FaMAF

Notas de Clase

2025

Ingeniería en Software

Estudiante
Pedro Villar

August 15, 2025

1 Introducción

El **dominio del problema** de la materia son los programas que resuelven algún problema de ciertos usuarios, donde sistemas o negocios grandes dependen de dicho software, y donde los problemas en el software pueden llevar a pérdidas significativas, directas o indirectas.

Def. 1

Llamamos **software de nivel industrial** a aquel que pertenece a un negocio muy grande y en el que los problemas de software son críticos para el negocio, donde la calidad del software es fundamental. Y vamos a llamar **software** al código + todo lo del alrededor.

1.1 Alumno vs software industrial

Un **sistema de un alumno** no se utiliza para resolver ningún problema real de ninguna organización, es decir, nada de importancia o relevancia depende del correcto funcionamiento del software. Por lo tanto la presencia de *bugs* no es un problema grave. Por eso nos planteamos la pregunta de si nosotros manteminos en algun momento software que hicimos durante la cursada. Y la respuesta claramente es *no*.

- **El desarrollo** lo hace una sola persona (el alumno y usuario).
- **El error** no afecta a nadie más que al alumno.
- La **interfaz** de usuario **no es importante**.
- **No existe documentación**.

Por otro lado, un **sistema de nivel industrial** se construye para resolver un problema de un cliente y es utilizado por la organización del cliente para operar alguna parte de su negocio.

- Los **usuarios son otros**.
- **El error no es tolerable**.
- La interfaz de usuario **es importante**.
- **Se requiere documentación** para el **usuario** y para la **organización**.

Obs. 1

Actividades importantes dependen del correcto funcionamiento del sistema. Y un mal funcionamiento de dicho sistema puede tener un gran impacto en términos de pérdidas económicas o comerciales, inconvenientes para los usuarios o incluso pérdida de bienes y vidas. En consecuencia, el sistema de software necesita ser de alta calidad en cuanto a propiedades como confiabilidad, usabilidad, portabilidad, etc.

Alta calidad requiere mucho **testing**, además se debe descomponer el desarrollo en etapas para poder detectar distintos *bugs* en distintas etapas.

1.2 Costo del software

El software de fuerza industrial es muy costoso, principalmente debido a que el desarrollo de software es una actividad extremadamente intensiva en **mano de obra**.

Obs. 2

Las líneas de código (LOC) o miles de líneas de código (KLOC) entregadas son, la medida más comúnmente utilizada para determinar el tamaño del software en la industria.

Como el principal costo de producir software es la mano de obra empleada, el costo de desarrollo suele medirse en términos de persona-mes de esfuerzo invertido en el desarrollo. Y la productividad en la industria se mide con frecuencia en términos de LOC (o KLOC) por persona-mes.

1.3 Demorado y poco confiable

Las fallas de software son distintas de las fallas mecánicas o eléctricas. En software, las fallas **no** son consecuencia del uso y el deterioro. La falla que causa el problema **existe desde el comienzo**, sólo que se manifiesta tarde.

1.4 Mantenimiento

Una vez entregado, el software *requiere mantenimiento*. El software necesita mantenimiento no porque alguno de sus componentes se desgaste y deba reemplazarse, sino porque a menudo quedan errores residuales en el sistema que deben eliminarse cuando se descubren.

Aunque el mantenimiento no se considera parte del desarrollo de software, es una actividad extremadamente importante en la vida de un producto software. Si consideramos toda su vida útil, el costo del mantenimiento generalmente supera al costo de desarrollo.

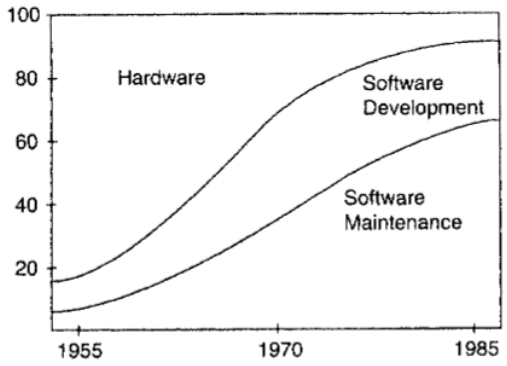

Figure 1: Costo de mantenimiento vs costo de desarrollo.

Def. 2

El mantenimiento **correctivo** se refiere a la eliminación de errores residuales o *bugs* que se descubren una vez que el software ha sido entregado y desplegado. Implica la eliminación de defectos que causan que el software se comporte de manera inconsistente con sus requisitos o necesidades del cliente.

Def. 3

El mantenimiento **adaptativo** se realiza para mejorar funcionalmente el software (conocidos como "upgrades") o para adaptarlo a los cambios en su entorno. También abarca la adición de nuevas funcionalidades o características que no estaban presentes en la versión original, a menudo impulsadas por la evolución de las necesidades del negocio o la aparición de nuevas tecnologías.

2 Enfoque de la Ingeniería en Software

En gran medida, el proceso de software determina la calidad del producto y la productividad alcanzada. Por lo tanto, para abordar el dominio del problema y enfrentar con éxito los desafíos de la ingeniería de software, es necesario centrarse en el proceso de software. El diseño de procesos adecuados y su control se convierten así en un objetivo clave de la investigación en ingeniería de software.

2.1 El proceso de desarrollo en fases

- Cada fase termina con una **salida definida**.
- Las fases se realizan en el **orden especificado** por el modelo de proceso que se elija seguir.
- El motivo de separar en fases es la **separación de incumbencias**: cada fase manipula distintos aspectos del desarrollo de software.
- El proceso en fases permite **verificar la calidad y progreso** en momentos definidos del desarrollo, al final de la fase.

En general, las fases son:

- **Análisis y especificación de requerimientos**.
- **Arquitectura**.
- **Diseño**.
- **Codificación**
- **Testing**.
- **Entrega e instalación**.

3 Análisis y especificación de requerimientos

2 Clase del JUEVES 14 DE AGOSTO - **Introducción**