Pruebas estáticas y dinámicas

Las pruebas son una combinación de múltiples actividades del ciclo de vida del software relacionadas con la planificación, el diseño y la evaluación del producto de software, con el objetivo de encontrar los defectos y determinar si el software cumple o no con los requisitos especificados.

Es por ello que en este módulo continuaremos desarrollando otros tipos de pruebas: las pruebas estáticas y las pruebas dinámicas. Estas se complementan entre sí y nos permiten entregar un software con la mejor calidad posible.

*“Las pruebas estáticas y dinámicas tienen el objetivo de proporcionar una* ***evaluación de calidad*** *de los productos de trabajo e* ***identificar defectos*** *en forma temprana.”*

**Conceptos básicos de la prueba estática**

La prueba estática se basa en la **evaluación manual** de los productos de trabajo (es decir, revisiones) o en la evaluación basada en herramientas del código u otros productos de trabajo (es decir, análisis estático). Este tipo de pruebas no requieren la ejecución del software que se está probando.

Se utilizan este tipo de pruebas para examinar cualquier producto de trabajo, como por ejemplo:

* Especificaciones, requisitos de negocio, funcionales y de seguridad.
* Épicas, historias de usuarios y criterios de aceptación.
* Especificaciones de arquitectura y diseño.
* Código.
* Productos de prueba: planes, casos, procedimientos y guiones de prueba.
* Manuales de usuario.
* Contratos, planes de proyecto, calendarios y presupuestos.

**Ventajas de las pruebas estáticas tempranas**

Cuando se aplica al principio del ciclo de vida del desarrollo del software, la prueba estática permite la detección temprana de defectos. Esto genera una reducción de costos y tiempo de desarrollo y prueba.

Por el contrario, si el defecto se encuentra luego de las pruebas dinámicas, solucionarlo va a requerir el cambio de código, realizar una prueba de confirmación y luego incluir el mismo en pruebas de regresión, además de los cambios de toda la documentación asociada.

**Defectos encontrados con pruebas estáticas**

Algunos de los defectos encontrados con pruebas estáticas que son más fáciles y económicos de detectar y corregir son:

* Defectos en los requisitos (inconsistencias, ambigüedades, etc.).
* Defectos de diseño (estructura de base de datos ineficiente, alto acoplamiento, etc.).
* Defectos de codificación (variables con valores no definidos, código inalcanzable o duplicado, etc.).
* Desviaciones con respecto a estándares (falta de uso de estándares de codificación).

**Defectos encontrados con pruebas estáticas**

* Especificaciones de interfaz incorrectas (unidades de medida diferente, etc.).
* Vulnerabilidades de seguridad (susceptibilidad a desbordamiento de la memoria intermedia).
* Diferencias o inexactitudes en la trazabilidad o cobertura de la base de prueba (falta de pruebas para un criterio de aceptación).
* Defectos de mantenibilidad (mala reutilización de componentes, modularización inadecuada, etc.).

**Pruebas Dinámicas**

**Conceptos básicos de la prueba dinámica**

Las pruebas dinámicas requieren la ejecución del software, componente o sistema.

Se complementan con las pruebas estáticas debido a que encuentra diferentes tipos de defectos. Para la generación de casos de prueba se utilizan diferentes técnicas de caja negra, caja blanca o basadas en la experiencia de usuario.

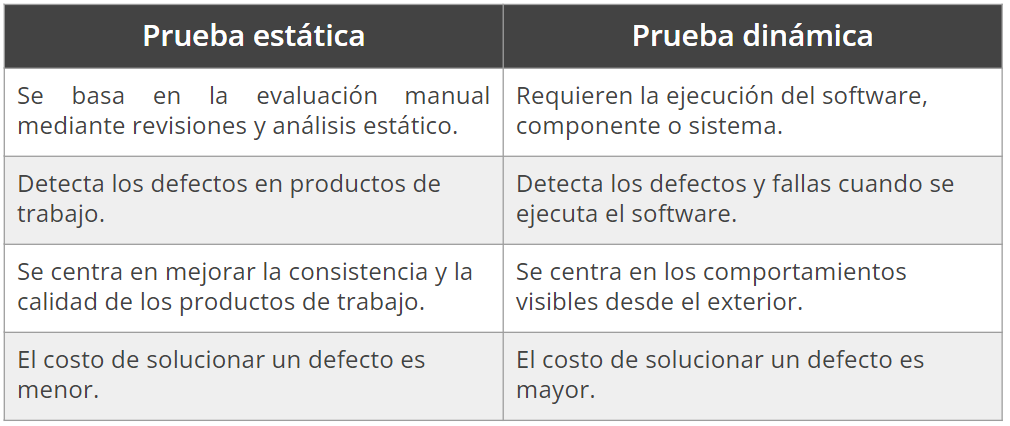
**Conceptos básicos de la prueba dinámica**

Durante las pruebas dinámicas se ejecuta el software utilizando un conjunto de valores de entrada y su resultado se analiza y compara con el resultado esperado.

Las fallas más comunes encontradas con este tipo de pruebas son:

* Fallas de funcionalidad.
* Fallas de interacción entre módulos.
* Fallas de rendimiento y seguridad.

**Conclusión**



**Proceso de revisión**

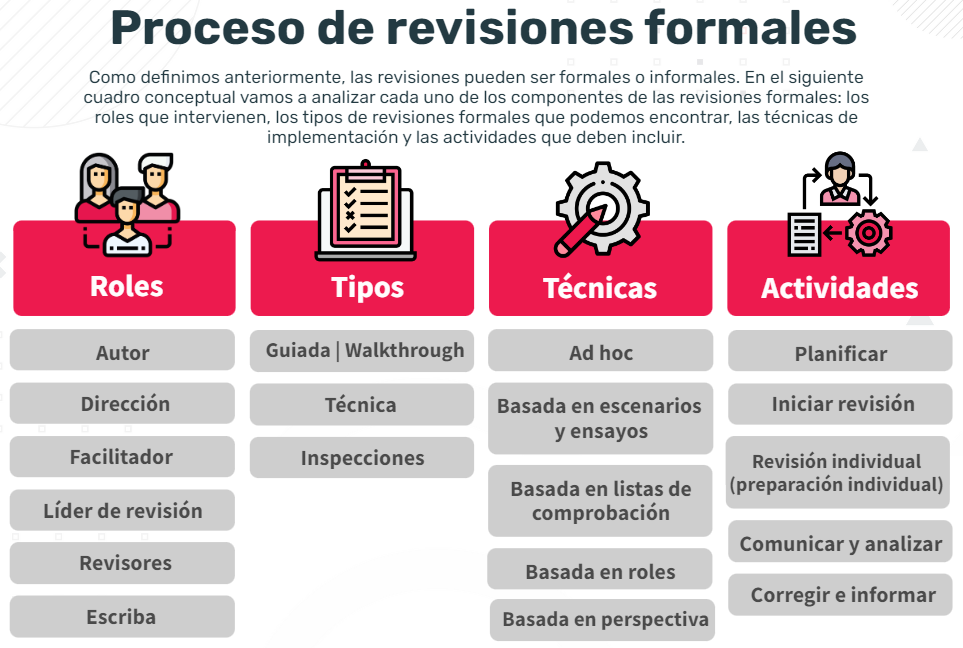
Dentro de las pruebas estáticas, una forma de detectar errores es mediante un proceso de revisión.

Las revisiones consisten en examinar cuidadosamente un producto de trabajo con el principal objetivo de encontrar y remover errores. Pueden ser realizadas por una o más personas.

Las revisiones pueden ser:

* Revisiones formales: Tienen roles definidos, siguen un proceso establecido y deben ser documentadas.
* Revisiones Informales: No siguen un proceso definido y no son documentadas formalmente.

El grado de formalidad del proceso de revisión está relacionado con factores, como el modelo del ciclo de vida del desarrollo del software, la madurez del proceso de desarrollo, la complejidad del producto del trabajo que se debe revisar, cualquier requisito legal y/o la necesidad de un rastro de auditoría.



**ROLES**

* **Autor**  
  Creador del producto de trabajo bajo revisión y quien corrige los defectos, en caso de ser necesario.
* **Dirección**  
  Planifica y controla las revisiones.
* **Facilitador**  
  Asegura el funcionamiento efectivo de las reuniones de revisiones y las modera.
* Líder de revisión  
  Asume la responsabilidad general de la revisión; decide quiénes estarán involucrados y organiza cuándo y dónde se llevará a cabo.
* **Revisores**  
  Identifican posibles defectos en el producto de trabajo bajo revisión; pueden representar diferentes perspectivas —por ejemplo,  probador, programador, usuarios, operador, analista de negocio, experto en usabilidad—.
* **Escriba**  
  Recopila los posibles defectos encontrados, puntos abiertos y decisiones en las revisiones.

**TIPOS**

* **Guiada | Walkthrough**  
  Dirigida por el autor del producto de trabajo, el escriba es obligatorio. El uso de listas de comprobación y la preparación individual previa son opcionales. Puede variar de informal a formal. Puede o no haber documentación de defectos potenciales.
* **Técnica**  
  Se realiza entre pares técnicos del autor. Si hay una reunión, debe haber una preparación individual previa. Tienen que existir un facilitador y un escriba obligatoriamente, quien idealmente no será el autor. Se elaboran registros de defectos potenciales e informes de revisión.
* **Inspecciones**  
  Los resultados se documentan formalmente. Requieren preparación individual, tienen roles definidos: autor, director, facilitador, escriba, revisores y líder de revisión, pueden incluir también un lector dedicado. El autor no puede actuar como facilitador, líder de revisión, lector o escriba. Se hace uso de listas de comprobación. Se elaboran registros de defectos potenciales e informes de revisión.

**TÉCNICAS**

* **Ad Hoc**  
  Los revisores leen el producto de trabajo de forma secuencial y a medida que van identificando los defectos, los van documentando, recibiendo poca o ninguna orientación en el proceso.
* **Basadas en escenarios y ensayos**  
  Fundada en el uso esperado del producto de trabajo descrito en un documento, por ejemplo, en un caso de uso.
* **Basadas en listas de comparación**  
  Los revisores detectan defectos a partir de un conjunto de preguntas basadas en defectos potenciales, producto de la experiencia del autor de la lista.
* **Basada en roles**  
  Los revisores evalúan el producto de trabajo desde la perspectiva de usuarios experimentados, inexpertos, adultos, niños o roles específicos en la organización, como un administrador de usuarios, de sistemas o un probador del rendimiento.
* **Basada en perspectiva**  
  Esta técnica es similar a una revisión basada en roles, los revisores adoptan los diferentes puntos de vista del usuario final, del personal de marketing, del diseñador, del probador o del personal de operaciones.

**ACTIVIDADES**

* **Planificar**  
  Definir el alcance, establecer objetivos, roles, tiempo y plazos. Definir y comprobar el cumplimiento de criterios de entrada y salida para revisiones más formales.
* **Iniciar revisión**  
  Distribuir el material e instruir a los participantes.
* **Revisión individual (preparación individual)**  
  Revisión del material y tomar notas de los defectos encontrados.
* **Comunicar y analizar**  
  Comunicar defectos a los responsables.
* **Corregir en informar**  
  Comunicar los defectos potenciales encontrados en una reunión de revisión. Elaborar informes de hallazgos, documentar las características de calidad y comprobar criterios de salida.

**Requisitos**

Una de las revisiones que se realizan en las pruebas estáticas es examinar los requisitos del software. Pero ¿sabemos qué son los requisitos? ¿Qué tipos de requisitos existen?

Un requisito define las funciones, capacidades o atributos intrínsecos de un sistema de software, es decir, describe cómo debe comportarse un sistema. Para decir que un sistema tiene calidad deben cumplirse los requisitos **funcionales y no funcionales**.

**Requisitos funcionales**  
Definen lo que un sistema permite hacer desde el punto de vista del usuario. Estos requisitos deben estar especificados de manera explícita. Por ejemplo: “El campo de monto acepta únicamente valores numéricos con dos decimales” (pruebas funcionales y de sistema).

**Requisitos no funcionales**  
Definen condiciones de funcionamiento del sistema en el ambiente operacional. Ejemplos:

* **Requisito de usabilidad:**la usabilidad se define como el esfuerzo que necesita hacer un usuario para aprender, usar, ingresar datos e interpretar los resultados obtenidos de un software de aplicación (pruebas de usabilidad).
* **Requisito de eficiencia:**relacionado con el desempeño en cuanto al tiempo de respuesta, número de operaciones por segundo, entre otras mediciones; así como consumo de recursos de memoria, procesador y espacio en disco o red (pruebas de rendimiento, pruebas de carga, estrés y escalabilidad, pruebas de gestión de la memoria, compatibilidad e interoperabilidad).
* **Requisito de disponibilidad:**disposición del sistema para prestar un servicio correctamente (pruebas de disponibilidad). Requisito de confiabilidad: continuidad del servicio prestado por el sistema (pruebas de seguridad).
* **Requisito de integridad:**ausencia de alteraciones inadecuadas al sistema (pruebas de seguridad, pruebas de integridad). Requisito de mantenibilidad: posibilidad de realizar modificaciones o reparaciones a un proceso sin afectar la continuidad del servicio (pruebas de mantenimiento y de regresión).