**SOFTWARE E INGENIERIA DEL SOFTWARE**

La invención de la tecnología puede tener efectos profundos e inesperados en otras tecnologías con las que en apariencia no tiene ninguna relación. A este fenómeno de lo denomina “la ley de las consecuencias imprevistas”.

Por ejemplo nadie podía haber predicho que el software se convertiría en una tecnología indispensable en los negocios, la ciencia y la tecnología; tampoco que permitiría la creación de nuevas tecnologías.

**El papel evolutivo del software**

El software tiene el papel de un producto y un vehículo mediante el cual se entrega un producto.

Sin importar el lugar en que resida el software, este es un transformador de información; realiza la producción, el manejo, la adquisición, la modificación, el despliegue o la transmisión de la información.

En su papel de vehículo para la entrega de un producto, el software actúa como la base para el control de la computadora (sistemas operativos), la comunicación de información (redes), y la creación y el control de otros programas (utilerías de software y ambientes).

El software entrega el producto más importante de nuestro tiempo: información.

**Software**

El software es el elemento lógico de un sistema. Tiene características muy diferentes a las del hardware, como ser:

* Se desarrolla o construye, no se manufactura.

Existen similitudes entre el desarrollo del software y la manufactura del hardware, pero son diferentes. En ambas, la alta calidad se alcanza por medio del buen diseño, pero la fase de manufactura del hardware puede incluir problemas de calidad inexistentes en el software.

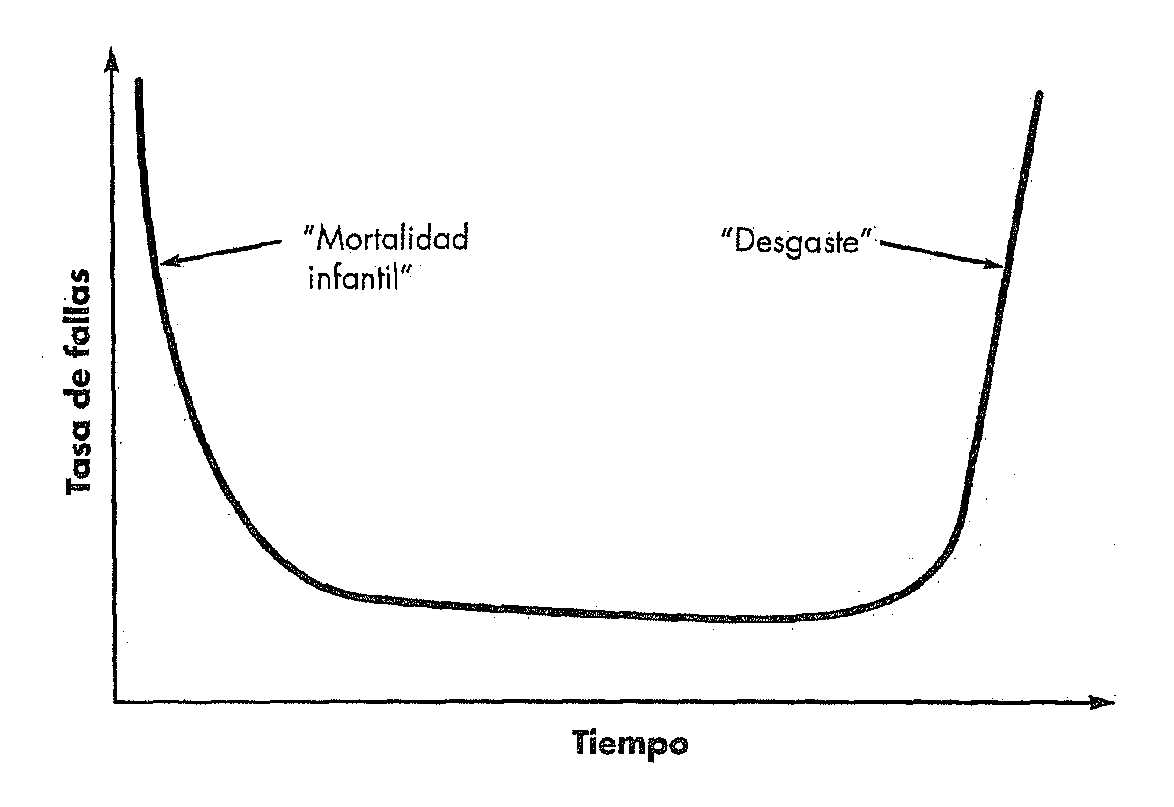
Ambas actividades requieren la construcción de un “producto”, pero los enfoques son diferentes.

Los costos del software se concentran en la ingeniería. Esto significa que los proyectos de software no se pueden manejar como si fueran proyectos de manufactura.

* No se desgasta pero se deteriora.

Curva de la bañera para el hardware:

Para el hardware la tasa de fallas se muestra en función del tiempo, al inicio de su vida tiene un alto número de fallas, los defectos se corrigen y la tasa de fallas baja hasta un nivel estable. Conforme pasa el tiempo la tasa de fallas se elevan de nuevo debido al desgaste.

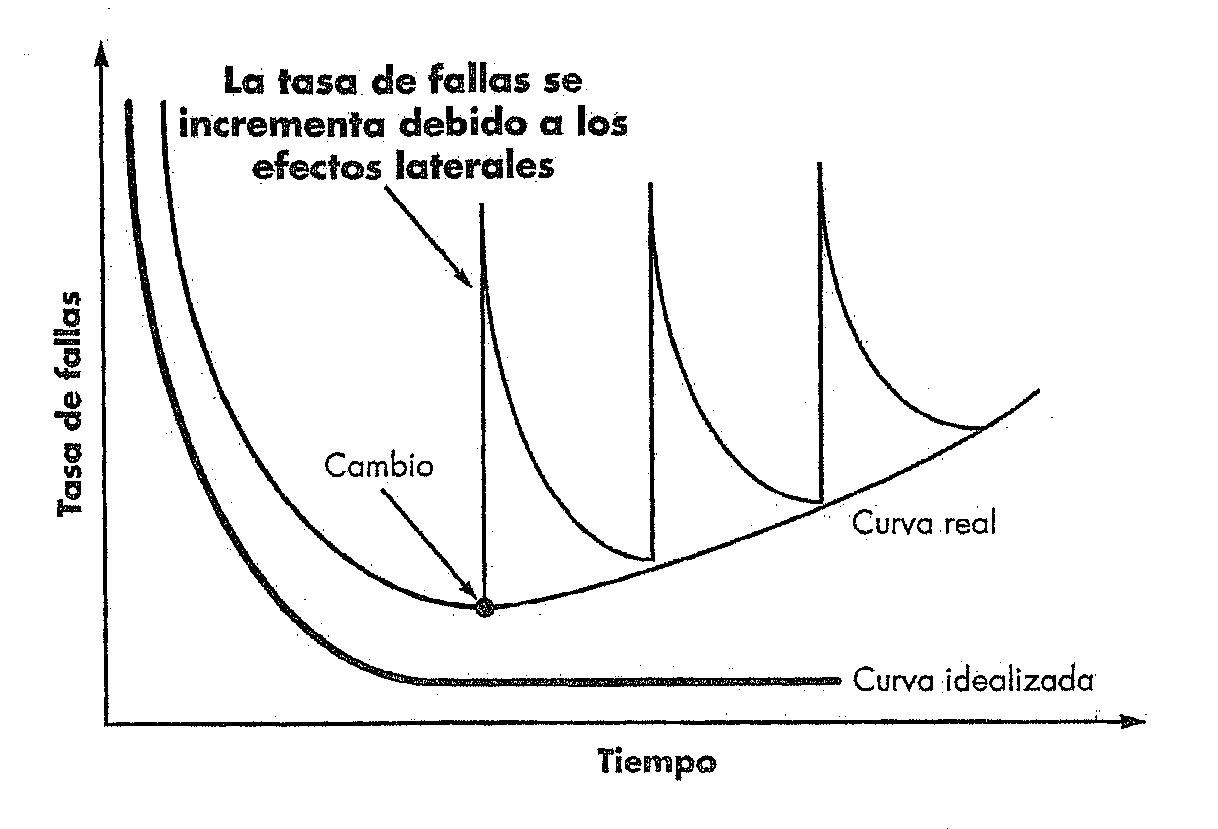


La curva de la tasa de fallas para el software debería tener la forma de Curva idealizada. Donde los defectos sin descubrir causan tasas de falla altas en las primeras etapas de vida de un programa. Los errores se corrigen y la curva se aplana. Esta curva es una simplificación burda del modelo de fallas real para el software. La implicación es clara: el software no se desgasta, pero si se deteriora.

Curva real:

El software durante su vida experimenta cambios, conforme éstos ocurren se presenta la posibilidad de tener errores, lo que ocasiona que la curva tenga un pico.

Antes de que la curva pueda regresar a su estado original con una tasa de fallas estable, se requiere otro cambio, lo que ocasiona que la curva tenga otro pico, el nivel de fallas mínimo se comienza a elevar, el software se deteriora debido a los cambios. Cualquier falla del software implica un error en el diseño.



* La mayoría aún se construye a medida. (el hardware se sustituye, el software se rediseña)

En el mundo el hardware, la reutilización de componentes ya una parte natural. En el ámbito del software, dicha actividad apenas se ha comenzado a extender.

Un componente de software se debe diseñar e implementar de forma que pueda utilizarse en muchos programas diferentes.

Los componentes reutilizables modernos encapsulan tanto los datos como el proceso que se aplica a estos, lo que permite crear aplicaciones nuevas a partir de partes reutilizables.

Ej.: Las interfaces actuales con el usuario se construyen con componentes reutilizables que permiten la creación de ventanas graficas, menús desplegables y mecanismos de interacción.

**La naturaleza Cambiante del Software**

En la actualidad existen siete grandes categorías de software:

**1- Software de sistemas:** Colección de programas escritos para servir a otros programas. Algunos programas procesan estructuras de información (compiladores, editores) otras procesan datos indeterminados (sistema operativo). Se caracteriza por la interacción muy intensa con el hardware; utilización por múltiples usuarios.

**2- Software de aplicación**: Programas independientes que resuelven la necesidad de negocios específica. Facilitan las operaciones de negocios o toma de decisiones técnicas o de gestión. También, se utiliza para controlar las funciones de negocios en tiempo real (procesamientos de transacciones de venta).

**3- Software científico y de ingeniería:** Abarca desde la astronomía hasta la vulcanología, desde la biología molecular hasta la manufactura automatizada. Se utiliza para el diseño asistido por computadora, la simulación de sistemas y otras aplicaciones interactivas, aunque éstas han comenzado a tomar características de software en tiempo real e incluso de software de sistemas.

**4- Software empotrado:** Reside dentro de la memoria de solo lectura del sistema y con él se implementan y controlan características y funciones para el usuario final y el sistema mismo.

Ej.: Puede desempeñar funciones como el control del teclado de un horno de microondas hasta funciones digitales de un automóvil, como el control de combustible.

**5- Software de línea de productos:** Diseñado para proporcionar una capacidad especifica y utilización de muchos clientes diferentes. Se puede enfocar en un nicho de mercado limitado, como productos para el control de inventarios, o dirigirse hacia mercados masivos como procesadores de texto, hojas de cálculo, manejo de bases de datos, etc.

**6- Aplicaciones basadas en Web:** En su forma más simple son apenas un poco más que un conjunto de archivos de hipertexto ligados que presentan información mediante texto y algunas gráficas.

Proporcionan características, funciones de cómputo y contenidos independientes al usuario final, así como también integración con bases de datos corporativas y aplicaciones de negocios.

**7- Software de inteligencia artificial:** Incluyen robótica, los sistemas expertos, el reconocimiento de patrones (imagen y voz), las redes neuronales artificiales, la comprobación de teoremas y los juegos de computadora.

Se espera que la generación actual facilite la tarea de los ingenieros de software del futuro, Sin embargo, han aparecido retos nuevos:

**Computación ubicua:** El reto es desarrollar software de sistema y de aplicación que permita que dispositivos pequeños, computadoras personales y sistemas de empresa se comuniquen a través de redes.

**Alimentación de la red:** El reto es crear aplicaciones que beneficien a mercados de usuarios finales específicos alrededor del mundo.

**Fuente abierta:** El reto es construir un código fuente que sea descriptivo en sí mismo, y desarrollar técnicas que permitan tanto a los clientes como a los diseñadores conocer los cambios realizados y la forma en que se manifiestan dentro del software.

**La “nueva economía”:** El reto es construir aplicaciones que faciliten la comunicación y la distribución de productos.

**Software Heredado**

Fueron desarrollados hace décadas y modificados en forma continua para cumplir los requerimientos de los cambios en los negocios y en las plataformas de cómputo.

* Se perciben como costosos en su mantenimiento y riesgosos en su evolución.
* Persisten como el soporte de las funciones centrales de negocios y son indispensables para las empresas.
* Lo caracterizan su longevidad y el ser crítico para los negocios.
* Son de la poca calidad.
* Tienen diseños imposibles de extender, código complicado, documentación escasa o inexistente, casos de prueba y resultados que nunca fueron archivados, un historial de cambio manejado con pobreza, etc.

Conforme pasa el tiempo, los sistemas heredados evolucionan por una o más de las razones siguientes:

1. El software debe adaptarse para satisfacer las necesidades de los nuevos ambientes o las nuevas tecnologías de cómputo.
2. El software debe mejorarse para implementar los nuevos requerimientos de los negocios.
3. El software debe extenderse para hacerlo operable con sistemas y bases de datos más modernos.
4. El software debe rediseñarse para hacerlo viable dentro de un ambiente de red.

Es por eso que se debe someter a una reingeniería de modo que conserve su viabilidad en el futuro.

**Evolución del Software**

El software evoluciona a través del tiempo. El cambio conduce este proceso, y se presenta:

* cuando se corrigen errores.
* cuando el software se adapta a un nuevo ambiente.
* cuando el cliente solicita características o funciones nuevas.
* cuando la aplicación experimenta una reingeniería para proporcionar beneficios en un contexto moderno.

Mitos

**Mitos de la administración**

*Ya se tiene un libro lleno de estándares y procedimientos para la construcción de software. ¿Esto proporcionara a mi gente todo el conocimiento?*

*SI se está atrasado en el itinerario es posible contratar más programadores para así terminar a tiempo.*

*Si decido subcontratar el proyecto de software a un tercero, puedo relajarme y dejar que esa compañía la construya.*

**Mitos del Cliente**

*Un enunciado general de los objetivos es suficiente para comenzar a escribir programas, los detalles se pueden afinar después.*

*Los requerimientos del proyecto cambian de manera continua, pero el cambio puede ajustarse con facilidad porque el software es flexible*

**Mitos del Desarrollador**

*Una vez que el programa ha sido escrito y puesto a funcionar, el trabajo está terminado*

*Mientras el programa no se esté ejecutando, no existe forma de evaluar su calidad*

*El único producto del trabajo que puede entregarse para tener un proyecto exitoso es el programa en funcionamiento*

*La ingeniera del software obligara a emprender la creación de una documentación voluminosa e innecesaria y de manera invariable tornara más lento el proceso.*