Guia de Aprendizaje Pruebas de Software

Módulo Registro de Usuarios

Desarrollado por:

Jeiffer Harley Mosquera Abadia

Yeraldin Olaya Ríos

Septiembre 2025

Instructor: Jonathan Espitia

SENA

Centro de Gestión de Mercados, Logística y TIC (CGMLTI)

Análisis y Desarrollo de Software (ADSO)

Ficha: 3147246

Bogotá D.C

III. Formulación de las Actividades de Aprendizaje

• Actividades de Reflexión Inicial

- Pregunta detonante: ¿Qué consecuencias puede tener un software que no ha sido adecuadamente probado?

RTA: Cuando las pruebas de software son insuficientes o están mal diseñadas, las consecuencias pueden ser mucho más graves de lo que parece a simple vista. Uno de los efectos más visibles es que el programa se bloquee o se cierre de forma inesperada, justo cuando el usuario más lo necesita. Esto no solo interrumpe la experiencia, sino que puede hacer que se pierda trabajo, tiempo o incluso dinero.

Otra consecuencia crítica son los errores de seguridad. Sin una validación rigurosa, los datos personales o financieros de los usuarios pueden quedar expuestos, abriendo la puerta a vulnerabilidades que podrían ser explotadas por terceros. A esto se suma la pérdida de información: documentos importantes, archivos o configuraciones pueden desaparecer sin previo aviso, generando frustración y desconfianza en la herramienta o en la empresa que la desarrolla.

- Discusión guiada sobre ejemplos reales de fallos de software.

RTA: **1. Mariner 1: La omisión de un guión**

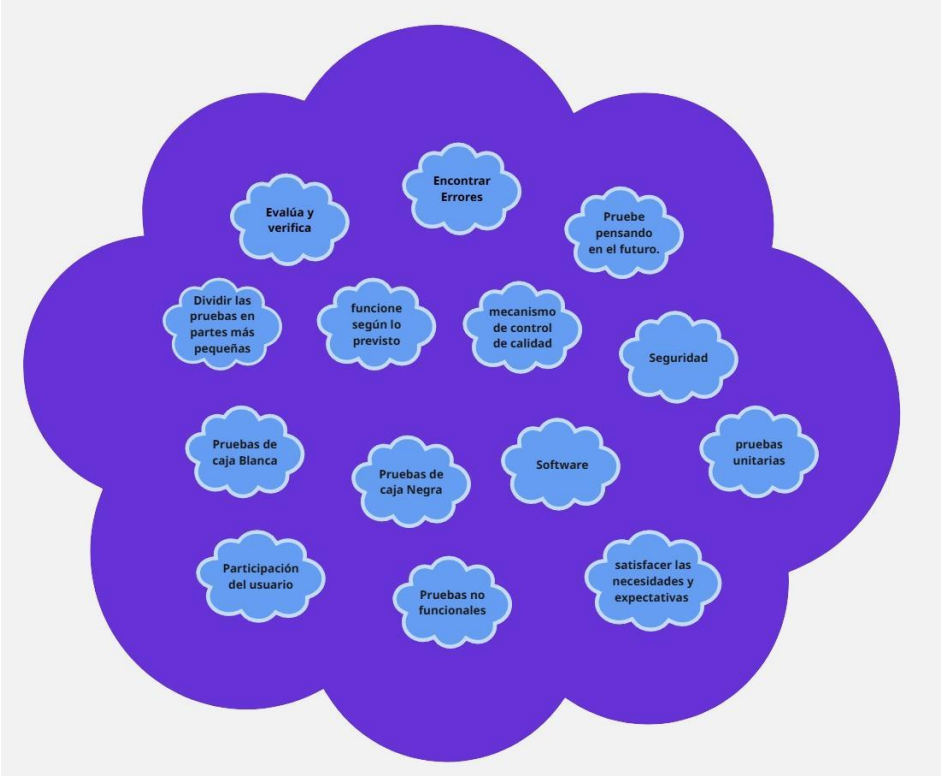
En el año 1962, la NASA se disponía realizar el lanzamiento al espacio de la misión Mariner 1. Con el fin de navegar la órbita de Venus. Sin embargo, un error de programación representó la diferencia entre el éxito y una catástrofe total.

La Mariner 1 fue creada para recabar datos sobre la temperatura y atmósfera de Venus. Esta no logró salir de la atmósfera de la tierra. La catástrofe se originó a solo 5 minutos del despegue debido a la mala transcripción de un código y la omisión de un guión "-".

El fracaso de la Mariner 1 produjo la pérdida de 18,5 millones de dólares a la NASA.

• Actividades de Contextualización e Identificación de Saberes Previos

- Lluvia de ideas: "¿Qué entienden por prueba de software?"



- Socialización de experiencias previas con herramientas de software y errores encontrados.

• **Actividades de Apropiación**

- Lectura dirigida sobre tipos de pruebas de software.

RTA:

Pruebas de integración

Las pruebas de integración verifican que los distintos módulos o servicios utilizados por tu aplicación funcionan bien en conjunto.

Pruebas unitarias

Las pruebas unitarias son de muy bajo nivel y se realizan cerca de la fuente de la aplicación. Consisten en probar métodos y funciones individuales de las clases, componentes o módulos que usa tu software.

Pruebas de extremo a extremo

Las pruebas integrales replican el comportamiento de un usuario con el software en un entorno de aplicación completo. Además, verifican que diversos flujos de usuario funcionen según lo previsto

Pruebas funcionales

Las pruebas funcionales se centran en los requisitos empresariales de una aplicación. Solo verifican el resultado de una acción y no comprueban los estados intermedios del sistema al realizar dicha acción.

Pruebas de aceptación

Las pruebas de aceptación son pruebas formales que verifican si un sistema satisface los requisitos empresariales. Requieren que se esté ejecutando toda la aplicación durante las pruebas y se centran en replicar las conductas de los usuarios.

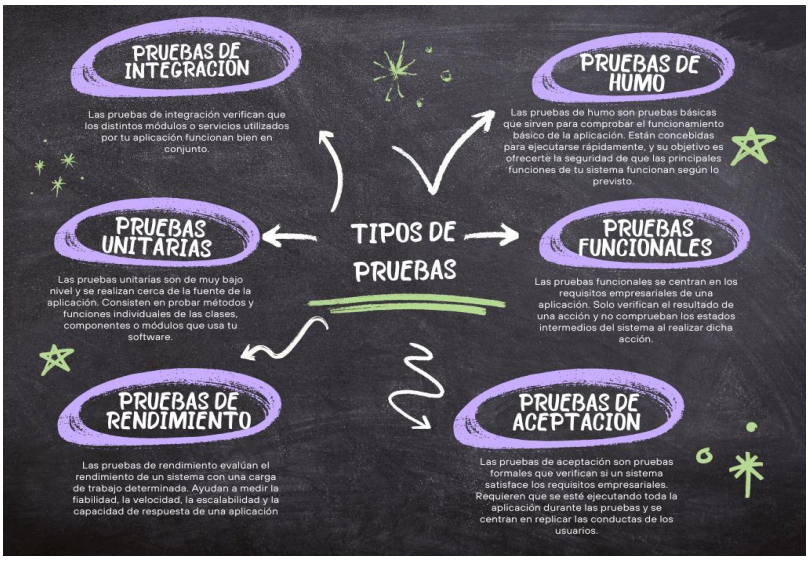
Pruebas de humo

Las pruebas de humo son pruebas básicas que sirven para comprobar el funcionamiento básico de la aplicación. Están concebidas para ejecutarse rápidamente, y su objetivo es ofrecerte la seguridad de que las principales funciones de tu sistema funcionan según lo previsto.

Pruebas de rendimiento

Las pruebas de rendimiento evalúan el rendimiento de un sistema con una carga de trabajo determinada. Ayudan a medir la fiabilidad, la velocidad, la escalabilidad y la capacidad de respuesta de una aplicación

- Investigación individual y grupal de conceptos clave.
- Elaboración de un mapa mental o diagrama conceptual sobre los tipos de pruebas.



IV. Planteamiento de Evidencias de Aprendizaje

- **Fase del Proyecto Formativo:** Construcción del software
- **Actividad del Proyecto Formativo:** Aplicación de pruebas funcionales al módulo de gestión del sistema desarrollado
- **Actividad de Aprendizaje:** Investigación en grupo y/o individual que permita socializar y adquirir conocimientos de la temática propuesta, usando herramientas como mapas mentales, diagramas, etc.

Evidencias:

- **De Conocimiento:** Identificación y definición de los tipos de pruebas de software.
- **De Desempeño:** Participación en actividades grupales e individuales.
- **De Producto:** Mapa mental, diagrama conceptual u otro producto visual sobre tipos de pruebas.

Evaluación:

- **Criterios de Evaluación:** Claridad conceptual, participación activa, calidad del producto entregado.
- **Técnicas e Instrumentos:** Lista de chequeo, observación directa, retroalimentación grupal.

V. Glosario de Términos

- **Prueba de software:** Proceso para evaluar la funcionalidad de una aplicación.
- **Verificación:** Comprobación de que el producto cumple con los requisitos especificados.
- **Validación:** Aseguramiento de que el producto cumple su propósito.
- **Mapa mental:** Herramienta visual para organizar información.

VI. Referentes Bibliográficos

- Sommerville, I. (2011). *Ingeniería del software*.
- ISTQB Foundation Level Syllabus.
- Recursos digitales: blogs especializados, YouTube, documentación de herramientas de testing.

VII. Control del Documento

- **Nombre del autor:** Jonathan David Espitia Rivera
- **Cargo:** Instructor
- **Dependencia:** SENA

- **Fecha de elaboración:** Mayo de 2025

Referencias Bibliográficas

https://infseg.com/informatica/el-arte-de-probar-software-calidad-cofianza-errores/

https://fyccorp.com/10-grandes-errores-de-software/

https://www.testdevlab.com/blog/software-testing-101-definition-types-everything-else