Resumen: Capitulo 3

Nombre: Marco Rodríguez Vargas

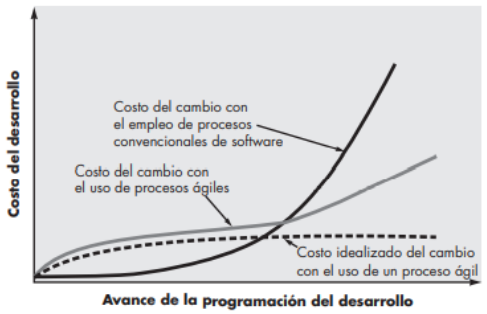
Carnet: 2022149445

En 2001, Kent Beck y 16 desarrolladores de software, escritores y consultores (“Alianza Ágil”) firmaron el “Manifiesto por el **desarrollo ágil** de software”. 1. Los individuos y sus interacciones, sobre los procesos y las herramientas. 2. El software que funciona, más que la documentación exhaustiva 3. La colaboración con el cliente, y no tanto la negociación del contrato 4. Responder al cambio, mejor que apegarse a un plan

El **desarrollo ágil** proporciona beneficios, pero no es aplicable a todo. Para la economía moderna es difícil-imposible de predecir porque estamos en un sistema basado en computadora. El mercado cambia con rapidez, las necesidades de los usuarios se transforman y emergen nuevas amenazas. En muchas situaciones no será posible definir los requerimientos antes de que el proyecto comience. La fluidez implica cambio, el cual es caro, más cuando es descontrolado o se administra mal. El enfoque ágil es reducir los costos del cambio durante el proceso del software. Para funcionar, los modelos de proceso deben proveer un mecanismo realista que estimule la disciplina, o deben caracterizarse por la “tolerancia” con los que hacen el trabajo de ingeniería de software. La tolerancia es más fácil de adoptar, pero menos productiva

Que es la agilidad: Estructuras y actitudes que faciliten la comunicación, entrega rápida de software funcional y resta importancia a los productos intermedios; adopta al cliente como parte del equipo de desarrollo; reconoce que la planeación en un mundo incierto tiene sus límites y que un plan debe ser flexible. Es esencial que se diseñe en forma que permita al equipo adaptar las tareas, ejecutar la planeación de manera que entienda la fluidez de un enfoque ágil del desarrollo, eliminar todos los productos excepto los más esenciales y mantenerlos, estrategia de entrega incremental que haga el software tan rápido como sea posible para el cliente, según el tipo de producto y el ambiente de operación.

La agilidad y el costo del cambio: Un proceso ágil bien diseñado “aplana” el costo de la curva de cambio, lo que permite que el equipo haga cambios en una fase tardía de un proyecto sin que haya un efecto notable en el costo y en el tiempo. Cuando ésta se acopla con otras prácticas ágiles, como las pruebas unitarias continuas y la programación por parejas, el costo de hacer un cambio disminuye. Aunque hay debate sobre el grado en el que se aplana la curva de costo, existen evidencias que sugieren que es posible lograr una reducción significativa del costo.



Que es un proceso ágil: 1. Es difícil predecir qué requerimientos de software persistirán y cuáles cambiarán. También es difícil pronosticar cómo cambiarán las prioridades del cliente a medida que avanza el proyecto. 2. Para muchos tipos de software, el diseño y la construcción están imbricados. Es decir, ambas actividades deben ejecutarse en forma simultánea, de modo que los modelos de diseño se prueben a medida que se crean. Es difícil predecir cuánto diseño se necesita antes de que se use la construcción para probar el diseño. 3. El análisis, el diseño, la construcción y las pruebas no son tan predecibles como nos gustaría.

Un proceso que pueda manejar la impredecibilidad está en la adaptabilidad del proceso (cambio rápido del proyecto y condiciones). Un proceso ágil debe ser **adaptable** incrementalmente. Un equipo ágil requiere retroalimentación con el cliente para hacer las adaptaciones apropiadas. Deben entregarse incrementos del software en periodos cortos de tiempo, de para que la adaptación vaya a ritmo con el cambio. Esto permite que el cliente evalúe de forma regular el incremento del software, de la retroalimentación necesaria e influya en las adaptaciones necesarias.

Principios de agilidad: 1. La prioridad más alta es satisfacer al cliente a través de la entrega pronta y continua de software valioso. 2. Son bienvenidos los requerimientos cambiantes, aun en una etapa avanzada del desarrollo. Los procesos ágiles dominan el cambio para provecho de la ventaja competitiva del cliente.3. Entregar con frecuencia software que funcione, de dos semanas a un par de meses, de preferencia lo más pronto posible. 4. Las personas de negocios y los desarrolladores deben trabajar juntos, a diario y durante todo el proyecto. 5. Hay que desarrollar los proyectos con individuos motivados. Debe darse a éstos el ambiente el apoyo que necesiten, y confiar en que harán el trabajo. 6. El método más eficiente y eficaz para transmitir información a los integrantes de un equipo de desarrollo, y entre éstos, es la conversación cara a cara. 7. La medida principal de avance es el software que funciona. 8. Los procesos ágiles promueven el desarrollo sostenible. Los patrocinadores, desarrolladores y usuarios deben poder mantener un ritmo constante en forma indefinida. 9. La atención continua a la excelencia técnica y el buen diseño mejora la agilidad. 10. Es esencial la simplicidad: el arte de maximizar la cantidad de trabajo no realizado. 11. Las mejores arquitecturas, requerimientos y diseños surgen de los equipos con organización propia. 12. El equipo reflexiona a intervalos regulares sobre cómo ser más eficaz, para después afinar y ajustar su comportamiento en consecuencia.

Factores humanos: Si los miembros del equipo de software son los que van a generar las el proceso de software, entre ellos debe existir características clave, mismas que debe compartir el equipo ágil como tal: **Competencia**: incluye el talento innato, las habilidades relacionadas con el software y el conocimiento general del proceso que el equipo haya elegido aplicar. La habilidad y el conocimiento del proceso pueden y deben considerarse para todas las personas que sean miembros ágiles del equipo**. Enfoque común:** Aunque los miembros del equipo ágil realicen diferentes tareas y aporten habilidades distintas al proyecto, todos deben centrarse en entregar al cliente en la fecha prometida un incremento de software que funcione. Para lograrlo, el equipo también se centrará en adaptaciones continuas que hagan que el proceso se ajuste a las necesidades del equipo. **Colaboración**: Para efectuar estas tareas, los miembros del equipo deben colaborar, entre sí y con todos los participantes. **Habilidad para tomar decisiones**: autonomía en el equipo que entregue autoridad para tomar decisiones sobre asuntos tanto técnicos como del proyecto. **Capacidad para resolver problemas difusos**: el equipo debe aceptar el hecho de que el problema que resuelven ahora tal vez no sea el que se necesite resolver mañana. Sin embargo, las lecciones aprendidas de cualquier actividad relacionada con la solución de problemas (incluso aquellas que resuelven el problema equivocado) serán benéficas para el equipo en una etapa posterior del proyecto. **Confianza y respeto mutuo**: Un equipo pegado tiene la confianza y respeto que son necesarios para hacer “su tejido tan fuerte que el todo es más que la suma de sus partes” **Organización propia**: 1) el equipo ágil se organiza a sí mismo para hacer el trabajo, 2) el equipo organiza el proceso que se adapte mejor a su ambiente, 3) el equipo organiza la programación del

trabajo a fin de que se logre del mejor modo posible la entrega del incremento de software. Sirve para mejorar la colaboración y elevar la moral del equipo. El equipo sirve como su propio gerente.

Programación extrema: Valores que establecen el fundamento para todo trabajo realizado como parte de XP: comunicación, simplicidad, retroalimentación, valentía y respeto. XP pone el énfasis en la colaboración estrecha pero informal (verbal) entre los clientes y los desarrolladores, en el establecimiento de metáforas para comunicar conceptos importantes, en la retroalimentación continua y en evitar la documentación como medio de comunicación. Para alcanzar la simplicidad, XP restringe a los desarrolladores para que diseñen sólo para las necesidades inmediatas. Si hay que mejorar el diseño, se rediseñará en un momento posterior. La retroalimentación se obtiene de tres fuentes: el software implementado, el cliente y otros miembros del equipo de software. Al diseñar e implementar una estrategia de pruebas eficaz, el software (por medio de los resultados de las pruebas) da retroalimentación al equipo ágil. XP usa la prueba unitaria como su táctica principal de pruebas. A medida que se desarrolla cada clase, el equipo implementa una prueba unitaria para ejecutar cada operación de acuerdo con su funcionalidad especificada.

El proceso XP: **Planeación**: permite que los miembros entiendan el contexto del negocio para el software y adquieran la sensibilidad de la salida y características y funcionalidad que se requieren. **Diseño**: mantenerlo sencillo. **Codificación**: Pruebas unitarias para luego empezar a programar, se usa la programación por parejas. XP recomienda que dos personas trabajen juntas con el fin de crear código. Es frecuente que dos cabezas piensen más que una y el código se revisa conforme se crea. Mantiene a los desarrolladores centrados en el problema. Cada persona adopta un papel un poco diferente. **Pruebas:** se centran en las características y funcionalidad del sistema que son visibles y revisables por parte del cliente.

XP industrial: inclusión de la gerencia, el papel más amplio de los clientes y prácticas técnicas actualizadas. Garantiza que un proyecto XP tenga éxito dentro de una organización grande. **Evaluación de factibilidad**: si es posible desarrollarlo. **Comunidad del proyecto**: sugiere que se utilice personal apropiado para formar el equipo. **Calificación del proyecto**: evalúa el proyecto para determinar si existe una justificación apropiada de negocios y si el proyecto cumplirá las metas y objetivos generales de la organización. **Administración orientada a pruebas**: establece una serie de “destinos” medibles y luego define los mecanismos para determinar si se han alcanzado o no éstos. **Retrospectivas**: examina “los temas, eventos y lecciones aprendidas”. **Aprendizaje continuo**: Aprender nuevos métodos y técnicas que conduzcan a una calidad más alta del producto.

El debate de XP: Depender de XP significa tanto fortaleza como debilidad. **Volatilidad de los requerimientos:** Cuando se solicitan cambios de manera informal el proyecto cambia y el trabajo inicial se modifica para dar acomodo a las nuevas necesidades. **Necesidades conflictivas del cliente:** el equipo mismo tiene la tarea de asimilar las necesidades de distintos clientes, trabajo que tal vez esté más allá del alcance de su autoridad. **Los requerimientos se expresan informalmente:** Las historias de usuario y las pruebas de aceptación son la única manifestación explícita de los requerimientos. **Falta de un diseño formal**: XP desalienta la necesidad del diseño de la arquitectura y, en muchas instancias, sugiere que el diseño de todas las clases debe ser relativamente informal.

Desarrollo adaptativo de software (DAS): centrada en la colaboración humana y en la organización propia del equipo. **Especulación**: se inicia el proyecto y la planeación adaptativa del ciclo. **Colaboración**: Recabar requerimientos. **Aprendizaje**: Recibir retroalimentación. Para posteriormente ser entregado. **Scrum**: guiar actividades de desarrollo dentro de un proceso de análisis. **Retraso**: lista de prioridades de los requerimientos o características del proyecto que dan al cliente un valor del negocio. **Sprints**: consiste en unidades de trabajo que se necesitan para alcanzar un requerimiento definido en el retraso que debe ajustarse en una caja de tiempo14 predefinida (lo común son 30 días). **Reuniones Scrum**: son reuniones breves (de 15 minutos, por lo general) que el equipo Scrum efectúa a diario. **Método de desarrollo de sistemas dinámicos (MDSD):** sigue la regla de 80 por ciento. Es decir, se requiere sólo suficiente trabajo para cada incremento con objeto de facilitar el paso al siguiente. Los detalles restantes se terminan más tarde, cuando se conocen los requerimientos del negocio y se han pedido y efectuado cambios. **Cristal:** El objetivo es permitir que equipos ágiles seleccionen al miembro de la familia Cristal más apropiado para su proyecto y ambiente. **Desarrollo impulsado por las características (DIC):** pone el énfasis en las actividades de aseguramiento de la calidad del software mediante el estímulo de la estrategia de desarrollo incremental, el uso de inspecciones del diseño y del código, la aplicación de auditorías de aseguramiento de la calidad del software, el conjunto de mediciones y el uso de patrones (para el análisis, diseño y construcción). **Desarrollo esbelto de software (DES):** eliminar el desperdicio, generar calidad, crear conocimiento, aplazar el compromiso, entregar rápido, respetar a las personas y optimizar al todo. **Modelado ágil (MA):** Conjunto de valores, principios y prácticas para hacer modelos de software aplicables de manera eficaz y ligera a un proyecto de desarrollo de software. Los modelos ágiles son más eficaces que los tradicionales porque son sólo buenos, sin pretender ser perfectos **El proceso unificado ágil (PUA):** revestimiento en serie que permite que el equipo visualice el flujo general del proceso de un proyecto de software.

Conjunto de herramientas para el proceso ágil: Las herramientas deben verse como algo complementario mas no esencial.

1. ¿Cuál es el problema que plantea el capítulo?

En el capítulo se habla sobre el desarrollo digital y como se puede aplicar

2. ¿Por qué el problema es interesante o importante?

Es interesante porque se hablan de metodologías poco convencionales

3. ¿Cuál es la solución propuesta por el autor?

La solución propuesta es implementar acciones inusuales que podrían funcionar en el desarrollo de software

4. ¿Qué tan exitosa es esta solución?

Es exitosa siempre y cuando se sigan las reglas estipuladas y se aplique en una situación donde la solución sea posible.