FUNDAMENTOS DE PRUEBAS DE SOFTWARE

OBJETIVOS

- Reconoce los diferentes conceptos que puede encontrar en el tema de pruebas de software en diferentes contextos
- Identifica las relaciones entre los diferentes conceptos
- Entiende los principios de pruebas
- Documenta las pruebas en un proyecto de software

MOTIVACIÓN

Debemos evitar:

- Defectos
- Sobrecostos
- Trabajo innecesario
- Daños en la imagen

Algunas fallas recientes

- 05 Marzo, 2024
  - Instagram & Facebook
  - Login principalmente afectado
  - Horario importante para creadores de contenido

- 26 Enero, 2024
  - Microsoft Teams
  - Afecta envío de mensajes, imágenes, login, entre otros
  - Horario de trabajo, que afecta comunicación de trabajadores.

1

CONCEPTOS GENERALES

Definiciones

ERROR (Bug)

¿Qué es?

DEFECTO (defect)

- Fuera de rangos normales (anomalía)
- Parte de algo más grande
- No corresponde a sus especificaciones

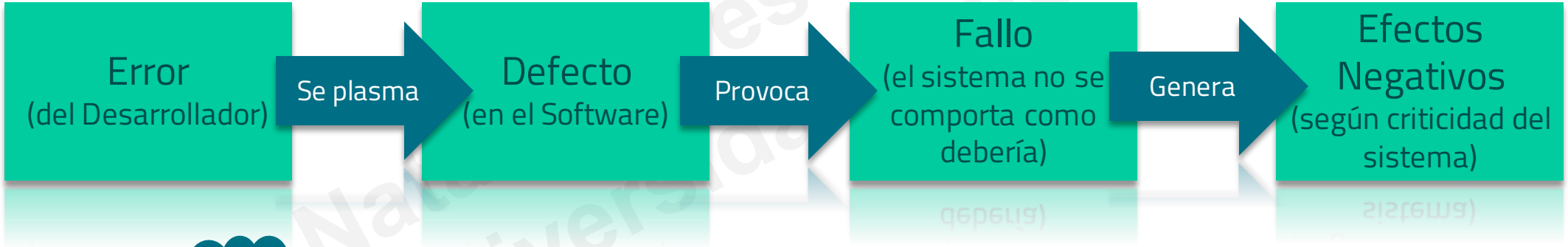
Es el resultado de una deficiencia o error durante la construcción del software.

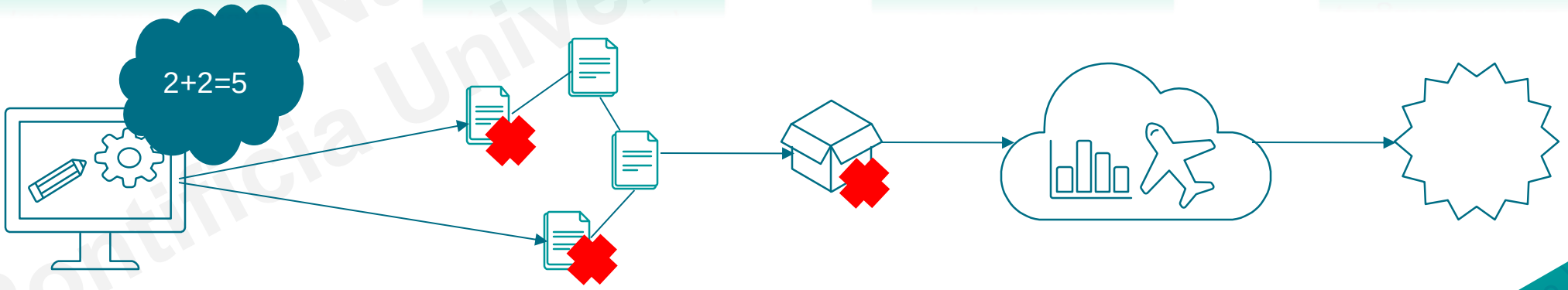
FALLO (failure)

- Resultado no esperado
- Hay (¡o no hay!) un mensaje de error
- Interrupción

Es la incapacidad del software (o una de sus partes) de realizar sus funciones especificadas.

Relación entre ERROR, DEFECTO y FALLO





Sistema Gestión de Aeropuerto

Sistema de Aproximación

ERROR (Bug)

Una equivocación cometida por el desarrollador.

OJO, pueden ser:

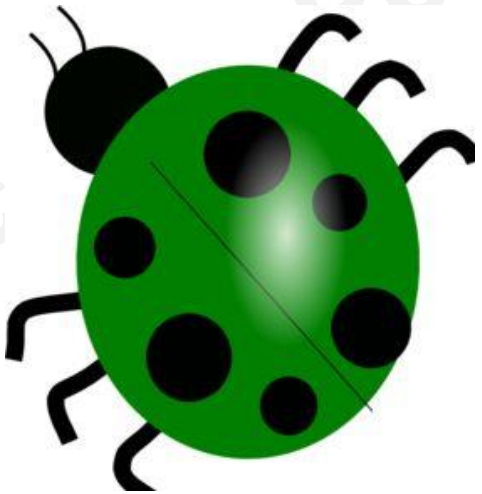
--> De requisitos

--> De diseño

--> De implementación

Errores comunes

- División entre cero
- Ciclo infinito
- Desbordamientos (overflow)
- Condición de carrera
- Variable no inicializada
- Memoria no permitida (Violación de acceso)
- Pérdida de memoria (memory leak)
- Desbordamiento del arreglo
- Desbordamiento de búfer (buffer overflow)
- Bloqueo mutuo (deadlock)
- Indizado inadecuado de tablas en bases de datos



2

VALIDACIÓN Y VERIFICACIÓN

Contrastes

Validación y Verificación

- ¿Construimos el producto correcto?

  Proceso de evaluación del SW para determinar si cumple lo esperado por el cliente.

- ¿Construimos correctamente el producto?

  Proceso de análisis y revisión de que el SW cumple los objetivos establecidos al inicio.

3

PRUEBAS EN EL CICLO DE VIDA DE SOFTWARE

Inspecciones, depuración y pases

INSPECCIÓN

- Enfocarse en el código fuente
- Menos costoso
- Tratar de no encubrir/minimizar errores
- Cumplimiento de estándares
- No reemplaza una prueba

Proceso de revisión del código (u otros elementos) para encontrar defectos.

PRUEBA

- Técnicas de descubrimiento de anomalías.
- Diseñadas y planificadas
- Diferentes clases y en distintos niveles
- Ingreso de datos de prueba.

Proceso que busca demostrar que el sistema hace lo que debe hacer, localizando errores y/o anomalías.

¿Quién debe realizar las inspecciones y pruebas?

¡Depende!

Equipo de desarrollo

- Conoce el funcionamiento del sistema.
- Sesgo al probar
  (evita comportamientos raros)
- Económico (a corto plazo)

Equipo de pruebas

- Personas especializadas en calidad
- Más costoso
- Sin sesgos (más objetivos)
- Paralelizable con desarrollo

Ojo: En cualquiera de los casos pueden automatizarse

DEPURACIÓN

- Eliminar los bugs encontrados.
- Usar las salidas de las pruebas (mensajes, resultados esperados, datos introducidos) para brindar soluciones.
- Realizadas por el equipo de desarrollo.
- Se debe dar un tiempo estimado (explícito) en el cronograma.

Proceso para corregir los errores y problemas encontrados por las pruebas.

PASE A PRODUCCIÓN

- Culminadas las pruebas.
- Pasar del entorno de desarrollo al entorno final.
- Pueden haber entornos de preProducción intermedios.
- Largo o directo dependiendo de la complejidad.

Proceso de instalación, implantación o puesta a disposición de un SW en el ambiente donde debe realizar sus funciones.

4

ASEGURAMIENTO DE CALIDAD Y CONTROL DE CALIDAD

Distinciones

Aseguramiento de Calidad (QA) y Control de Calidad (QC)

	QA	QC
Definición	Aseguran la calidad en los procesos que desarrollan el producto.	Aseguran la calidad del producto.
Enfoque	Prevenir defectos. Es proactivo.	Identificar defectos. Es reactivo.
Objetivo	Mejorar el desarrollo y pruebas para que no surjan defectos al desarrollar.	Identificar defectos al desarrollar pero antes de su entrega.

Aseguramiento de Calidad (QA) y Control de Calidad (QC)

	QA	QC
Cómo	Establecer un sistema de gestión de calidad y revisar que sea adecuado. Auditorías pe riódicas.	Encontrar y eliminar problemas de calidad para asegurar que se cumplan los requerimientos.
Responsable	Todo el equipo encargado del de sarrollo.	Es el equipo encargado de las pruebas.
Tipo de Herramienta	Es una herramienta de gestión	Es una herramienta correctiva / de control

5

GESTIÓN DE LA CONFIGURACIÓN

Relación con las pruebas

Gestión de la Configuración

- Gestión de elementos (y versiones) de un Software.
- Incluye:
  - Código fuente, Scripts de pruebas, SW de terceros
  - Hardware, datos, documentación
  - Parámetros de configuración, entre otros.

¿Cuál es su relación con las pruebas?

Gestión de la Configuración

- Asegurar entrega de la versión correcta de elementos a probar.
- Especificar la configuración de dichos elementos a probar.
- Identificar qué defectos se encuentran en qué versión.

6

DOCUMENTACIÓN DE PRUEBAS

Planes y otros

Plan de Pruebas

- Especifica cómo se deben ejecutar las pruebas.
- Puede incluir:
  - Contexto de las pruebas: Uso principal del sistema, Elementos a probar, Asunciones, restricciones
  - Comunicación: Interesados, forma de comunicación de resultados
  - Riesgos en el desarrollo de las pruebas
  - Calendarización e identificación de responsables en cada actividad de las pruebas
  - Casos de prueba especificados

Otros Documentos

Dependiendo de la organización y/o proyecto se pueden pedir cierta documentación adicional. Algunos documentos según el estándar ISO/IEC/IEEE 29119-3*:

- Plan de pruebas
- Política de prueba
- Especificación de diseño de pruebas
- Especificación de casos de pruebas
- Informe de preparación de datos de prueba / entorno de prueba
- Resultados actuales
- Informe de incidentes de prueba

* ISO/IEC/IEEE 29119 es una serie de cinco estándares internacionales para pruebas de software

7

PRINCIPIOS DE PRUEBAS

Los 7 Principios de Pruebas

En los últimos años, se aceptan como válidos estos 7 principios de las pruebas:

- **Principio 01: Las pruebas sirven para demostrar defectos**

  Cuando probamos, sólo estamos mostrando los defectos que existen, pero no aseguramos que ellos no existan. Pueden existir en algo que no se probó.

Los 7 Principios de Pruebas

- **Principio 02: Las pruebas exhaustivas son imposibles**

No es posible realizar todas las pruebas con todos los valores posibles. Se requeriría una cantidad astronómica de casos de pruebas. Los casos son sólo una muestra, por lo que los esfuerzos se deben controlar y priorizar.

Los 7 Principios de Pruebas

- **Principio 03: Las pruebas deben empezar lo más pronto posible**

Se debe empezar a probar lo más pronto que se pueda en el ciclo de vida del Software. Eso ayuda a detectar los defectos antes que se distribuyan en más elementos.

Los 7 Principios de Pruebas

- **Principio 04: Los defectos se aglutinan**

Los defectos no suelen distribuirse homogéneamente en el sistema. Si se encuentran varios defectos en una parte del Software, es probable que haya más en partes cercanas. Suele suceder que, cuando hay más complejidad, se generan "clústers" de defectos.

Los 7 Principios de Pruebas

- **Principio 05: La paradoja del pesticida**

  Así como los insectos y bacterias se hacen resistentes, el software se hace resistente a las pruebas que no son mantenidas. Si se prueba siempre lo mismo, eventualmente ya no se encuentran defectos. Hay que ir nutriendo la batería con nuevos casos (o actualizar los antiguos), según se van desarrollando nuevas funcionalidades.

Los 7 Principios de Pruebas

- **Principio 06: Las pruebas dependen del contexto**

No se pueden probar las mismas cosas en diferentes sistemas. Cada sistema tendrá sus propios criterios de pruebas, de exhaustividad de acuerdo a su criticidad y uso. El contexto puede incluir los riesgos, los tipos de usuario, los tipos de dispositivos, el uso lúdico o profesional, el uso para atención a otras personas, entre otros.

Los 7 Principios de Pruebas

- **Principio 07: La ausencia de errores es una falacia**

  Encontrar errores y repararlos no indica que el sistema cumplirá todas las expectativas del cliente. Tampoco que no habrá ningún defecto u error en el software. El proceso de pruebas disminuye la posibilidad de que suceda pero no lo evita.

8

MINI-CASOS

Aplicando

Mini-caso 01

En qué consistiría el pase a producción de un app móvil? (Asuma que es auto-contenida, es decir, no utiliza servicios externos y/o de un servidor propio)

- A: En realizar pruebas de aceptación.
- B: En ponerla a disposición en el marketplace correspondiente.
- C: En depurar sus errores.
- D: En integrar sus módulos.

¿Qué etapa de desarrollo debo haber terminado antes de que se realice?

Mini-caso 02

Un equipo de desarrollo acaba de terminar de implementar un sistema para una cadena de pastelería, que permite manejar el inventario de productos para elaborar la cobertura de chocolate. Al terminar la demo de forma satisfactoria, los clientes, sin embargo, les comentan que el sistema que buscaban era el que manejara la cobertura de los seguros que tienen contratados en sus tiendas. ¿Qué proceso falló?

- A: La Verificación
- B: La Validación
- C: El Pase a Producción
- D: La Depuración

Mini-caso 03

Una startup dedicada al desarrollo de una aplicación de envíos tiene una política para incluir una nueva funcionalidad, que exige:

- Una serie de documentos establecidos.
- La funcionalidad debe encontrarse versionada.
- Se debe haber superado los casos de prueba.

Esta política es un ejemplo de:

- A: Gestión de la Configuración
- B. Control de Calidad
- C. Aseguramiento de la Calidad
- D. Desarrollo Iterativo

> Las pruebas sólo pueden mostrar la presencia de errores, mas no su ausencia.

(Dijkstra et al., 1972)

9

REFERENCIAS

BIBLIOGRAFÍA

- Sommerville, I. (2011). Software engineering (ed.). America: Pearson Education Inc.
- ISO/IEC/IEEE 29119-1:2022 Software and systems engineering https://www.iso.org/standard/81291.html
- Spiller, A. Linz, T. & Shaefer, H. (2014) Software Testing Foundations (4th Ed.) O'Reilly Media

Créditos:

- Plantilla de la presentación por SlidesCarnival
- Diseño del fondo Hero Patterns