Notas

Video 31 Retrospectives

Existen dos tipos de retrospectivas, **las retrospectivas de sprint**, que se enfocan en reflexionar sobre lo ocurrido en el último sprint, y **las retrospectivas de proyectos**, que se llevan a cabo al finalizar un proyecto completo. Una retrospectiva analiza lo que salió bien, lo que no salió bien y cómo mejorar el proceso en el futuro.

En las retrospectivas de sprint, las ideas aprendidas se aplican al siguiente sprint, lo que permite una mejora continua del proyecto.

las retrospectivas de proyectos, se revisa el proyecto en su totalidad una vez que se terminó. Aunque no haya más sprint, se realiza una retrospectiva para aplicar las lecciones aprendidas en futuros proyectos y permitir que el equipo resuelva diferencias y repare relaciones dañadas.

video 32 Retrospectives issues

algunos de los problemas más comunes por lo que no se implementan las retrospectivas son, la falta de tiempo, prefieren mirar adelante en lugar de hacia atrás, miedo a hablar de sí mismos o simplemente falta de interés.

Es importante que el equipo se sienta seguros para hablar y sobre el proyecto, ya que si no lo hacen, los problemas pueden pasar desapercibidos y no ser resueltos. Esto se puede resolver con una comunicación abierta, mostrar interés por las contribuciones, un liderazgo positivo y respetar a todos los miembros del equipo por igual. Es importancia de tener un equipo funcional en lugar de una disfuncional, donde todos trabajen juntos para lograr los objetivos del proyecto.

Las retrospectivas no deben ser momentos de culpa, sino oportunidades para reflexionar, mejorar y construir relaciones de confianza.

video 33 Sprint Retrospective Example 1

El video muestra un ejemplo de un equipo realizando una retrospectiva rápida de su sprint anterior. Discuten los puntos positivos, como completar todas las historias de usuario esperadas y tener reuniones diarias de stand-up. También identifican áreas de mejora, como escribir pruebas unitarias, mejorar el uso del panel de tareas y considerar la programación en pares.

video 33 Sprint Retrospective

las reuniones retrospectivas de sprint permiten evaluar tanto el producto como el proceso de desarrollo, brindando la oportunidad de identificar lo que funcionó bien, lo que salió mal y qué se puede mejorar. Es importante tener un ambiente seguro y donde se pueda tener una conversación abierta y honesta, donde los participantes puedan expresar sus opiniones sin temor.

En una reunión hay tres preguntas principales ¿qué fue bien este sprint?, ¿qué no salió bien este sprint?, ¿qué podríamos mejorar para el próximo sprint?, el Scrum Máster debe dirigir la reunión de manera constructiva, comenzando con aspectos positivos, fomentando un ambiente respetuoso y concluyendo con acciones concretas para el siguiente sprint. Las reuniones no deben centrarse únicamente en el producto, sino también en el proceso de desarrollo

video 34 Project Retrospective Example

El video muestra un ejemplo de un equipo realizando una retrospectiva de proyecto, se discutió si el proyecto fue exitoso y se reconoció el esfuerzo del equipo, después, se construyó una línea de tiempo destacando los eventos importantes y se habló sobre lo que funcionó bien, se compartieron lecciones aprendidas, como la importancia de escribir casos de prueba y la necesidad de consultar antes de tomar decisiones. Se termino la reunión con lo que se haría diferente en futuros proyectos.

video 35 Anti-Patterns - Team

trabajar en grupos separados puede llevar a la falta de comunicación entre ellos, lo que se conoce como trabajar en silos. Estos son malos para las organizaciones de desarrollo de software, ya que dificultan mantener una imagen unificada y una estrategia de gestión efectiva. Para evitar este problema debemos reducir la cantidad de gestión en la organización y fomentar una atmósfera de comunicación positiva entre los equipos.

Un antipatron es la **dependencia del proveedor**, es cuando un equipo de desarrollo crea un producto que depende en gran medida de una única solución (tecnológica o proveedor). Esto problemas para adaptarse al cambio, si esa tecnología no puede adaptarse al cambio. Para evitarlo, se debe realizar una investigación antes de comprometerse con una tecnología o solución.

La **sobreingeniería** es un antipatron que, a crear un producto más complejo de lo necesario, agregando características innecesarias o especulativas.

Otro antipatron es **el chapado en oro** se refiere a esforzarse demasiado en un proyecto sin obtener un beneficio adicional significativo.

Es importante detenerse cuando el producto funciona bien y no agregar características adicionales de lujo que no sean realmente necesarias.

video 36 Anti-Patterns - Development

centrarse excesivamente en la documentación completa puede ser contraproducente, ya que consume tiempo valioso de los desarrolladores. Existe el antipatrón Viewgraph Engineering, que ocurre cuando se dedica demasiado tiempo a crear presentaciones y documentación en lugar de desarrollar código.

Se debe evitar tareas que bloqueen el desarrollo del producto y sugiere que el tiempo dedicado a informes y presentaciones podría ser utilizado para crear prototipos básicos.

El antipatron fire drill, es cuando se trabaja poco durante la mayor parte del proyecto porque se pierde el tiempo en actividades no relacionadas con el proyecto, y luego se hace un gran esfuerzo hacia el final

la heroica, donde se depende de una persona para resolver los problemas del proyecto. estas personas son responsables de resolver la mayoría de los problemas técnicos y llevar a cabo las tareas críticas, esto aumenta la dependencia y el riesgo asociado.

la Marcha de la Muerte, es donde el equipo de desarrollo continúa trabajando en un proyecto a pesar de saber que está destinado al fracaso, lo cual afecta negativamente la moral y la calidad del producto.

video 37 Anti-Patterns - Individual Developers

Se mencionan los siguientes anitpatrones

El **cañón suelto** se refiere a una persona que toma decisiones importantes en un proyecto sin consultar al resto del equipo, y también puede causar problemas al expresar sus opiniones sobre cualquier tema, esto puede generar problemas en los demás miembros del equipo, lo que puede llevar a decisiones poco efectivas.

La **violencia intelectual** es cuando una persona utiliza su conocimiento avanzado para intimidar a otros miembros del equipo, ejemplo, menospreciar a aquellos que hacen preguntas sobre temas que no conocen, esto puede afectar negativamente la moral del equipo y dificultar la colaboración.

el correo electrónico puede ser una forma de comunicación ineficiente y puede provocar malentendidos, es mejor utilizar otros medios, como el teléfono o las reuniones cara a cara, para asegurar una mejor comunicación y evitar malentendidos.

video 37 Test Anti-Patterns - Management

Se mencionan los siguientes anitpatrones

la **microgestión** ocurre cuando un gerente se involucra excesivamente en cada detalle de un proyecto, pidiendo ser CC en todos los correos electrónicos, ofreciendo su opinión en cada decisión y controlando constantemente el trabajo de los desarrolladores, esto puede ser perjudicial para la moral del equipo y la calidad del producto.

El **gerente gaviotas**, que es aquella persona que aparece solo cuando hay un problema, causa conmoción y estrés en el equipo y luego desaparece.

Para evitar estos antipatrones se recomienda, la comunicación efectiva y la reducción de la carga de trabajo para mejorar el ambiente laboral, evitar medios de comunicación como los correos electrónicos pueden generar este tipo de problemas, preferible utilizar métodos más directos, como la comunicación en persona.

Ideas Principales

la importancia de las retrospectivas en los proyectos de desarrollo de software. El beneficio de realizar estas reuniones para reflexionar sobre lo ocurrido, identificar áreas de mejora y fortalecer las relaciones del equipo, y la importancia de la comunicación.

Resumen

Los videos hablan sobre la importancia de las retrospectivas en los proyectos de desarrollo de software. Se mencionan dos tipos de retrospectivas: las de sprint, que se enfocan en el último sprint, y las de proyectos, que se realizan al finalizar un proyecto completo. Las retrospectivas permiten reflexionar sobre lo ocurrido, identificar áreas de mejora y fortalecer las relaciones del equipo. Se presentan antipatrones comunes que impiden la implementación de las retrospectivas, y la importancia de la comunicación abierta y el trabajo en equipo para lograr mejores resultados en los proyectos de desarrollo de software.