Martín Josué Andrade Salazar – 219737144

Maestría en Ingeniería de software

Descripción breve

El presente documento muestra 10 sucesos historicos consecuencia de fallas en el software

Pruebas de software

Prueba y mantenimiento de software II

Logotipo

Descripción generada automáticamente

Tabla de contenido

[Introducción 1](#_Toc174897635)

[Desarrollo 2](#_Toc174897636)

[Suceso 1: El accidente del vuelo 501 de Ariane 5 (1996) – Kourou, Guayana Francesa. 2](#_Toc174897637)

[Suceso 2: El error del año 2000 (Y2K) (1999-2000) – Problema global 3](#_Toc174897638)

[Suceso 3: El desastre de Therac-25 (1985-1987) - Estados Unidos y Canadá 3](#_Toc174897639)

[Suceso 4: El fallo del software de AT&T (1990) – EE. UU. 4](#_Toc174897640)

[Suceso 5: El fallo del sistema bancario del Banco TSB (2018) – Reino Unido 4](#_Toc174897641)

[Suceso 6: El error del sistema de control del Mariner 1 (1962) 4](#_Toc174897642)

[Suceso 7: El fallo del software del USS Yorktown (1997) – EE. UU 5](#_Toc174897643)

[Suceso 8: El gusano de Morris (1988) – Universidad de Cornell 5](#_Toc174897644)

[Suceso 9: Error de cálculo de Intel (1994) – Universidad e Lynchburg 6](#_Toc174897645)

[Suceso 10: Falla en global de Windows 19 de Julio (2024) 6](#_Toc174897646)

[Conclusión 7](#_Toc174897647)

[Referencias 7](#_Toc174897648)

# **Introducción**

En [1] se dice que las pruebas de software son el proceso de evaluar y verificar que un producto o aplicación de software hace lo que se supone que debe. El realizar estas pruebas de software aseguran la prevención de errores y la mejora del rendimiento.

Las pruebas de software hoy en día son más efectivas cuando son continuas, lo que indica que las pruebas se inician durante el diseño, continúan a medida que se crea el software e incluso se producen cuando se despliega en producción [1].

Hay muchos tipos de técnicas de pruebas de software que se pueden llevar a cabo para asegurar de que los cambios en el código funcionen según lo esperado. Sin embargo, no todas las pruebas son iguales [2]. Cada una nos ayuda a asegurar la calidad en cada apartado del software que se está desarrollando. En este artículo veremos 10 sucesos históricos catastróficos que han sido consecuencia de fallas en el software.

# **Desarrollo**

# **Suceso 1: El accidente del vuelo 501 de Ariane 5 (1996) – Kourou, Guayana Francesa.**

Un 4 de junio, le majestuoso cohete Ariane 5 fabricado por la Agencia Europea realizaba un vuelo inaugural.

Los técnicos realizaron la cuenta atrás, se encendieron los propulsores y el gigantesco ingenio espacial comenzó a elevarse por los cielos. A los pocos segundos dio la impresión de estarse inclinando hacia debajo de forma extraña, para un instante después hacer un requiebro descontrolado y explotar en mil pedazos.

El equipo de CSI de la tecnología aeroespacial examinó los restos, esparcidos por la zona de lanzamiento. Según el informe oficial, a los 40 segundos del lanzamiento y 3.700 metros de altitud “la lanzadera Ariane se desvió de sui ruta, se partió y explotó”. Lo más increíble fue leer el detalle de la descripción del fallo, que convenientemente destilado apuntaba a un “bug” del software que guiaba el cohete.

El problema había surgido al convertir un número almacenado en algo que los programadores llaman “Coma flotante de 64 bits en entero de 16 bits. Los números de coma flotante son otra forma de decir números decimales. Y lo que sucedió fue algo así como convertir el valor ‘32790,789’ en un número entero en vez de ‘32790’ el resultado fuera ‘0’, ‘-500’ o un error similar (en los números enteros de 16 bits el valor máximo es 32767).

El resultado del incidente fueron muchas lecciones aprendidas, unos 500 millones de dólares en carga perdidos y meses de desarrollo desaprovechados, a cambio de unos curiosos y carísimos ‘fuegos artificiales’ [3].

# **Suceso 2: El error del año 2000 (Y2K) (1999-2000) – Problema global**

El bug del milenio se refería a un problema informático que surgiría cuando los relojes y sistemas informáticos que almacenaban fechas de dos dígitos (como ‘99’ para 1999) cambiaran al año 2000. La preocupación radicaba en que los sistemas interpretarían “00” como 1900 en lugar de 2000, lo que podría causar errores en los cálculos y malfuncionamientos en cascada. A medida que se acercaba el 31 de diciembre de 1999, creció el temor de que el cambio de milenio desencadenaría fallas en sistemas críticos, como servicios financieros, transporte, energía y comunicaciones. Gobiernos, empresas y organizaciones de todo el mundo invirtieron millones de dólares en la corrección de códigos y pruebas exhaustivas. Equipos de expertos se desplegaron para mitigar los riesgos y garantizar que el paso al año 2000 fuera lo más suave posible [4].

A pesar de la anticipación del apocalipsis tecnológico, el Bug del Milenio resultó ser menos catastrófico de lo temido. El evento destacó la importancia de la preparación y la colaboración global en el ámbito de la tecnología. Aunque y2K no trajo el fin del mundo, dejó una marca indeleble en la historia de la informática y la conciencia pública. Algunas de las consecuencias fueron:

* Gastos masivos en actualización de sistemas
* Interrupciones menores
* Impacto en la economía global

# **Suceso 3: El desastre de Therac-25 (1985-1987) - Estados Unidos y Canadá**

Uno de los fallos más grandes de software que se recuerdan fue el que afectó al acelerador línea médico modelo Therac-25. Esta máquina que costaba 1 millón de dólares se usaba para tratar a pacientes de cáncer con radio terapia y debido a un error de software costo la vida a 6 personas entre los años 85 y 87. Este modelo, fabricado conjuntamente por la empresa AECL (Atomic Energy of Canada Limited) y CGR (una empresa francesa) era la versión modernizada de su anterior y popular modelo el Therac-20. Estas máquinas aceleran los electrones y los enfocan en los tumores para destruirlos.

Para penetración de baja intensidad en tejidos se usaban haces de electrones, y para alcanzar tumores más internos el haz se convertía en rayos-X. Para ello se lanzaba un haz de electrones de alta potencia (¡25 millones de electron-voltios!) y se insertaba una placa metálica (de número atómico alto, tipo Tungsteno) entre el haz y el paciente.

Lo que ocurrió es que, debido a un fallo en el software de control, en ocasiones se exponía a los pacientes al haz de alta intensidad sin la protección intermedia de la placa metálica, causando una exposición a dosis letales de radiación (100 veces mayores de lo esperado) y acabando con la vida de estos meses más tarde [5].

Algunas de las consecuencias fueron:

* Muertes y lesiones graves

# **Suceso 4: El fallo del software de AT&T (1990) – EE. UU.**

A las 2:39 del 15 de enero de 1990, el Centro de Operaciones de AT&T de Nueva Jersey, USA, detectaba que su red de ordenadores era incapaz de gestionar las llamadas de su red telefónica. EL colapso bloqueó 50 millones de llamadas y duró nueve horas, durante las cuales el país entró en pánico.

La red estaba formada por 114 conmutadores computarizados capaces de gestionar 700.000 llamadas por hora. El proceso estándar de detección, reserva y verificación de calidad de línea llevaba solo unos segundos. Si en este proceso el sistema detectaba un error en uno de los interruptores, este debía reiniciarse y enviar un mensaje al resto de que no recibiría más llamadas hasta nuevo aviso. El software contenía funciones de auto reparación que aislaban los interruptores defectuosos [6].

Algunas de sus consecuencias fueron:

* Perdida de ingresos significativas para AT&T
* Alteraciones masivas en las comunicaciones en EE. UU

# **Suceso 5: El fallo del sistema bancario del Banco TSB (2018) – Reino Unido**

Los errores en ese caso fueron generalizados y graves, y esto tuvo un impacto real en la vida diaria de una parte importante de los clientes de TSB. TSB actualizó sus sistemas informáticos en abril de 2018 y trasladó los datos de sus clientes a una nueva plataforma, que registró fallos técnicos y perturbó los servicios en las sucursales, telefónicos y en línea. A causa de ellos, muchos clientes no pudieron acceder a sus servicios bancarios [7].

Algunas de las consecuencias fueron:

* TSB pago 39,7 millones de dólares a sus clientes afectados.
* Pérdidas económicas significativas
* Daño considerable a su reputación

# **Suceso 6: El error del sistema de control del Mariner 1 (1962)**

corría el año 1962. La NASA se preparaba para lanzar al espacio la misión Mariner 1. Iba a ser la primera vez que el ser humano lograba enviar una sonda no tripulada a dar la vuelta alrededor del planeta, y el elegido para ese honor era Venus.

El objetivo de la [Mariner 1](https://en.wikipedia.org/wiki/Mariner_1) era circunnavegar Venus y recabar datos sobre su temperatura y la composición de su atmósfera, pero la sonda nunca llegó a abandonar la atmósfera terrestre. Apenas cinco minutos después de su lanzamiento, el centro de control de Cabo Cañaveral emitió una orden de autodestrucción que hizo trizas el cohete y la sonda que viajaba en él.

El escritor de ciencia ficción Arthur C. Clarke lo definió como el guion más caro de la historia, pero en realidad no era un guion, sino un guion alto o superguion. El programador que transcribió el código que regía uno de los sistemas de navegación se comió este símbolo [8].

Algunas consecuencias fueron:

* Perdida del vehículo espacial
* Perdida de 18.5 millones de dólares

# **Suceso 7: El fallo del software del USS Yorktown (1997) – EE. UU**

El incidente tuvo lugar en 1997, cuando el portaaviones estadounidense USS Yorktown perdió el control de su sistema de propulsión debido a la desconexión de todas las máquinas del sistema. Resultó que el administrador del sistema había introducido un valor 0 en la base de datos para uno de los parámetros al intentar solucionar una válvula de combustible, por lo que cuando los sistemas intentaron utilizar esta variable, provocó la división por 0. La nave se detuvo en el agua y se necesitaron dos horas y cuarenta y cinco minutos para reiniciar los sistemas y hacerla funcionar de nuevo [9].

Algunas consecuencias fueron:

* Pérdida Temporal de Control de la Nave
* Costos Financieros y Operativos
* Críticas al Programa de Automatización

# **Suceso 8: El gusano de Morris (1988) – Universidad de Cornell**

En 1988 un estudiante de posgrado lanzó lo que debería haber sido un inofensivo experimento para medir el tamaño de internet. Sin embargo, lo que resultó fue una infección masiva que desencadenó el caos en la incipiente red de redes. Este experimento, más tarde conocido como el gusano Morris, marcó el primer caso documentado de un gusano informático.

El gusano Morris se propagó explotando vulnerabilidades en el protocolo de transferencia de archivos (FTP) y en programas de correo electrónico. En lugar de dañar archivos o sistemas, su objetivo era replicarse a sí mismo y propagarse a través de la red. La propagación del gusano se basaba en dos métodos principales [10]:

* Vulnerabilidad de contraseñas: El gusano intentaba adivinar contraseñas débiles de cuentas de usuario en sistemas Unix utilizando un diccionario de contraseñas conocidas. Cuando obtenía acceso, se instalaba y comenzaba a buscar nuevos *hosts* para infectar.
* Vulnerabilidad de Sendmail: El gusano Morris también aprovechaba una vulnerabilidad en el programa de correo electrónico Sendmail. Al explotar esta vulnerabilidad, se abría un camino para la infección a través del correo electrónico, multiplicando exponencialmente su propagación.

Algunas consecuencias fueron:

* Saturación de redes
* Dificultades en a identificación
* Daño colateral

# **Suceso 9: Error de cálculo de Intel (1994) – Universidad e Lynchburg**

En 1994 el profesor Thomas Nicely descrubrió un error en la unidad de coma flotante de algunos de los procesadores Intel Pentium. Nicely observó que algunas operaciones de división devolvían siempre un valor erróneo por exceso.

Algunas consecuencias fueron [11]:

* Con un número estimado de entre 3.000.000 y 5.000.000 de chips defectuosos en circulación, el error cuesta finalmente a Intel unos 475.000.00 dólares.

# **Suceso 10: Falla en global de Windows 19 de Julio (2024)**

El incidente fue causado por un error en una actualización de CrowdStrike, una plataforma de ciberseguridad. Esta falla afecta a aproximadamente 8,5 millones de dispositivos Windows, según estimaciones de la compañía.

Aunque [Microsoft](https://www.infobae.com/tecno/2024/07/19/falla-global-de-microsoft-y-crowdstrike-tres-recomendaciones-a-tener-en-cuenta-si-aparece-una-pantalla-azul/)afirma que esto representa “menos del uno por ciento de todas las máquinas Windows”, el impacto fue significativo, causando problemas a minoristas, bancos, aerolíneas y muchas otras industrias, así como a todos los que dependen de estos servicios.

Algunas consecuencias fueron:

* Perdida de usabilidad de muchos sistemas, entre ellos:
  + Software de aeropuertos
  + Sistemas ferroviarios
  + Bancos
  + Organizaciones de atención médica
  + Hoteles
  + Estaciones de televisión

Entre los aeropuertos impactados se encontraban el Aeropuerto Internacional de Los Ángeles (LAX), el Aeropuerto Internacional de Heathrow en Londres, el Aeropuerto Internacional de Frankfurt, el Aeropuerto Internacional de Sídney, el Aeropuerto Internacional de Hong Kong, el Aeropuerto Internacional de Dubai y el Aeropuerto Internacional de Tokio Narita [12].

# **Conclusión**

En conclusión, se sabe y prueba que las pruebas en el software son importantes para asegurar la calidad, usabilidad y satisfacción del usuario que lo usará. Un software mal desarrollado contrae muchas consecuencias, entre ellas, perdida de recursos, mala gestión de la información y en el peor de los casos, perdida de la información. Al llevar a cabo pruebas al software desarrollado nos damos cuenta de los errores y de la forma en que operará en cualquier escenario, al llevar a cabo estas pruebas y corrigiendo los errores generados por las mismas, el software que se le entrega al cliente final es de calidad, usable y sobre todo seguro.

Escribir un software es como hacer una construcción notable. No es necesario cometer errores en cualquier ladrillo y toda la estructura debe ser la fuerte. La automatización de pruebas consiste en la comprobación del sitio que nos permite encontrar y eliminar cualquier grieta o defecto incluso antes de que surjan como problemas reales. Al igual que el arquitecto utiliza un plano para echar un primer vistazo al edificio, la revisión del código sirve como paso necesario de validación del diseño en el ciclo de desarrollo del software. El concepto de integración continua es el de una cadena de producción en la que siempre se comprueba cada pieza, y sólo se ensamblan y prueban las que funcionan juntas.

# **Referencias**

[1] <https://www.ibm.com/mx-es/topics/software-testing>

[2] <https://www.atlassian.com/es/continuous-delivery/software-testing/types-of-software-testing>

[3] <https://www.rtve.es/noticias/20140604/error-software-convirtio-lanzamiento-espacial-carisimos-fuegos-artificiales/948262.shtml>

[4] <https://adises.com.mx/el-bug-del-milenio-el-fin-del-mundo-en-el-ano-2000/#:~:text=El%20Bug%20del%20Milenio%20se,1999)%20cambiaran%20al%20a%C3%B1o%202000>

[5] <https://www.campusmvp.es/recursos/post/GAMBADAS-Therac-25-la-maquina-de-radiacion-asesina.aspx#:~:text=Uno%20de%20los%20fallos%20m%C3%A1s,los%20a%C3%B1os%201985%20y%201987>

[6] <https://www.mtp.es/blog/testing-software/qabalgando-por-la-historia-iv-att-en-1990-el-gran-colapso-de-la-red-a-larga-distancia/>

[7] <https://www.france24.com/es/minuto-a-minuto/20221220-el-banco-brit%C3%A1nico-tsb-multado-con-casi-49-millones-de-libras-por-un-fallo-inform%C3%A1tico>

[8] <https://es.gizmodo.com/como-una-sola-errata-de-codigo-le-costo-a-la-nasa-150-m-1827097911>

[9] <https://qawerk.es/blog/errores-de-programacion-que-cuestan-vidas-tiempo-y-dinero/#:~:text=USS%20Yorktown,-El%20incidente%20tuvo&text=Result%C3%B3%20que%20el%20administrador%20del,provoc%C3%B3%20la%20divisi%C3%B3n%20por%200>

[10] <https://keepcoding.io/blog/que-es-el-gusano-morris/>

[11] <https://testeandosoftware.com/pentium-fdiv-bug-error-de-division-en-coma-flotante/>

[12] <https://www.infobae.com/tecno/2024/07/20/falla-global-de-microsoft-afecto-a-casi-9-millones-de-dispositivos-windows-en-el-mundo/>