

# Testing de Software

## V&V

Alejandro Mauricio González

# Software fuera de errores?

- Has tenido alguna experiencia de error de Software?
  - Corel, Windows, Eclipse, Office, OpenOffice...
  - Facturas, telefonía, Mail, aeronáutica,
- La capacidad para medir la fiabilidad del software es muy inferior vs otras ingenierías.
- La imposibilidad de aplicar métodos matemáticos rigurosos → verificación empírica (errores detectados al final, se verifica solo una parte).
- Dada la entidad no física del software, los defectos en los programas son inherentes a su naturaleza (imposibilidad práctica).



# Naturaleza del Software

- La depuración de los sistemas software obedece a la ley del rendimiento decreciente (+ costo vs mejoras del SF ).
- La fiabilidad de un software no se mide por la cantidad de faltas que tiene un programa, sino por el tiempo medio entre fallos.
- El objetivo de las técnicas de evaluación del software no es tanto la eliminación total de las faltas existentes en los programas, como la eliminación de las faltas que provocan fallos frecuentes.
- Si se persevera durante muchísimo tiempo en la depuración de un software, acabamos descubriendo faltas que producirán fallos tan infrecuentes que su enmienda no incide en la fiabilidad percibida del sistema.

# Fallas quinquemilenarias

- El software más depurado y considerado de alta fiabilidad contiene defectos remanentes.
- Edward N. Adams de IBM analizó los “tamaños” de las faltas en una BD de cobertura mundial que suponía el equivalente de miles de años de uso de un sistema informático particular. El descubrimiento más extraordinario consistió en que alrededor de la tercera parte de las faltas contenidas en un programa son quinquemilenarias (producirían un fallo una vez cada 5,000 años)
- Las faltas responsables de fallos más frecuentes habían sido descubiertas y eliminadas durante la fase de evaluación y en los primeros meses de operación del sistema.
- Emplear tiempo en detectar faltas que producen fallos cada más allá de 75 años es malgastar recursos.

# ¿Cuántos errores son lo “normal” por Sistema de Software?

- SEI, un programador experto introduce un defecto por cada 10 LOC; si se detectasen el 99% de los defectos introducidos (siendo optimistas) aún permanecerían 1 defecto por KLOC.
- Sistemas operativos. La tasa de defectos de Linux es 0.1 defectos/KLOC. Las distintas versiones de Unix van de 0.6 a 0.7 defectos/KLOC. Con interfaz gráfica como Windows 95 o MacOS poseían una tasa tan elevada que fallaban cada tres horas o menos, en funcionamiento ininterrumpido.
- Siemens sufrió 6-15 defectos /KLOC en el desarrollo de uno de sus sistemas operativos.
- Unisys 2 a 9 defectos/KLOC en el desarrollo de software de comunicaciones.
- IBM en el desarrollo normal de software, tiene una tasa de 30 defectos/KLOC.
- La variabilidad entre unos casos impide sacar reglas sobre qué es “normal”. •

# ¿Cuántos errores son lo “normal” por Sistema de Software? (continuación...)

- Incluso utilizando técnicas avanzadas (utilizadas en sistemas de alto riesgo [Cleanroom Development]) es imposible lograr un software totalmente libre de defectos.
- IBM no consiguió bajar su tasa de defectos de 2.3-3.4 defectos/KLOC utilizando la técnica Cleanroom.
- La UK Civil Aviation 0.81 defectos/KLOC en el desarrollo del sistema de control de tráfico aéreo del Reino Unido.
- Un defecto puede considerarse el límite inferior de la tasa de defectos alcanzable.
- La NASA ha sufrido tasas de defectos en el rango 4-12 defectos /KLOC.
- Existen autores que afirman que es habitual encontrar en el software comercial entre 25-30 defectos/KLOC. Aplicando una visión exigente tenemos que:
  - Lo mejor que se puede conseguir es 0.5-1 defectos/KLOC .
  - Software comercial es esperable encontrar entre 3-6 defectos/KLOC.
  - Se considera un software de alta calidad cuando tiene menos de 15 defectos/KLOC.

# ¿Qué hacer?

- Lo recomendable es que el producto software vaya siendo evaluado a medida que se va construyendo.
- Es necesario llevar cabo en paralelo al proceso de desarrollo, un proceso de evaluación o comprobación de los distintos productos o modelos que se van generando, en el que participaran desarrolladores y clientes.