Caso Ariane 5 (vuelo 501):

THERAC-25: FALLAS DE SOFTWARE EN SISTEMAS CRÍTICOS

EL SERVICIO DE AMBULANCIAS DE LONDRES

Hecho por:
David Martínez Cara
Mateo Daniel Ortuño Ovando

caso Ariane 5 (vuelo 501):

El cohete Ariane 5 explotó 40 segundos después de su lanzamiento en 1996, provocando una pérdida de 500 millones de dólares.

La Causa Principal: Un Error de Software

- Fallo técnico: Un "desbordamiento" (overflow) al convertir un número demasiado grande (de 64 bits) a un formato más pequeño (16 bits).

- El Ariane 5 era más rápido que su predecesor, por lo que los números superaron el límite del sistema.

EN QUE FASE OCURRIO EL FALLO?

El principal problema, ocurrio en tres fases del desarrollo de software:

Fase de analisis: No se detectaron correctamente las nuevas necesidades del Ariane 5 respecto al Ariane 4

Fase de DISEÑO: No se incluyeron mecanismos de control o validación para evitar el desbordamiento numérico.

Fase de PRUEBAS: Las pruebas fueron insuficientes: no se simularon las condiciones reales del Ariane 5, por lo que el error nunca se manifestó antes del vuelo.

THERAC-25: FALLAS DE SOFTWARE EN SISTEMAS CRÍTICOS

Contexto:

- Máquina de radioterapia controlada por computadora (AECL, 1980s).
- Tres modos: campo luminoso, electrones y fotones (rayos X).
- Reemplazó los mecanismos físicos de seguridad por software.

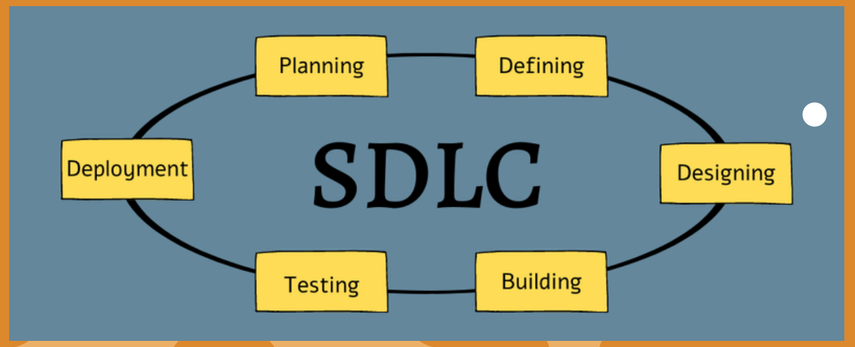
El problema:

- Si el rayo de fotones se activaba sin el atenuador, el paciente recibía una sobredosis letal.
- En varios casos, el sistema emitió dosis 100 veces mayores a las terapéuticas.
- Error clave: “Malfunction 54” → software no reconocía cambios rápidos del operador.

Consecuencias:

- Múltiples accidentes, muertes y lesiones graves.
- Fallas de comunicación y respuestas inadecuadas del fabricante.

THERAC-25: FALLAS DE SOFTWARE EN SISTEMAS CRÍTICOS



Causas principales:

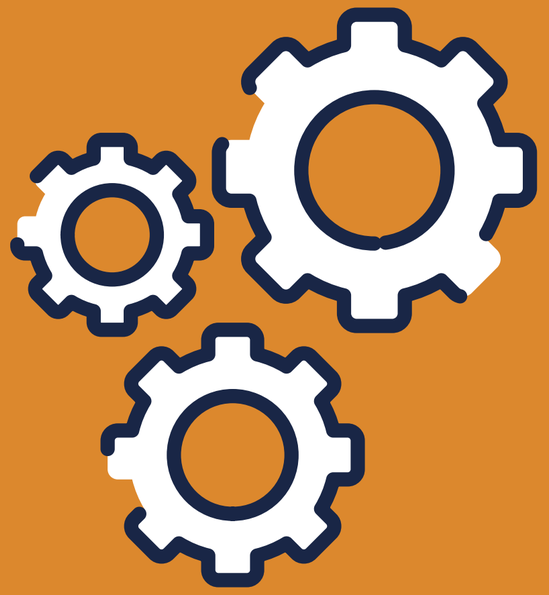
- Fase de diseño: eliminación de seguridad física, dependencia total del software.
- Fase de pruebas: validación deficiente, errores no detectados.
- Mantenimiento: respuestas tardías, soluciones superficiales.

Lecciones aprendidas

- Nunca confiar la seguridad solo al software.
- Incluir redundancia física y verificación independiente.
- Realizar pruebas exhaustivas e integrar factores humanos.
- La seguridad del usuario debe tener prioridad sobre la usabilidad.
- Gestión ética y transparente de fallas en sistemas críticos.

El Servicio de Ambulancias de Londres

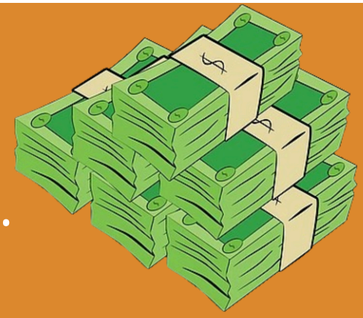
- El SAL atendía 5000 pacientes al día con 750 ambulancias.
- En 1990 inició un proyecto para automatizar el despacho con un **Sistema Asistido por Computadora (DAC)**.
- Objetivo: optimizar tiempos de respuesta y coordinación de emergencias.



- Se intentó pasar de lo manual a lo totalmente automatizado en un solo paso.
- No se consideraron etapas intermedias ni pruebas piloto.
- Los riesgos fueron subestimados desde el inicio.

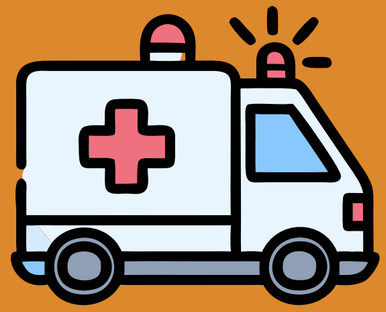
Problemas de Planificación

- Se aceptó la oferta más barata (1 millón de libras).
- El calendario fue de solo 11 meses, pese a las advertencias.
- La empresa elegida no tenía experiencia en proyectos de esta magnitud.



- Operadores y paramédicos no fueron consultados en la fase de diseño.
- El sistema no se adaptaba a sus rutinas reales.
- La falta de capacitación y resistencia al cambio generó caos operativo.

- Asignaba ambulancias solo por cercanía, sin considerar conocimiento del área.
- Los equipos terminaban lejos de su base y perdían tiempo.
- La centralización excesiva eliminó la flexibilidad local.



- Un pequeño error de programación causó pérdida de memoria en el sistema.
- Después de tres semanas, el sistema colapsó completamente (4 nov 1992).
- Fue el desencadenante del fracaso total del proyecto.