  En Scrum se usa el término Product Owner para referirse al cliente. Su misión principal es encargarse de que exista una priorización clara de los objetivos a conseguir. Así, sus responsabilidades son:

  – Conocer el mercado y los comportamientos de los clientes/usuarios finales.

  – Ser el representante de todas las personas interesadas.

- **Equipo de desarrollo**.

  Cuando se habla de Development Team, o Equipo de desarrollo, nos referimos al conjunto de personas más técnicas que de manera conjunta desarrollan el producto del proyecto. Estas personas incluyen a los mencionados Product Owner y Scrum Master, entre otros.

  Tienen un objetivo común, comparten la responsabilidad del trabajo que realizan (así como de su calidad) en cada iteración y en el proyecto.

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

Como hemos mencionado anteriormente, el Scrum Master es el encargado de que se sigan los principios de Scrum dentro del proyecto, pero, ¿cuáles son esos requisitos? Los podemos listar de la siguiente manera:

– **Cultura de empresa** basada en trabajo en equipo, delegación, creatividad y mejora continua.

– **Compromiso del cliente** en la dirección de los resultados del proyecto, gestión de ROI, etc.

– **Compromiso de la Dirección** de la organización para resolver problemas endémicos y realizar cambios organizativos.

– **Compromiso conjunto y colaboración** de los miembros del equipo

– **Relación entre proveedor y cliente** basada en ganar-ganar, colaboración y transparencia.

– **Facilidad para realizar cambios** en el proyecto

– **Tamaño de cada equipo entre 5 y 9 personas**.

– **Equipo trabajando en un mismo espacio común** para maximizar la comunicación.

– **Dedicación del equipo** a tiempo completo.

– **Estabilidad de los miembros del equipo**.

https://proyectosagiles.org/requisitos-de-scrum/

10. XP. Explica los 5 valores de la Programación Extrema.

La programación extrema es una metodología ágil de gestión de proyectos que se centra en la velocidad y la simplicidad con ciclos de desarrollo cortos y con menos documentación. La estructura del proceso está determinada por 5 valores fundamentales, 5 reglas y 12 prácticas de XP.

Los 5 valores de la Programación Extrema:

– **Simplicidad**: En la programación extrema, tu atención se centra en realizar primero el trabajo más importante. Esto significa que debes buscar un proyecto simple que sabes que puedes lograr.

“¿Cuál es el proceso más simple y que también funciona?”

– **Comunicación**: Para trabajar de manera efectiva, el equipo debe ser abierto y honesto entre sí. Cuando surgen problemas, se espera que todos aporten sus comentarios e ideas, ya que probablemente alguno de ellos ya tenga una solución adecuada.

– **Retroalimentación/Comentarios**: El enfoque de XP es producir trabajo de forma rápida y sencilla, para luego compartir los resultados para obtener comentarios de forma casi inmediata. Por eso, los desarrolladores están en contacto casi constante con los clientes durante todo el proceso.

– **Valentía**: Siempre se espera que seas honesto al brindar actualizaciones al equipo sobre tu progreso. Si no cumples con una fecha de entrega en la programación extrema, es probable que al líder de tu equipo no le interese analizar los motivos. En cambio, le dirías que no cumpliste con la fecha de entrega, te responsabilizarías por ello y te pondrías a trabajar nuevamente.

– **Respeto**: Para la programación extrema, estas son las expectativas:

→ Respeto mutuo entre los clientes y el equipo de desarrollo.

→ Respeto mutuo entre los miembros del equipo.

→ El reconocimiento de que todos en el equipo aportan algo valioso al proyecto.

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

La programación extrema implementa un conjunto de 12 prácticas a lo largo del proceso. Se basan en el Manifiesto Ágil, pero se adaptan a las necesidades de la programación extrema:

1. **El juego de planificación**: La planificación XP se usa para guiar el trabajo.

2. **Prueba de clientes**: Al finalizar una función, el cliente desarrolla una prueba de aceptación para determinar si has cumplido con la historia de usuario.

3. **Pequeñas entregas**: La programación extrema realiza entregas pequeñas y periódicas para obtener información durante todo el proceso.

4. **Diseño simple**: El sistema XP está diseñado para ser simple, producirá solo lo necesario y nada más.

5. **Programación en parejas**: Toda la programación la realizan simultáneamente dos desarrolladores que se sientan físicamente uno al lado del otro.

6. **Desarrollo guiado por pruebas (TDD)**: A través de ciclos cortos, los programadores realizan pruebas automatizadas para luego reaccionar de inmediato.

7. **Refactorización**: Observar los detalles más finos del código base, para eliminar los duplicados y asegurarse de que el código sea coherente.

8. **Propiedad colectiva**: Cualquier par de desarrolladores puede modificar el código en cualquier momento, independientemente de que lo hayan desarrollado o no.

9. **Integración continua**: Los equipos de XP no esperan a que se completen las iteraciones, sino que se integran constantemente.

10. **Ritmo de trabajo sostenible**: Los equipos deben determinar cuánto trabajo pueden producir a este ritmo por día y por semana, y usarlo para establecer plazos de trabajo.

11. **Metáfora**: La metáfora es, literalmente, una metáfora. Se decide en equipo y se usa un lenguaje para expresar cómo debe funcionar el equipo.

12. **Estándares de codificación**: Los equipos de XP siguen un estándar, así los desarrolladores deben codificar de la misma manera unificada para que parezca que el código está escrito por un solo desarrollador.

http://www.davidvalverde.com/blog/introduccion-a-la-programacion-extrema-xp/