
Universidad
Tecmilenio®

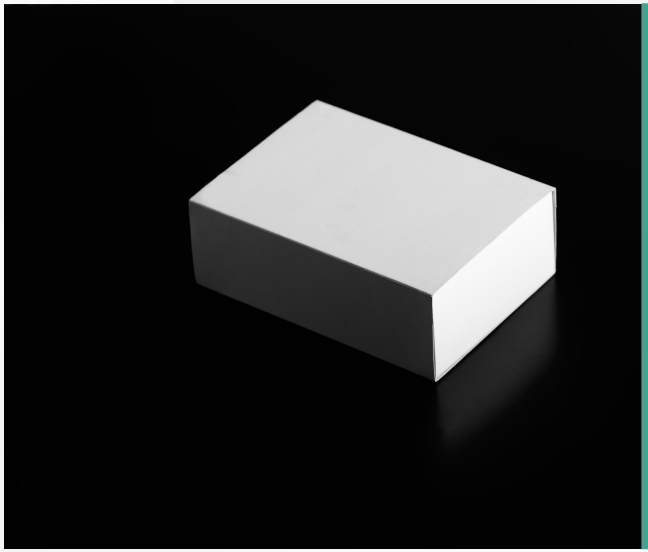
Universidad Tecmilenio

Software Testing

Pruebas de caja blanca

Semana 3

Introducción



Todas las organizaciones afrontan un gran problema cuando hablamos de calidad de software, ya que ninguno puede estar 100% libre de errores en su primera entrega.

Por ello, surge la necesidad de realizar pruebas de calidad que converjan hacia el aseguramiento de la eficiencia del software antes de salir al mercado, ya sea de forma interna o externa.

Explicación

Las pruebas de caja blanca, también conocidas como pruebas estructurales o de caja transparente, son una técnica de prueba de software, donde se conocen las funciones, métodos, clases y todo lo relacionado con el código.

Los tipos de pruebas de caja blanca son las siguientes:

- Pruebas de caja blanca.
- Flujo de control.
- Flujo de datos.
- Bifurcación.
- Caminos básicos.

Universidad Tecmilenio

Explicación

Objetivo de la prueba de caja blanca

Crear los casos de prueba que mejor se adapten a las necesidades y ejecutarlas.

Comprender la funcionalidad de la aplicación mediante el código fuente.

Explicación

Herramientas para prueba de caja blanca

- Veracode.
- EclEmma.
- Unidad de control.
- Cfix.
- Pruebas de Google.
- Emma.
- Nunit.
- Unidad Cpp.
- Junit.
- JsUnit.

Explicación

Pruebas de declaración o sentencia.

Pruebas de ruta o decisión y cobertura.

La cobertura de declaración o sentencia dicta que se ejecute por lo menos una vez cada sentencia del programa. Este criterio es muy necesario, pero no llega a ser del todo suficiente.

La cobertura de ruta o decisión comprueba todas las rutas posibles para el software. Cada sentencia se ejecuta al menos una vez junto con cada cobertura de ruta o decisión (if, else, etc).

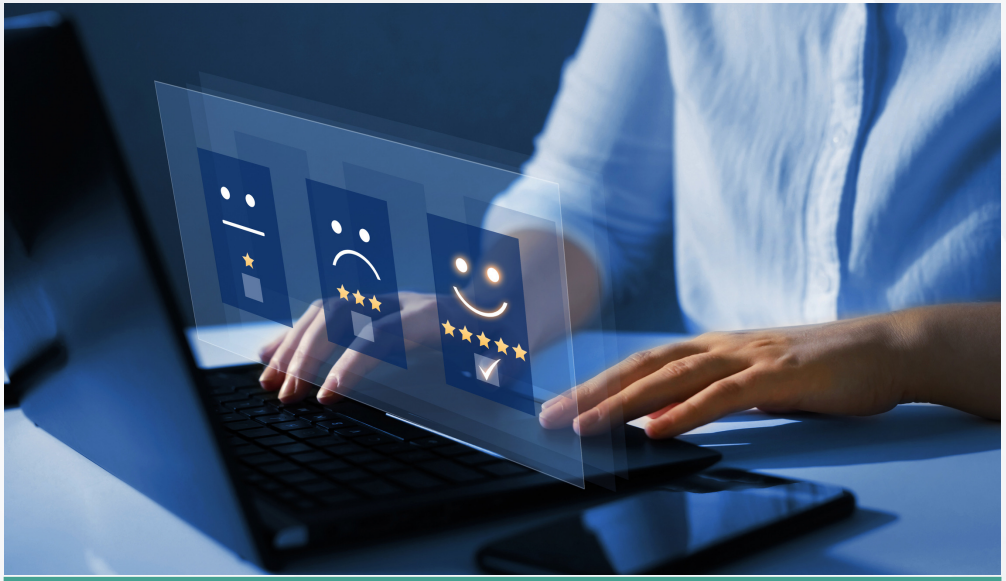
Actividad

¿Consideras que estas pruebas son suficientes para asegurar la calidad total del sistema?

¿De qué otra forma te imaginas que puedes realizar las pruebas?

Universidad Tecmilenio

Cierre



La ejecución de pruebas de caja blanca implica conocer la estructura interna del código. A partir de ello, se elaboran los casos de prueba que derivarán en asegurar la calidad del sistema e incrementar su eficiencia.

Para la correcta definición y ejecución de las pruebas, siempre será necesario tener conocimientos técnicos adecuados para poder analizar el código, realizar los casos de prueba, ejecutar las pruebas e interpretar los resultados.

Universidad Tecmilenio

Software Testing

Pruebas basadas en experiencia

Semana 3

Los defectos son más difíciles de identificar y generalmente suceden bajo condiciones muy específicas.

En ocasiones, esta situación solo puede resolverla un probador con mucha experiencia, conocimiento, uso de aplicaciones y tecnologías en particular, incluso entran en juego la imaginación e intuición de dicha persona.

A continuación, aprenderás que las pruebas basadas en la experiencia son fundamentales en el desarrollo de un software de calidad.

Explicación

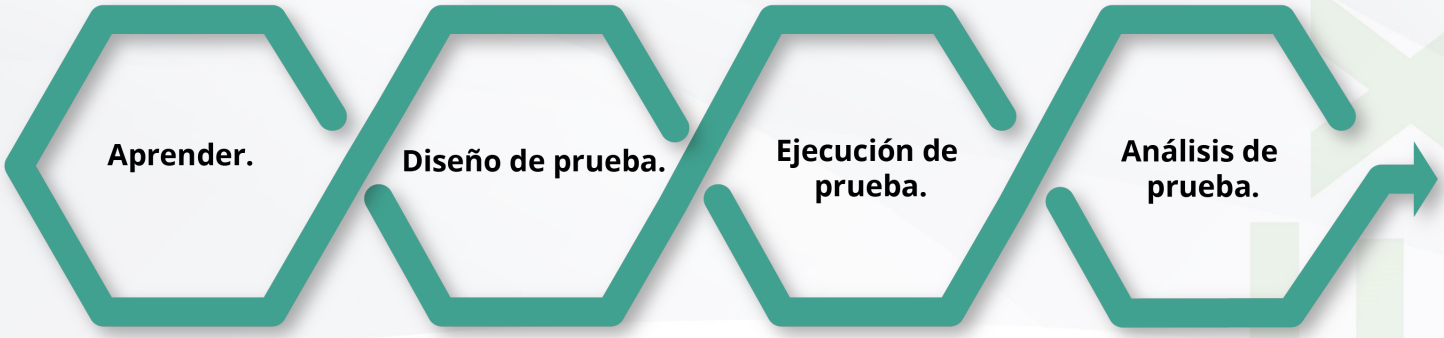
Las pruebas basadas en la experiencia son aquellas que se derivan de la habilidad e intuición del probador y de su experiencia con aplicaciones y tecnologías similares.

Predicción de errores	Pruebas exploratorias	Checklist
• No se especifica en ningún manual o documento de instrucciones de prueba. • Las personas con experiencia en el sistema pueden adivinar errores.	• Se utilizan para diseñar casos de prueba, cuando la información base no está estructurada.	• Realiza el diseño de software teniendo en cuenta su estructura general, módulos que tendrá, acceso a la información, etc.

Explicación

Pruebas exploratorias

Técnica que se utiliza para diseñar casos de prueba cuando la información base se encuentra poco estructurada y cuando el tiempo de prueba es escaso.



Explicación

Pruebas de checklist

Técnica de prueba basada en la experiencia. El probador experimentado utiliza una lista de alto nivel de elementos para anotar, verificar o recordar un conjunto de reglas o criterios para verificar un software.

Las ventajas de esta prueba son las siguientes:

Actividad

¿Consideras que estas pruebas son mejores que otras técnicas para asegurar la calidad total del sistema?

¿De qué otra forma te imaginas que puedes realizar las pruebas?

La ejecución de pruebas basadas en experiencia implica el conocimiento y la perspicacia del probador.

A partir de ello, se elaboran los casos de prueba que no siguen necesariamente pasos sistematizados, pero son sumamente efectivos para asegurar la calidad del sistema.

Para la correcta definición y ejecución de las pruebas, siempre será necesario tener un vasto conocimiento y sobre todo saber que estas pruebas arrojarán mejores resultados si se combinan con otras técnicas.

Universidad Tecmilenio

Software Testing

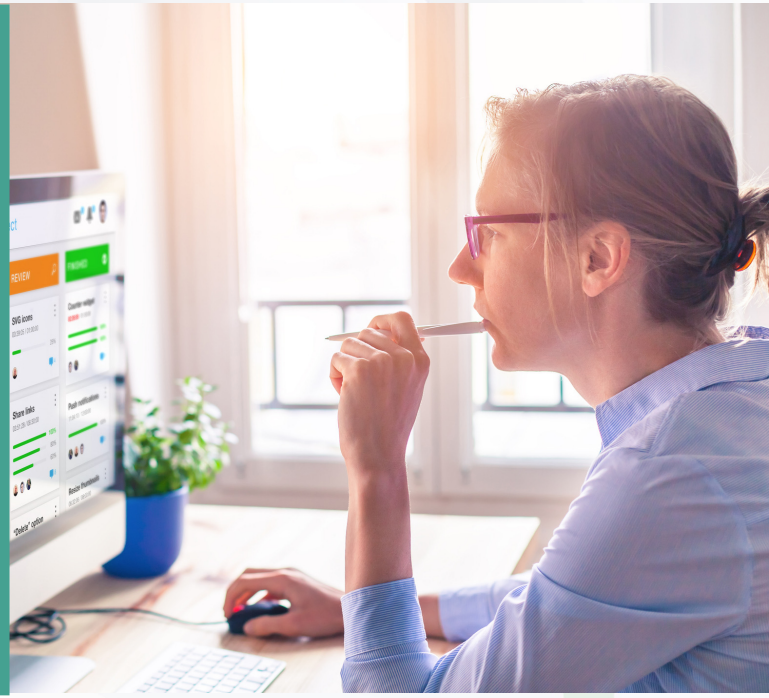
Fundamentos de Agile Testing

Semana 3

Introducción

En los últimos años, las metodologías ágiles han tenido un crecimiento importante. La flexibilidad de sus *frameworks* no radica en la gestión del proyecto mismo, sino en el planteamiento de nuevos principios, paradigmas de la tecnología o ideas innovadoras.

Por lo que antes de entender Agile Testing, es importante repasar los fundamentos de los modelos ágiles, incluyendo su manifiesto.



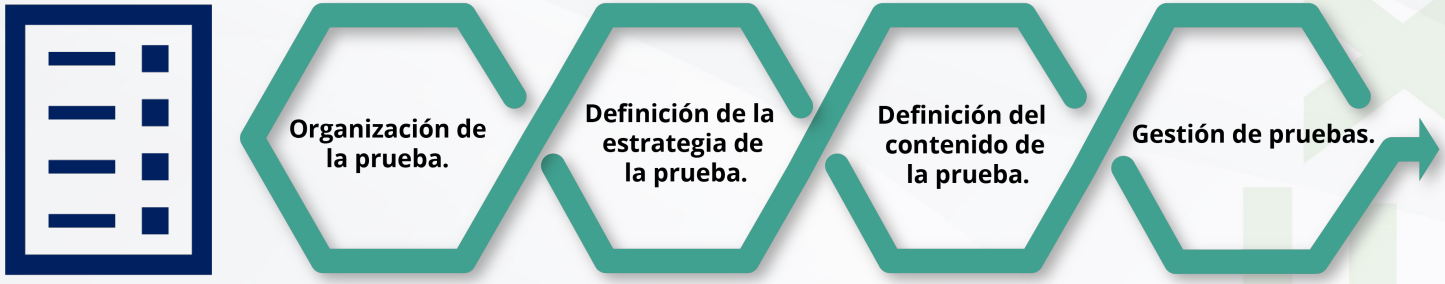
Explicación

Manifiesto ágil

- La interacción con los individuos es siempre más importante que los procesos y herramientas.
- El software en funcionamiento es siempre más importante que la documentación exhaustiva.
- La colaboración con el cliente es siempre más importante que la negociación de un contrato.
- Responder rápidamente al cambio es siempre más importante que seguir un plan.

Explicación

Proceso tradicional de pruebas: responsabilidades de un líder de pruebas



Explicación

Proceso ágil de pruebas

- Las pruebas son responsabilidad de todos.
- La codificación y las pruebas se llevan a cabo dentro del mismo equipo y no se tratan por separado.
- El rol de Scrum master es exclusivo de la metodología Agile-Scrum, sin embargo, es una posición que se ha convertido en el reemplazo del líder de proyecto.
- Las tareas de prueba se programan como tareas dentro de un sprint o implícitamente como parte de los criterios de finalización de otras tareas.

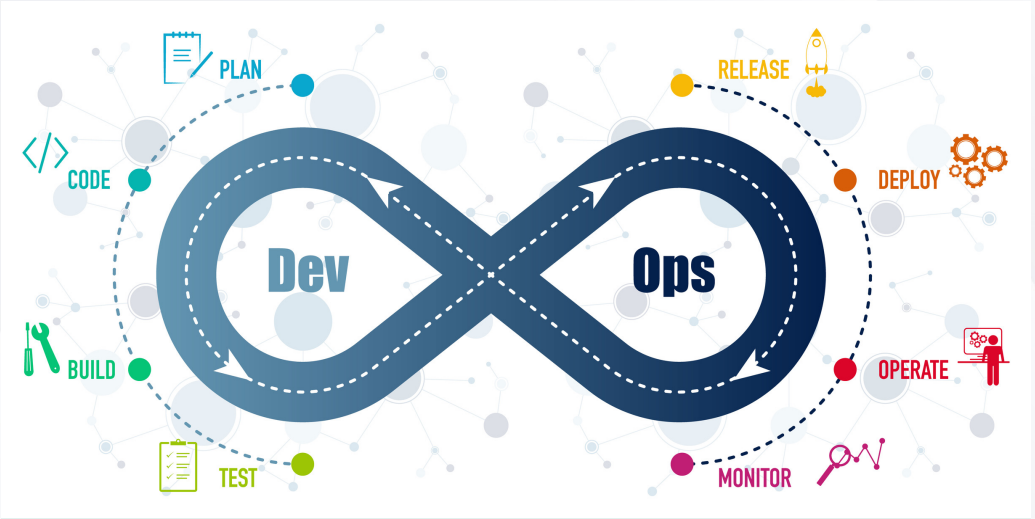
Actividad

¿Agile Testing es la solución perfecta para las pruebas de software en escenarios ágiles o hay un mejor método que se adapte cada vez más al acelerado desarrollo de la tecnología?

¿Agile Testing es un método sin riesgo o puede convertirse en un riesgo, en caso de no implementarse correctamente en un proyecto?

Universidad Tecmilenio

Cierre



La evolución de los procesos de desarrollo de software ha logrado modelos más rápidos y ha impactado la etapa de pruebas de software. Ahora, los equipos de trabajo deben participar en todas las fases.

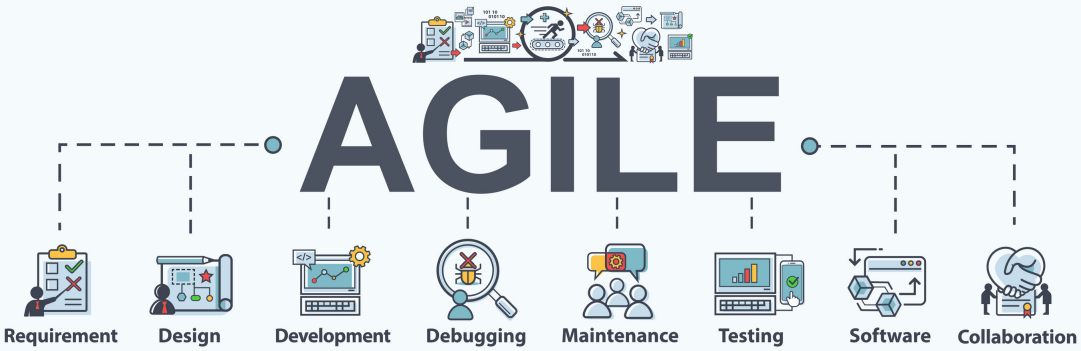
Sin embargo, Agile Testing resuelve estos problemas de forma ingeniosa. No obstante, has aprendido una característica de los métodos ágiles: utilizar elementos de las metodologías tradicionales cuando un proyecto lo amerite.

Universidad Tecmilenio

Software Testing

Métodos de Agile Testing

Semana 3



Uno de los grandes retos del Agile Testing fue encontrar la forma de implementar la fase de pruebas con éxito . Las pruebas de software en sí mismas se caracterizan por un proceso metódico y diseñado originalmente para métodos clásicos.

Esta fase es independiente y sucede hasta después de terminada la codificación.

Ahora, con los métodos agiles, debería ser totalmente diferente, ya que al final de cada iteración o sprint (cuando se usa Agile-Scrum), se debe entregar un producto de software terminado.

Explicación

Manifiesto de Agile Testing

Realizar las pruebas durante el proceso es más importante que hacer las pruebas al final.

Prevenir defectos es más importante que corregir defectos.

Entender las pruebas es más importante que comprobar funcionalidad.

Construir el mejor sistema es más importante que destruir el sistema.

Compartir la responsabilidad con todo el equipo es más importante que delegar la responsabilidad en un gerente de pruebas.

Explicación

Desarrollo orientado a pruebas

- Pruebas más cortas.
- Integración continua.
- Compilación completa por iteración.
- Probar el primer diseño y codificación.
- Probar lo que ha cambiado.
- Identificar soluciones genéricas.
- Aplicación automatizada de soluciones genéricas.

Universidad Tecmilenio

Explicación

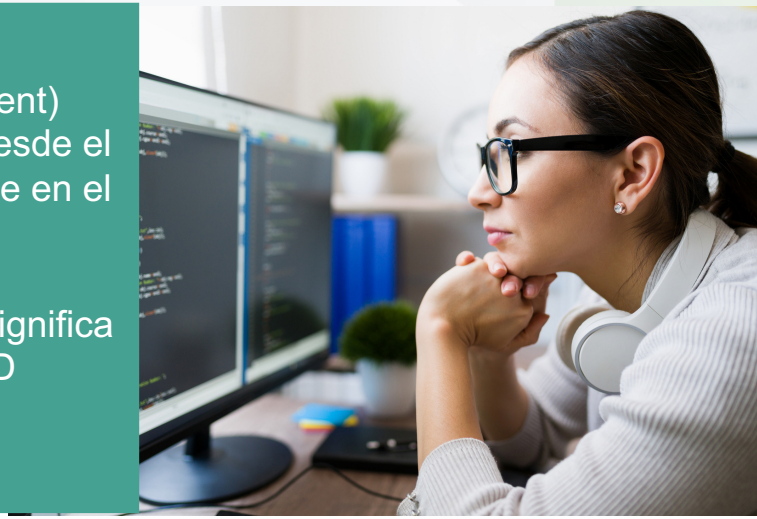
TDD es un método innovador y su objetivo no es solucionar el problema que se ha mencionado.

Fue concebido para desarrollar e implementar rápidamente códigos de alta calidad.

Para su aplicación, se utiliza un ciclo conocido como *Red-Green-Refactor cycle.*

ATDD (Acceptance Test- Driven Development) describe el comportamiento del software desde el punto de vista del usuario final, centrándose en el valor comercial de la aplicación.

Cuando una prueba de ATDD es exitosa, significa que todas las pruebas de las unidades TDD tuvieron éxito.



Explicación

Niveles de prueba

01 Unidad de prueba (unit testing): es la unidad más pequeña y se enfoca en los bugs (fallas de código) que se puedan presentar al escribir el código.

02 Pruebas de componentes (integration testing): nivel que tiene mayor significado en Agile Testing. Se prueba una suma de unidades de prueba que juntas cumplen una función.

03 Pruebas de sistema (system testing): este nivel es similar en una metodología tradicional y se prueba la solución completa.

04 Pruebas de aceptación del usuario (user acceptance testing): en este nivel, los dueños de negocio y usuarios dan el visto bueno del sistema por completo.

Explicación

Cuadrantes de pruebas

Cuadrante 1 (automatizado): está enfocado en la calidad del código.

Cuadrante 2 (automatizado y manual): está enfocado en los requerimientos de negocio.

Cuadrante 3 (manual): se enfoca en dar retroalimentación al primero y segundo cuadrante.

Cuadrante 4 (uso de herramientas): se enfoca en los requerimientos no funcionales.

Actividad

¿Cuáles son las ventajas al trabajar simultáneamente con la integración de pruebas de software y el desarrollo del código?

¿Cuál es la razón principal de utilizar diferentes pruebas al momento de desarrollar un proyecto?

Universidad Tecmilenio

Cierre

La industria del software es capaz de encontrar soluciones para cada uno de los retos que se presentan.

Ahora puedes afirmar que las soluciones de desarrollo de software para metodologías agiles suelen revolucionar todo y, en este caso, son capaces de cambiar paradigmas, al lograr un método para crear sobre la marcha un software a prueba de fallos.