**Unidad 2 – Filosofía Agile**

User Stories son una técnica ágil para especificar los requerimientos de usuario, mientras que en el enfoque de procesos se especifican mediante la técnica de los casos de uso.

**Desarrollo Ágil de Software**

Se hizo una mala implementación de los procesos definidos en el desarrollo de software. Tenía muchas tareas y actividades de más, que no agregaban valor y que generaban muchos problemas de costos, tiempos, recursos, entre otros.

Debido a esto surgió el concepto de agilismo para lograr un equilibrio entre nada de proceso y demasiado proceso.

Los enfoques ágiles se basan en los procesos empíricos, es decir, se basan en la experiencia del equipo, en ambientes de alta incertidumbre, solo se puede confiar en experiencias previas, y en base a las mismas puedo tomar decisiones.

Existe un patrón que me permite obtener conocimiento realizando procesos empíricos. El patrón es:

1. Asumir una hipótesis sobre un producto
2. Construir el producto (un incremento)
3. Obtener feedback
4. Revisar el producto basándonos en el feedback
5. Adaptar el producto
6. Repetir

Ciclos cortos de inspección y adaptación son los que me permiten generar la experiencia

Los procesos empíricos se basan en los tres pilares del empirismo, inspección, adaptación y transparencia. Este último hace referencia a que todos como equipo vamos a comprender que lo que generamos es de todos, y no debemos guardarnos u ocultar información.

**Valores Ágiles**

Existen 4 valores ágiles, los cuales podemos resumir como:

* Privilegiamos individuos e interacciones sobre procesos y herramientas.
* Privilegiamos software funcionando sobre documentación extensiva.
* Privilegiamos colaborar con el cliente sobre negociación contractual
* Privilegiamos responder al cambio sobre seguir un plan

**Principios Ágiles**

Los 4 valores se especializan en 12 principios. Hace foco en lo que ya se plantea con los valores.

1. Nuestra mayor prioridad es ***satisfacer al cliente*.**
2. Entender y aceptar que los requerimientos pueden ***cambiar***, incluso en etapas tardías.
3. Entregar software funcional frecuentemente. Lo que nosotros construimos se debe poder poner en producción (debe estar condiciones de hacer esto).
4. El PO tiene que pertenecer al negocio. Es decir, el PO y los desarrolladores deben ***trabajar juntos*** durante el proyecto. El PO debe identificar y escuchar a los involucrados para presentar una visión unificada.
5. Los individuos que desarrollan proyectos deben estar ***motivados***. Es una de las características que deben tener los individuos.
6. El mejor medio de comunicación es conversaciones ***cara a cara***.
7. La mejor forma de mostrar cómo vamos es el ***software funcionado*** (como medida de éxito)
8. El equipo trabajando a lo largo de un tiempo, logra realizar el ***desarrollo sostenible***. Es decir, es posible de prever.
9. La ***calidad*** del producto no se negocia. Me voy a ocupar todo el tiempo a la excelencia técnica y al buen diseño. La calidad es algo que debo atender durante todo el proceso.
10. La mejor solución es la más ***sencilla***. Se entiende simplicidad como la maximización del trabajo no hecho. No hagamos cosas de más.
11. Los mejores proyectos emergen de ***equipos auto-organizados.*** Implica darle poder al equipo. En equipos tradicionales hay líderes que junto con analistas definen el producto y dan una bajada de línea. En equipos agiles esto no se recibe como imposición, sino que emerge del equipo.
12. A intervalos regulares el equipo ***reflexiona*** sobre cómo ser más efectivo, y ***ajustan su comportamiento***. En el momento de reflexión debemos lograr la adaptación. Buscamos oportunidades de mejora. Si no hacemos esto, no hay aprendizaje.

**¿Qué es Ágil?**

No es una metodología o un proceso. Es una ideología con un conjunto definido de principios que guían el desarrollo del producto. Casi todos los frameworks agiles son de gestión, definen cómo se organiza el proyecto, pero no especifican cómo desarrollar software (excepto extreme programming).

Conocer frameworks ágiles no te permite hacer software, sino que necesitas más conocimientos de diseño, de ingeniería, de programación.

**Valores de los equipos ágiles**

* Planificación continua multinivel
* Facultados, auto-organizados, equipos completos.
* Entregas frecuentes, iterativas y priorizadas.
* Prácticas de ingeniería disciplinadas
* Integración Continua
* Testing Concurrente

**¿Qué significa Ágil?**

Ágil es cultura, es filosofía, son valores y principios. Es un cambio cultural que se debe sustentar con un cambio organizacional (estructuras más planas).

Implica un balance entre ningún proceso y demasiado proceso. La diferencia inmediata es la exigencia de una menor cantidad de documentación, sin embargo, no es lo más importante, sino:

* Los métodos ágiles son adaptables en lugar de predictivos.
* Los métodos ágiles son orientados a la gente en lugar de orientados al proceso

Hay varios frameworks ágiles, como, por ejemplo:

* ATDD (Acceptance Test Driven Development)
* Scrum
* XP (Extreme Programming)
* Crystal
* FDD (Feature Driven Development)

**Técnicas Efectivas Agile**

* Daily Standup
* Backlogs piorizados
* Iteraciones cortas
* Retrospectivas
* Planeación de Iteración

**¿Cuándo es aplicable Agile?**

Diagrama

Descripción generada automáticamente Agile da mejores resultados cuando los problemas a ser resueltos caen dentro del espacio de lo complejo, es decir, cuando hay una incertidumbre media, y no hay mucho acuerdo.

El desarrollo de nuevos productos tiende a estar en dicho espacio. En cambio, la investigación está dentro del espacio de lo anárquico, porque hay una falta total de acuerdo y una incertidumbre elevada.

Mientras que el mantenimiento, casi siempre cae dentro de lo simple.

En los proyectos tradicionales existe un plan grande y definido de antemano, y también existe una gran arquitectura establecida con anterioridad. Sin embargo, ser Agile no es ser indisciplinado, hacer agile no significa que no existe un plan o arquitectura. Sino que buscamos tener un plan y una arquitectura más áspera, pero adaptativa.

**El triángulo de Hierro vs el triángulo ágil**

Forma

Descripción generada automáticamente

Forma

Descripción generada automáticamenteForma

Descripción generada automáticamente

**U2 – Requerimientos Ágiles**

Implica determinar cómo interpretamos y gestionamos los requerimientos de manera ágil.

¿Cómo entregamos valor al negocio? Haciendo un producto de software que ayude y satisfaga al cliente, construyendo el producto correcto.

Usamos las User Stories para mostrar que construir.

También debemos determinar hasta dónde es suficiente. No todas las funcionalidades que tiene un producto se usan, el 80% casi ni se usa. Por lo que lo más importante es identificar el 20% restante y empezar por ahí

**Gestión Ágil de Requerimientos de Software**

Los requisitos cambiantes son una ventaja competitiva si se puede actuar sobre ellos. Existe un product backlog que es una lista priorizada de características del producto, y es propiedad y responsabilidad del Product Owner. Los requerimientos están ordenados de mayor a menor prioridad.

Cada iteración implementa los requerimientos de prioridad alta. Cada nuevo requerimiento es priorizado y agregado a la pila, y pueden ser movidos en cualquier momento.

**Just in Time**

El enfoque surge en la industria automotriz para ahorrar costos. Se evita lo que se conoce como “Upfront Specifitaction”, es decir, se evita especificar todos los requerimientos al inicio del proyecto, sino que solo vamos a analizar los requerimientos que vamos a consumir en la iteración actual.

Empiezo a construir el producto sin tener todos los requerimientos completos. Nunca el Product Backlog debe ser como la ERS.

**Comunicación**

Es necesario el cara a cara, debemos reemplazar la especificación de requerimientos por charlas cara a cara con el Product Owner. Ya que esta es la forma más efectiva de comunicación entre dos personas. El PO debe ser parte del equipo

**Triángulo Tradicional Vs Ágil**

Diagrama

Descripción generada automáticamenteLa negociación está en términos de cuánto software te puedo entregar en este tiempo y con estos recursos en esta iteración.

Tenemos ciertas cosas fijas, que no pueden cambiar, y en lo tradicional, los requisitos se trataban de fijar, en cambio en agile se abraza el cambio, por lo que son negociables. Lo que no es negociable son los recursos y tiempo dedicado en cada iteración.

**Tipos de Requerimientos**

* Requerimientos de negocio
* Requerimientos de usuario: Las user stories son requerimientos de usuario
* Requerimientos funcionales
* Requerimientos no funcionales
* Requerimientos de implementación

Diagrama

Descripción generada automáticamenteLos requerimientos de software están dentro del dominio de la solución, mientras que los de negocio y de usuario están dentro del dominio del problema.

En resumen, entendiendo la necesidad y el negocio, descubriendo la solución en forma colaborativa (trabajando just in time), junto a un equipo motivado y competente, entregamos valor a los stakeholders de manera frecuente. Siempre debemos apuntar a entregar valor al negocio.

**Principios ágiles relacionados a los requerimientos ágiles**

1. La prioridad es satisfacer al cliente a través de releases frecuentes.
2. Recibir cambios de requerimiento, aun en etapas finales.

**4**. Técnicos y no técnicos trabajando juntos todo el proyecto (presencia del PO)

**6.** El medio de comunicación por excelencia es cara a cara

**11**. Las mejores arquitecturas, diseños y requerimientos emergen de equipos auto-organizados.