
Universidad
Tecmilenio®

Universidad Tecmilenio

Software Testing

Técnicas para estimación de pruebas

Semana 5

Introducción

Toda persona que haya tenido que realizar una estimación de pruebas de un proyecto de desarrollo de software sabe que no es un tema sencillo.

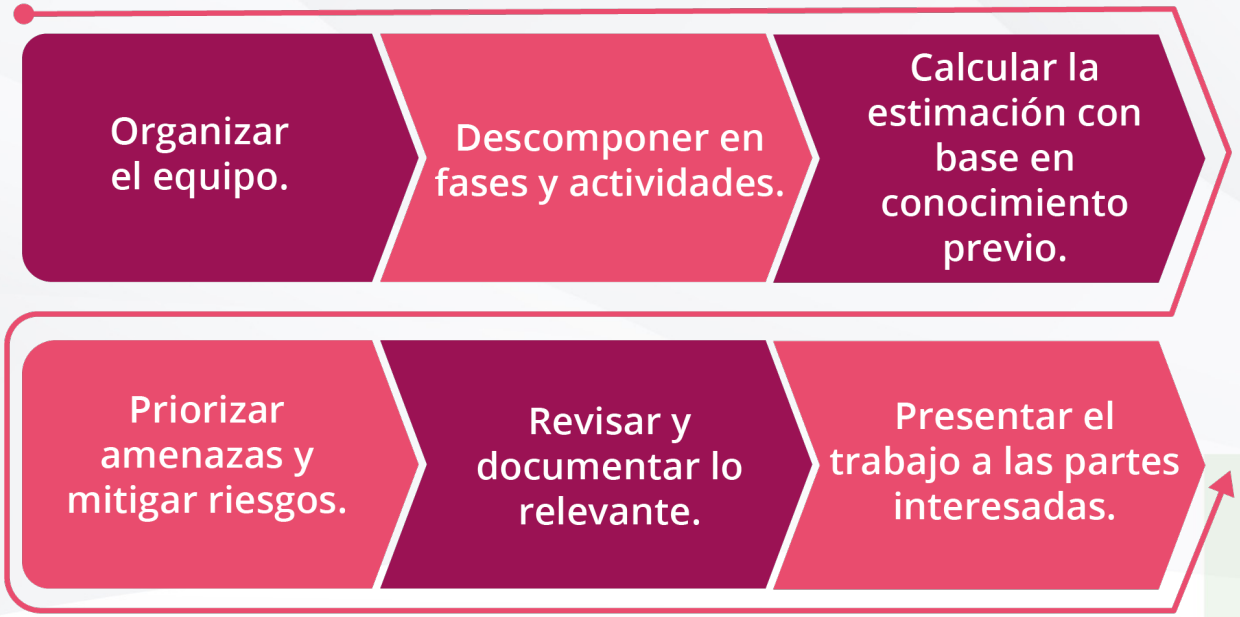
Para el éxito de cualquier proyecto, la estimación y la ejecución de pruebas adecuadas son de suma importancia, ya que, al cumplir con estas, y entregar en tiempo y forma, se construye una excelente confianza con el cliente y/o usuario.

Explicación

Tipos de estimaciones

Consisten en realizar un cálculo estimativo del esfuerzo y el número de casos de prueba que deben realizarse para llevar a cabo las pruebas necesarias y planteadas en la estrategia de calidad del aplicativo (Fundación MTP, s.f.).

Proceso de estimación



Explicación

Uno de los errores más comunes al planificar y estimar proyectos de pruebas es pensar solo en la ejecución. Para evitar esto, se deben tener en cuenta las siguientes tareas (PMO Informática, 2018):

- Plan de pruebas.
- Casos de prueba.
- Ejecución de pruebas.
- Gestión del defecto.
- Verificación completa.
- Regresión.
- Pruebas exploratorias, informe y resultados.

Universidad Tecmilenio

Estimación por juicio experto

Explicación

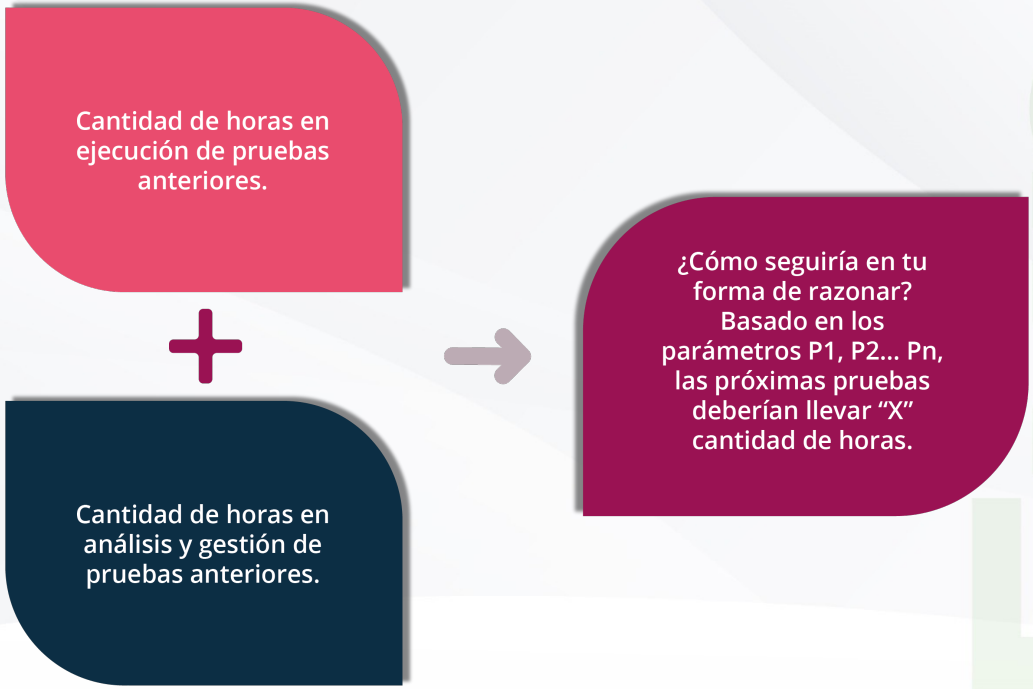
- Se consigue un experto en el software a probar.
- Asegurar que tengan los conocimientos suficientes y conocer cómo funciona el software.
- Hacer una descomposición funcional para estimar mejor.
- Tomar en cuenta posibles imprevistos para no tener una estimación muy optimista que pudiera afectar la entrega.

Explicación

Estimación paramétrica

- Puede contemplar datos pasados de referencia.
- Se hace con base en la relación entre variables.
- Método más preciso y confiable.
- Se consideran parámetros relevantes de pruebas pasadas.

Ejemplo



Explicación

Estimación basada en puntos función

- Técnica basada en la estimación de funcionalidad del software.
- Se usan fórmulas matemáticas basadas en parámetros, como tipo de componente, complejidad, factores de entorno, etc.
- “Es la principal herramienta para la medición funcional de productos de software y procesos involucrados en su desarrollo” (Impulse, 2022).
- Sentó el precedente el método IFPUG-FPA.

Universidad Tecmilenio

Explicación

- **IFPUG-FPA**
  - Ponderaciones basadas en la dificultad de probar.
  - Los puntos función se traducen en horas-hombre.
- **NESMA**
  - Método definido por la Netherlands Software Metrics Association.
- **MkII**
  - Método definido por la United Kingdom Software Metrics Association.
- **COSMIC**
  - Conocido como Full Function Points (COSMIC-FFP).
  - Definido por el Common Software Metrics International Consortium.
- **FiSMA**
  - De la asociación finlandesa de medición de software.

Actividad

¿Qué metodología de estimación es la más óptima y por qué?

¿Qué impacto habría en un proyecto de *testing* si no se realiza la estimación de las pruebas?

Bibliografía

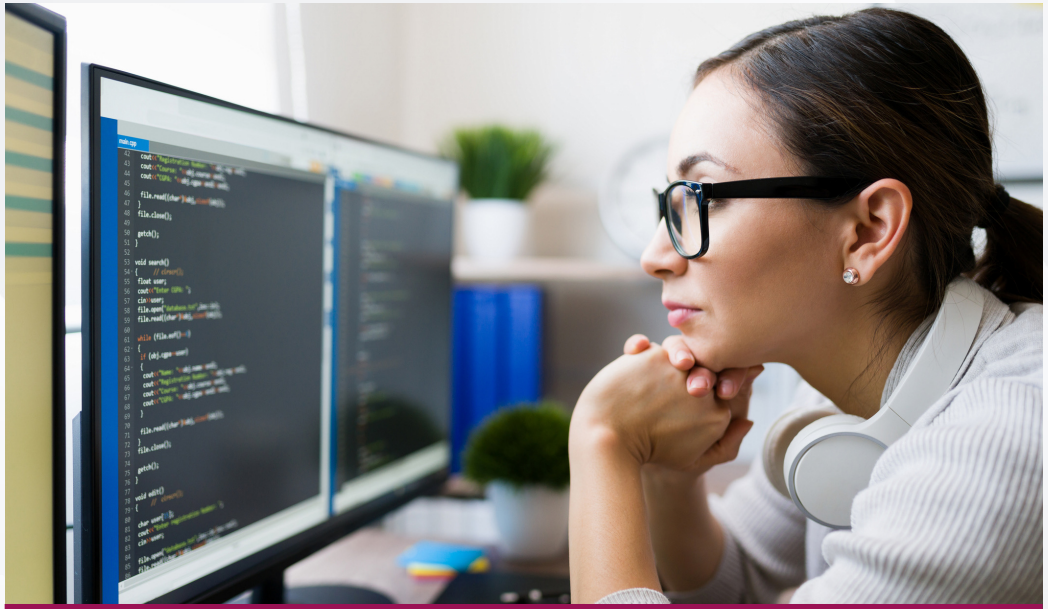
- Fundación MTP. (s.f.). *ESTIMACIÓN DE PRUEBAS.* Recuperado de https://www.mtp.es/aseguramiento-de-la-calidad/servicios-de-gobierno/estimacion-de-pruebas/

- Impulse. (2022). *Análisis de Puntos de Función y Estimación de Software*. Recuperado de https://impulseits.com/index.php?seccion=cursos&id_curso=42

- PMO Informática. (2018). *10 Técnicas de estimación de software*. Recuperado de http://www.pmoinformatica.com/2018/08/tecnicas-estimacion-software.html

Universidad Tecmilenio

Cierre



Como te podrás dar cuenta, la estimación de pruebas de software es una práctica que siempre necesitará de la participación de profesionales experimentados, así como de adoptar una mentalidad abierta para personalizar los procesos requeridos por la organización y el proyecto en general; teniendo en cuenta que la implementación exitosa de estos procesos y técnicas conducirán a una mejora en el esfuerzo de la ejecución de las pruebas.

Universidad Tecmilenio

Software Testing

Diseño de plan de pruebas

Semana 5

Introducción

A lo largo del tema, aprenderás el diseño, la estructura y el contenido de un plan detallado de pruebas que será la base para dirigir el esfuerzo en la ejecución de las mismas, convirtiéndose en una pieza clave y estratégica en el aseguramiento de la calidad del software.

Explicación

Definición y estructura del plan de pruebas

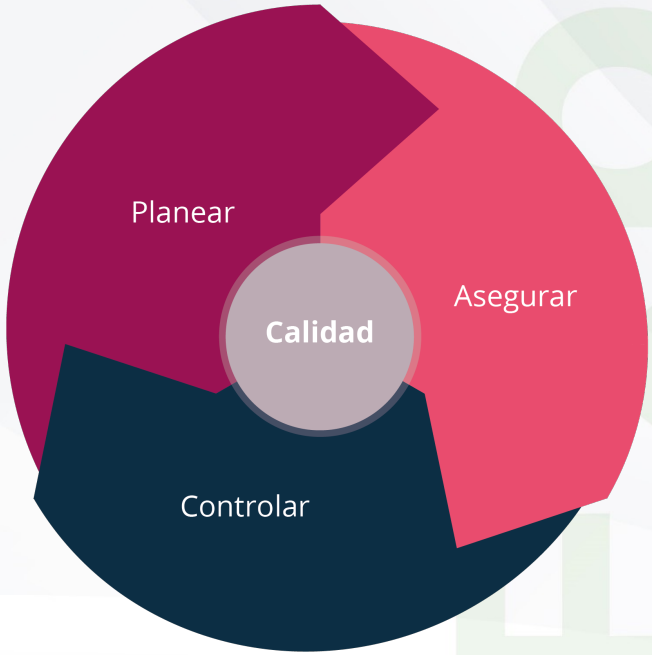
Se considera una guía para realizar las pruebas con base en el estudio y ejecución de casos de prueba definidos por los *testers*, para saber si el software cumple o no con los requisitos del proyecto y la calidad esperada.

Antes de construir el plan de pruebas, se recomiendan los siguientes pasos (Software Testing Bureau, 2022):

1. Identificar los objetivos de negocio con los que debe cumplir el software.

2. Conocer las fechas del proyecto en las que se planeó ejecutar las pruebas.

3. Conocer la metodología de desarrollo para a acoplar las instancias de prueba a los hitos del proyecto.



Actividad

Funcionalidades que se deben considerar para realizar un plan de pruebas (QALovers, 2019).

Usuario

- Cambios visualmente accesibles al usuario.
- Elementos con los que interactúa el usuario.

Internas

- Modificaciones de componentes o flujos de información interna.
- Se deben cubrir para cumplir con toda la funcionalidad del proyecto.

Explicación

Diez puntos que se deben considerar para definir la estructura de un plan de pruebas (PMOInformática, 2016):

- Analizar las necesidades del desarrollo de software.
- Identificar las nuevas funcionalidades que se probarán.
- Identificar necesidades de sistemas existentes que deben probarse.
- Definir la estrategia del plan de pruebas.
- Definir los criterios de inicio, aceptación y suspensión de pruebas.
- Identifcar los ambientes necesarios para las pruebas.
- Determinar las habilidades necesarias del personal y de entrenamiento.
- Establecer la metodología y los procedimientos de prueba.
- Elaborar el plan de pruebas.
- Identificar riesgos y un plan de acción para responder a ellos.

Universidad Tecmilenio

Diseño de contenido y uso del plan de pruebas

Explicación

No existe una forma o estándar general que dicte estrictamente cómo se debe realizar un plan de pruebas, sin embargo, sí hay organismos como el ISTQB (*International Soft ware Testing Qualifications Board*).

Explicación

Contenido de un plan de pruebas

Identificador	Características que no se prueban.
Introducción	Criterios de aprobación y fallo.
Objetivo	Criterios de suspensión y reanudación.
Estrategia	Tareas de las pruebas.
Alcance	Necesidades ambientales.
Propósito	Capacitaciones.
Entregables	Riesgos.
Características a probar	Laboratorio de usabilidad.

Actividad

¿Cuáles son los elementos más esenciales de un plan de pruebas y por qué?

¿Consideras importante tomar en cuenta las necesidades no funcionales para probar un sistema?

Bibliografía

- PMOInformatica. (2016). *Pruebas de software: 10 pasos para elaborar el plan de pruebas*. Recuperado de http://www.pmoinformatica.com/2016/01/elaborar-plan-pruebas-software.html
- QALovers. (2019). *Detallando un plan de pruebas*. Recuperado de https://www.qalovers.com/2019/01/detallando-un-plan-de-pruebas.html
- Software Testing Bureau. (2022). *¿Qué estructura debe tener el plan de pruebas?* Recuperado de https://www.softwaretestingbureau.com/crear-un-buen-plan-de-pruebas/

Universidad Tecmilenio

Cierre

Elaborar un plan de pruebas es indispensable en todo desarrollo de software, ya que este determinará si el software a entregar cumple con la calidad esperada por el cliente.

En la actualidad, los testers o personas dedicadas al mundo de las pruebas de software siempre necesitarán implementar un plan de pruebas para comunicar los entregables del proyecto a los involucrados.

Universidad Tecmilenio

Software Testing

Cobertura de pruebas

Semana 5

Introducción

Las pruebas son esenciales durante la ejecución del ciclo de vida de cualquier proyecto de desarrollo de software, ya que siempre van enfocadas en mejorar la calidad de este.

A lo largo del tema, aprenderás sobre la cobertura y la estructura de las pruebas, así como el análisis de requerimientos que será la base para dirigir el esfuerzo en la ejecución y cobertura de estas, siendo así una pieza estratégica en el aseguramiento de la calidad del software.

Explicación

Cobertura de pruebas

“Es un parámetro con el que podrás saber qué parte de tu fuente se ha sometido a pruebas” (Pittet, 2022). En otras palabras, la cobertura hace referencia a qué y cuánto se ha probado del software.

Ejemplo

"Si un código tiene 300 líneas y solo 270 de ellas se han ejecutado al realizar las pruebas, se tiene una **cobertura de pruebas** de 90%. La calidad de ellas no cubrirá el 10% que no está validado".

Explicación

Análisis y estructura de requerimientos

Para diseñar las pruebas, se toman como base los requerimientos funcionales, los cuales marcan los requisitos de lo que el software debe hacer.

Cobertura de prueba adecuada en el nivel de diseño

- Se pueden crear 20 requerimientos y 200 pruebas.
- El objetivo de las 200 pruebas son los 20 requerimientos y no se omite ninguno.

Estadísticas de prueba a nivel de ejecución

- Solo se ejecutan 180 de las 200 pruebas creadas y se direccionan solo 12 de los requerimientos.
- Por lo tanto, 8 requerimientos no están cubiertos, a pesar de que se ejecuta el 80% de las pruebas.

Cobertura de asignación de prueba

- Solo 190 de las pruebas están relacionadas con 16 requerimientos.
- El resto de los requerimientos no están asociados, por lo que la cobertura de asignación de prueba es del 80% (16 de 20 requisitos).

Explicación

Contexto de cobertura de pruebas

La cobertura de pruebas siempre puede tener un significado diferente en cada contexto y los requerimientos asociados.

01 Cobertura del software

¿Qué aspectos del software se han examinado?

Se centra sobre las áreas del software probadas y las que faltan de probar.

02 Cobertura de riesgos

¿Cuáles riesgos se han probado?

No se puede decir que el software se ha probado hasta que no se prueban los riesgos que existen.

03 Cobertura de requisitos

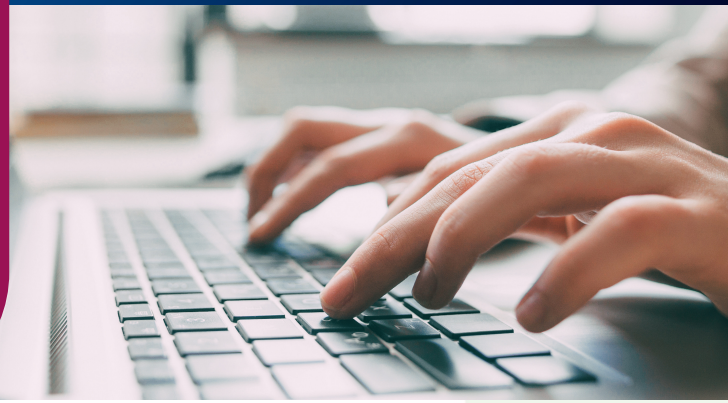
¿Cuáles requisitos se han probado?

Es muy importante, pues se valida que el software funciona como se espera.

Explicación

Ventajas del conocimiento de la cobertura para un tester

- Ayuda principalmente a priorizar los requerimientos de prueba.
- Ayuda a lograr una cobertura de requerimientos del 100% o evita la fuga de requisitos.
- El análisis de impactos se vuelve más fácil.
- Es útil para determinar los criterios de salida.

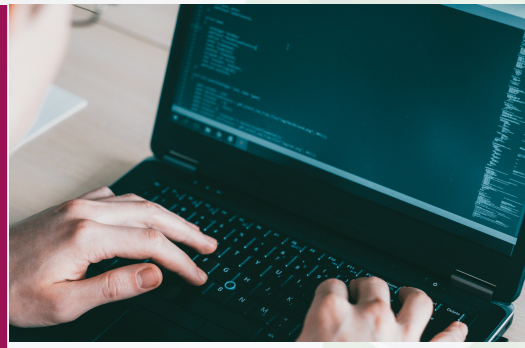


Explicación

Estructuras de la cobertura de pruebas con base en requerimientos

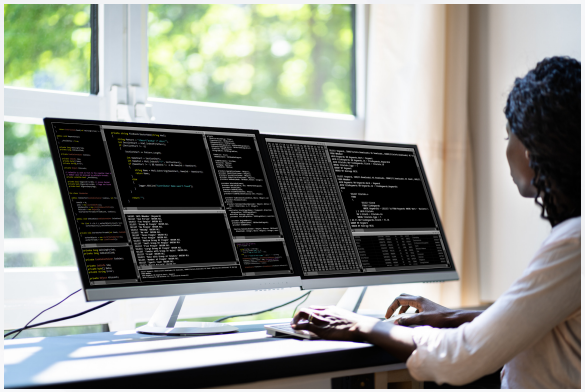
De acuerdo con IBM (2022), los equipos de testers usan dichos requisitos o requerimientos para poder brindar información y retroalimentación sobre lo siguiente:

- Últimos desarrollos realizados en el sistema.
- Los elementos que, dentro del sistema, necesitan probarse para asegurar la calidad del proyecto.
- Los requisitos de seguridad que hacen que el sistema cumpla con las expectativas.
- La respuesta real de una operación en comparación con la respuesta esperada.
- Criterios y *checklist*.
- Requerimientos mínimos de rendimiento.
- Revisar la redacción.
- Redacción corregida.



Explicación

Estructura básica de pruebas basadas en requisitos



Planeación	Diseño de casos de prueba	Ejecución	Cierre
• Estimación. • Alcance. • Requerimientos.	• Checklist para evaluación de cada requerimiento.	• Evaluación de cada requerimiento.	• Entrega de resultados y VoBo final de aceptación.

Actividad

¿Qué estrategia usarías para saber cuántos casos de prueba diseñar?

¿Cuál es la cobertura de pruebas que recomendarías para probar un un código?

Bibliografía

- IBM. (2022). *Realización de pruebas basadas en requisitos*. Recuperado de https://www.ibm.com/docs/es/elm/7.0.1?topic=requirements-based-testing
- Pittet, S. (2022). *¿Qué es la cobertura de código?* Recuperado de https://www.atlassian.com/es/continuous-delivery/software-testing/code-coverage

Universidad Tecmilenio

Cierre

La cobertura de las pruebas va ligada con la calidad del software que se está probando, por lo tanto, probar un requerimiento no solo es lanzar una prueba más, sino que se debe garantizar que el software cumpla con todos los requerimientos solicitados por el usuario.

Recuerda que no siempre aplica que, entre más pruebas realices, los resultados serán mejores, sino que, al tener un enfoque mejor estructurado, un objetivo determinado de cobertura de requerimientos y los métodos de prueba bien definidos, no se comprometerá la calidad del software.