Dinámica de Manifiesto Ágil

Integrantes: Velez, Matias – Konicoff, Sebastian – Caparroz, Ezequiel – Brito,Carolina – Zago, Agustin – Hadad, Agustin – Daggoto, Florencia – Donalisio, Juan Pablo

UTN - FRC – Departamento de sistemas–Ingeniería de Software- 4K4 -GRUPO 4- 2021

Individuos e interacciones por sobre procesos y herramientas

La prioridad es satisfacer al cliente a través de releases tempranos y frecuentes

La participación del cliente se hace más productiva a medida en que el software está siendo probado, revisado y aprobado constantemente por quien lo requiere y lo va a usar. De esta forma podemos dejar a un lado la desconfianza o insatisfacción producidas en el cliente.

Técnicos y no técnicos trabajando juntos TODO el proyecto

La importancia de las personas cumplen un rol importante, considerando que somos un solo equipo y que tenemos un mismo objetivo. Hay que trabajar en conjunto con el cliente y tomarlo como parte de nuestro equipo de trabajo para obtener mejores resultados, involucrarlo en el proyecto.

Hacer proyectos con individuos motivados

Mantener a los individuos, tanto técnicos como clientes, contentos, cómodos y motivados nos permite tener una mejor productividad, obteniendo mejores resultados.

El medio de comunicación por excelencia es cara a cara

El equipo de trabajo debe apoyarse con un buen sistema de comunicación, tanto entre los miembros del equipo de desarrollo como entre estos y el cliente. La mejor forma de hacer esto es hablando personalmente en cuyo caso se evitan intermediarios en el proceso de comunicación.

Las mejores arquitecturas, diseños y requerimientos emergen de equipos autoorganizados.

Confiar en que la gente y el equipo de trabajo se pueden autoorganizar, dándoles autonomía a ellos les da confianza para trabajar y genera mejores resultados que siguiendo un plan definido.

A intervalos regulares, el equipo evalúa su desempeño y ajusta la manera de trabajar.

El equipo de trabajo está todo el tiempo dispuesto a cambiar lo que sea necesario para mejorar. De esta forma resulta mejor debido a que existe la posibilidad de hacerlo de una manera más óptima la próxima vez.

Algunos ejemplos:

Valor: Individuos e interacciones por sobre procesos y herramientas

Principio: El medio de comunicación por excelencia es cara a cara

En **Facebook** tienen una rutina que suelen llevar a cabo. En vez de coordinar el proyecto por mails, chats o hacer una reunión a la semana para establecer quién hace qué y que cada uno se vaya a hacerlo a su puesto de trabajo, lo que se hace es juntar a todo el equipo en una sala, a parte del resto de la oficina, y trabajan allí hasta que se ha finalizado el proyecto. También pueden reunirse una o dos veces a la semana para dedicar todo el día al proyecto. En esa sala cuentan con todo el material necesario y trabajan cada uno en la parte del proyecto en la que son expertos. El hecho de estar todos juntos les permite hablar y comunicarse directamente, estar al tanto de los cambios que se van produciendo en tiempo real y tener siempre presente el trabajo de los demás compañeros, que al final, no deja de ser el trabajo del resto de la cadena de creación del proyecto.

Algunos ejemplos:

Valor: Individuos e interacciones por sobre procesos y herramientas

Principio: Técnicos y no técnicos trabajando juntos TODO el proyecto

Según explicaba Jobs, **Apple** es una empresa "increíblemente colaborativa". En ella no hay ningún comité, lo que encuentras son **personas al cargo de proyectos**: una persona está al cargo del sistema operativo del iPhone, otra persona está al cargo del hardware del mac, otra persona está al cargo de la ingeniería del hardware del iPhone, otra persona está al cargo del marketing mundial, otra persona está al cargo de operaciones... Apple se organiza como una startup, la mayor startup del mundo. Todos se reúnen durante 3 horas una vez a la semana y hablan de qué están haciendo y cómo lo están haciendo. Cada responsable sabe lo que hacen los demás, para tenerlo en cuenta en su propio trabajo, por lo que hay mucho trabajo en equipo entre los responsables y ese trabajo se filtra luego hacia abajo, para pasar los objetivos hacia el resto de los equipos de trabajo de la compañía. Este ejemplo puede también relacionarse con el principio de que el equipo evalúa su desempeño y ajusta su manera de trabajar

Software funcionando por sobre documentación detallada

Releases frecuentes (2 semanas a un mes)

No centrarse tanto en la documentación sino en el desarrollo, para poder anticipar el funcionamiento del producto final (Si documentar implica tener menos software funcionando es preferible dejar de lado la documentación), con prototipos previos que ofrecen un 'feedback' del cliente, que genera ideas.

La mejor métrica de progreso es la cantidad de software funcionando.

Estos prototipos nos brindan información acerca de si se están cumpliendo los requerimientos de la mejor manera posible o como el cliente desea.

El ritmo de desarrollo es sostenible en el tiempo.

Se hacen release cada cierto tiempo, lo que permite que sea más ágil el desarrollo y se intenta mantener el mismo ritmo para cada release.

Atención continua a la excelencia técnica

Brindarle continuidad y atención al desarrollo del software garantiza que el mismo funcione de manera correcta y logremos lo mejor posible una excelencia.

Simplicidad – Maximización del trabajo no hecho

Evitemos que se genere trabajo innecesario y soluciones muy complejas que puedan llevar a consumir mucho tiempo y retrasar la salida de software funcional.

Algunos ejemplos:

Valor: Software funcionando por sobre documentación detallada

Principio: Simplicidad – Maximización del trabajo no hecho

Zara aprovecha cuatro veces más sus recursos que la mayoría de marcas gracias a su sistema ágil, ¿qué hacen? Ellos lo llaman *postponement*, nosotros lo podemos traducir como aplazamiento. Esta técnica consiste en posponer la creación, o incluso el envío de productos, lo máximo posible y teniendo en cuenta su récord logístico están en condiciones de hacerlo.
Los productos que fabrica Zara son difíciles de pronosticar; es decir, tienen un ciclo de vida corto y nunca sabes con exactitud si van a gustar o no. Por lo tanto, definir si se necesitan mil unidades o un millón es casi imposible. La solución que ha encontrado Zara es hacer previsiones a la baja, en el caso de que finalmente falten productos entonces arrancan la rueda de creación y envío express, que ya tienen bien engrasada, y en menos de 15 días tienen el problema de stock solucionado.

Colaboración con el cliente sobre negociación contractual

La prioridad es satisfacer al cliente a través de releases tempranos y frecuentes

Los releases tempranos permiten aumentar la confianza del cliente hacia el equipo desarrollador ya que, al ver el software de manera más frecuente, este puede hacer cambios o ver que se esté haciendo lo acordado. Evita conflictos o errores que con el tiempo llegan a ser complicados de reparar.

Recibir cambios de requerimientos, aun en etapas finales

El cliente es un miembro más del proceso del desarrollo de software, se integra y colabora con el grupo de trabajo, reduciendo posibles riesgos. La retroalimentación del cliente de forma continua enriquece el producto final, y permite encontrar aquellas tareas que podrían darle mayor valor al producto y proporcionar ventaja competitiva.

El medio de comunicación por excelencia es cara a cara

Negociar sin haber antes hablado claramente qué es lo que busca el cliente o qué es lo que puede entregar el desarrollador está destinado a un trabajo ineficiente y posiblemente con muchos errores. El conversar cara a cara demuestra interés en la conversación así como también ayuda a identificar problemas que surgen espontáneamente, el contacto humano debe estar presente por su capacidad de dar soluciones, por otro lado, se evitan malos entendidos, ya que el cliente puede explayarse más y dar más detalles sobre lo que necesita. De esta forma se obtiene un objetivo común a todos y se genera confianza.

Técnicos y no técnicos trabajando juntos TODO el proyecto

Trabajar juntos es la definición de colaboración , y el contemplar tanto técnicos como no técnicos, hace referencia a que tanto el equipo de desarrollo como otros involucrados dentro de los cuales está el cliente trabajan en conjunto. Así, podemos saber lo que el cliente realmente quiere (y ayuda al cliente a reconocer qué es lo que él mismo quiere, también, que muchas veces no sabe).

Algunos ejemplos:

Valor: Colaboración con el cliente sobre negociación contractual

Principio: La prioridad es satisfacer al cliente a través de releases tempranos y frecuentes

En **PayPal** los objetivos se revisan cada día y todos los equipos involucrados echan un vistazo y discuten sobre qué van a hacer, qué han hecho y qué aprendieron el día anterior.

Responder a cambios por sobre seguir un plan

Recibir cambios de requerimientos, aun en etapas finales

Aunque tengamos un plan detallado a seguir, si el cliente desea cambiar, agregar o quitar un requerimiento debemos poder adaptarnos a ese cambio y ajustar el plan para poder dar una respuesta adecuada.

La mejor métrica de progreso es la cantidad de software funcionando

Podemos tener un plan específico a seguir. Sin embargo, podemos darnos cuenta de que el software que tenemos en funcionamiento no se adapta a los requerimientos vamos a tener que responder al resultado de esta métrica y adaptar el proyecto para que tengamos el mejor resultado. Las métricas siempre nos darán una idea de dónde estamos parados en cuanto al producto final.

Atención continua a la excelencia técnica

El equipo debe tener el conocimiento y las habilidades necesarias para que los cambios en el producto puedan darse de forma rápida

A intervalos regulares, el equipo evalúa su desempeño y ajusta la manera de trabajar

Como se menciona en el punto anterior, a intervalos regulares hay que tomar métricas para saber en qué lugar nos posicionamos en cuanto a la evolución del proyecto, ya sea que se evalúe el producto o el proyecto, hay que ajustar la manera de trabajar para siempre hacerlo de la manera más óptima.

Algunos ejemplos:

Valor: Responder a cambios por sobre seguir un plan

Principio: Recibir cambios de requerimientos, aun en etapas finales

Compartiendo una experiencia personal, participando en un proyecto en el que tuve que realizar un sistema para una empresa. Formulé un plan para el proyecto, en el que primero debía identificar los requerimientos, luego diagramar la resolución, seguido de la implementación y la prueba. El sistema ya estaba en funcionamiento, y estaba en período de prueba. En este momento, la empresa que solicitó el sistema me comunicó que necesitaba un cambio porque tuvo un error en el momento de expresar sus requerimientos, y eso me llevó al paso de implementación otra vez, tuve que retroceder y reajustar mi plan para adaptarme a la necesidad del cliente. Aceptar cambios de los requisitos nos demuestra que el equipo tiene capacidad de adaptación y flexibilidad.

También puedo mencionar que, en este caso, tuve que realizar varias entregas tempranas del software en funcionamiento, para que el cliente pueda usarlo, adaptarse y corregir en caso de que sea necesario. Justamente de una de estas entregas tempranas surgió la devolución del cliente, en la que me comunicó que el sistema tenía esta falla y tuve la posibilidad de corregirla antes de la entrega final. El sistema que tuve que realizar era algo "chico" con pocos requerimientos, no fueron entregas cada dos semanas sino cada 4-5 días, pero logré ir satisfaciendo al cliente con estas entregas continuas y pude entregarle el software que necesitaba y que mejor se adaptaba a sus requerimientos.

Algunos ejemplos:

Valor: Responder a cambios por sobre seguir un plan

Principio: A intervalos regulares, el equipo evalúa su desempeño y ajusta la manera de trabajar

Para explicarlo mejor, recurrimos a un ejemplo. Un equipo de trabajo está desarrollando un software; mientras unos están desarrollando el frontend, otros trabajan con el backend. Luego de dos semanas de trabajo, el equipo evalúa su desempeño y reconocen que hay desarrolladores que se ocupan del frontend pero que no están avanzando mucho, y deciden hacer un cambio y darles una tarea diferente, para seguir trabajando de la manera más útil posible. Pueden aprovechar y ajustar los tiempos que se dedican a cada parte del desarrollo si notan que están perdiendo tiempo en algo que no es muy necesario, y aprovechar ese tiempo para otras cosas más útiles

Bibliografia

- Material proporcionado por la catedra.
- https://www.iebschool.com/blog/metodologia-agil-agile-scrum/
- https://kzi.mx/el-manifiesto-agil-valores-y-principios/