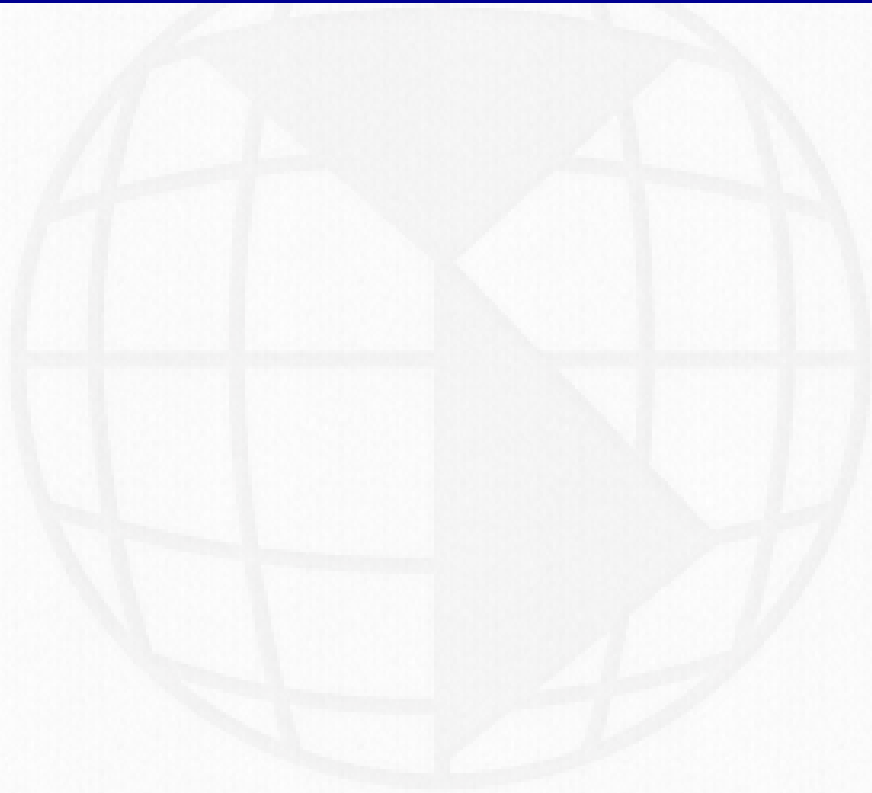




UNID®

U N I V E R S I D A D
I N T E R A M E R I C A N A
P A R A E L D E S A R R O L L O

INGENIERÍA DEL SOFTWARE



Contextualización

Introducción al Tema

¿Qué es la construcción de software?

La construcción de software es la etapa en la cual se crea el producto en un lenguaje de programación partiendo del diseño definido, una vez concluido se válida para posibles correcciones.

Es importante considerar realizar en la etapa de análisis de requerimientos un estudio de factibilidad, enfocándolo sólo en el aspecto de la construcción, esto sirve para detectar restricciones técnicas de uso para el desarrollo del software.

Algunos modelos de software recomiendan realizar pruebas entre la construcción del software, por ejemplo, si el sistema es modular, realizar pruebas por cada módulo; esto con el objetivo de identificar en etapas tempranas posibles fallas en el sistema.



Explicación

Construcción y pruebas del software

Las pruebas de software tiene como objetivo validar que su comportamiento sea el adecuado cumpliendo con los requisitos ya establecidos, en algunos casos se sugiera que al software construido a la medida se le realizase una prueba al menos por requerimiento para evitar complicaciones; en caso de que los resultados de la prueba sean negativos, el desarrollador debe regresar a verificar o en algunos casos, rehacer la funcionalidad.

¿Cuándo debemos realizar las pruebas?

En realidad no existe una regla estricta que indique cuando debemos de realizar las pruebas de software, como mencionamos anteriormente, algunos modelos sugieran que las pruebas se realicen cada vez que se termina un módulo, algunos otros sugieren que se realicen pruebas generales hasta que una versión del sistema esté completa, aunque podemos decir que cada vez que ejecutamos un módulo podemos decir que surge una nueva prueba, ya que hay veces que hasta ejecutar el modulo en producción puede que se presenten problemas. Para estos casos podemos utilizar un modelo estadístico de fiabilidad de software, del cual hablaremos más adelante.

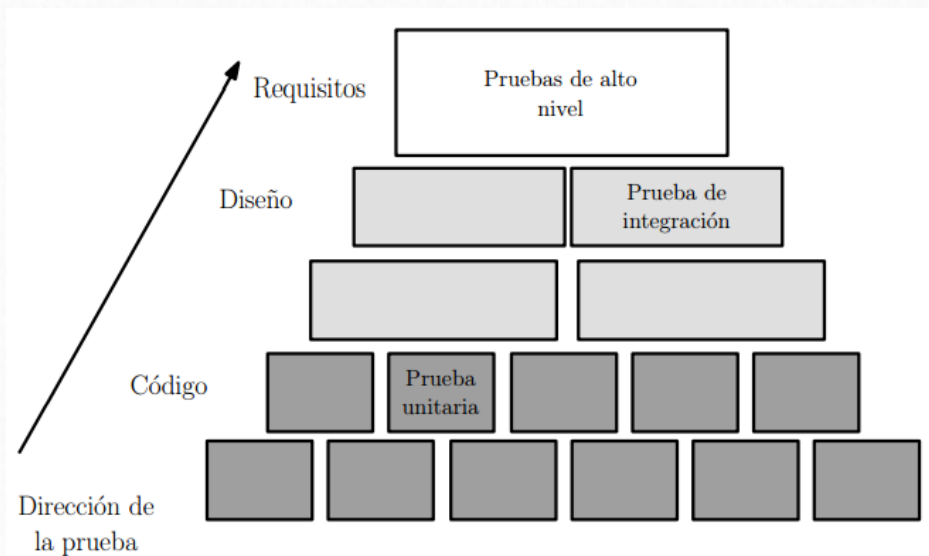


Figura 1. Pressman, R. (2002). Ingeniería de software un enfoque práctico [Etapas en la prueba de software.] (p.308). Madrid: McGraw-Hill

Principios de construcción

Al inicio de la construcción del software debemos de realizar una **planificación**, la cual define en qué orden serán creados los componentes e integrados, los procesos de gestión de calidad, distribución de recursos, y otras tareas.

Los principios de construcción de un software se determinan para cumplir con los lineamientos necesarios en la calidad, los cuales se estipulan en la IEEE, que estable un parámetro útil en la creación de aplicaciones.

Los principios de construcción pueden ser utilizados para crear las versiones finales de los elementos o también para crear los prototipos del software, pues las bases son las mismas. Algunos de los principios son:

- Implementación de código a cada clase a partir del modelo de diseño
- Estructura del programa para el acceso del sistema
- Estructura de programa para menú de la aplicación
- Estructura de programa para mantenimiento de una clase tipo entidad
- Integración de clases extraídas de biblioteca de reutilización
- Construcción del prototipo o incremento

Codificación

La codificación es el procedimiento en el que se comienzan a crear las estructuras de programación dentro de un documento, el cual será la aplicación para los dispositivos electrónicos. Este procedimiento se realiza con cualquier lenguaje de programación, pues la codificación es uno de los pasos finales que se realizan dentro de las fases para la creación de aplicación en la ingeniería de software.

Los lenguajes de programación también sirven para decodificar algunos elementos en específico, pero esto depende de que los complementos de lectura y traducción los contenga la computadora, pues de lo contrario, lo que se lograría sería solo tener algunos caracteres y algunas figuras que no son comprensibles, además de la visualización de cuadrados en lugar de letras y de figuras. Normalmente los lenguajes de codificación se basan los prefijos del inglés, pues se desarrollan en países en los que es la lengua principal de los programadores.



En otros elementos los aspectos de codificación se dan en los medios de video y audio, es decir, utiliza un código propio con el que se comunica la máquina con los archivos y los traduce de tal forma en que se muestren de una forma comprensible para los usuarios.

En la etapa de codificación, se sugieren utilizar técnicas que generalmente no afectan a la funcionabilidad del proyecto, pero generan una mejor comprensión del mismo.

Reutilización e integración de código

La reutilización de código e integración se da para disminuir los tiempos de producción y mejorar en base a un sistema funcional y bueno, pues actualmente debido a la complejidad de los sistemas es más tardado poder crear una nueva herramienta con los lapsos tan cortos de tiempo que se dan para cubrir las necesidades de los consumidores.

La reutilización de los códigos se puede dar para lanzar nuevas versiones de un sistema o aplicación, se retoman las bases funcionales y se optimizan, ya sea reduciendo el código o extendiéndolo y agregando herramientas de funciones, esto no determina si la aplicación es de complejidad o sencilla.



Fallos, errores y defectos

Los fallos, errores y defectos, son muy comunes en la mayoría de las aplicaciones, pues se presentan cuando algo no está bien ya sea en el software que se aplica o en el sistema operativo que lo soporta.

Muchos de estos errores se generan por la falta de memoria, por la mala gestión de la información, por la incompatibilidad de los scripts de ejecución de la aplicación y el sistema operativo, entre muchas otras razones.

Los virus informáticos son algunos de los mayores creadores de problemas en un sistema, pues debido a su naturaleza destructiva los principales códigos y los más importantes que controlan la mayoría de los procesos se ven afectados y presentan una mala condición del sistema, ocasionando que se deba restaurar por completo perdiendo toda la información o dejando que se agote la memoria disponible, ocasionando volcado de información y cierres inesperados de las aplicaciones.

Modelo de fiabilidad

Para evitar que el software funcione de manera incorrecta podemos realizar un modelo de fiabilidad, en el cual se verifican las estructuras del software, y su modelo de desarrollo.

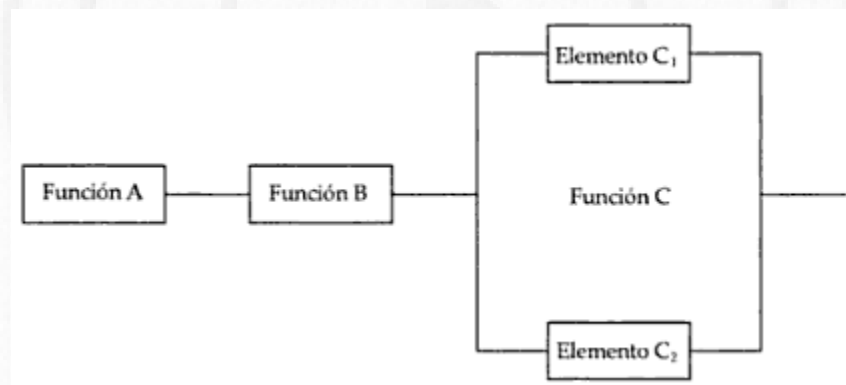


Figura 2.Creus, A. (2005).Fiabilidad y seguridad. Su aplicación en procesos Industriales [Modelo convencional de fiabilidad.] (p.187). España: Marcombo

¿Cuál es la diferencia entre defecto, error y falla?

Un error puede ser aquel que lleva a más errores. Una falla es un síntoma ocasionado por un defecto. Un defecto puede comprenderse como la diferencia entre la versión correcta y la incorrecta de un artefacto.

Niveles y tipos de pruebas

Los niveles y pruebas que se realizan a un software son importantes, pues con éstas se determina que tantos errores contiene o los elementos que se pueden mejorar y optimizar, además de cubrir algunas de las características que los usuarios puedan desear en el uso o interfaz de la misma. Las pruebas más comunes son;

Prueba unitaria: esta prueba se realiza para determinar el correcto funcionamiento de un modelo de código, esto sirve para asegurar que cada módulo de los que componen el sistema funcionan correctamente.

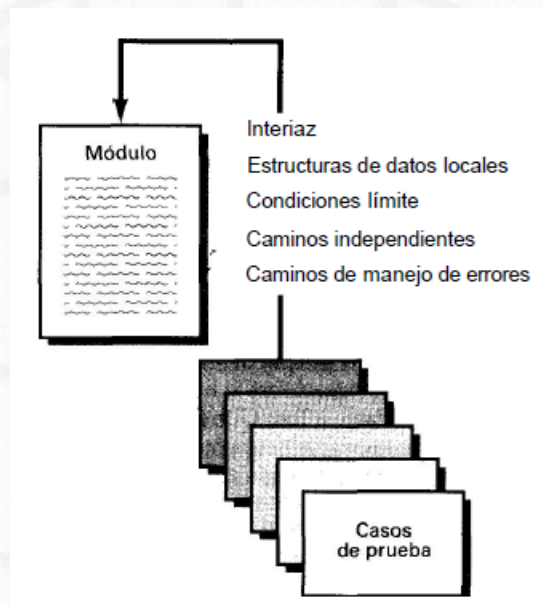


Figura 3. Pressman, R. (2002). Ingeniería de software un enfoque práctico [Prueba de unidad.] (p.310). Madrid: McGraw-Hill

Prueba funcional: esta prueba se basa en la ejecución, revisión y retroalimentación de las funcionalidades diseñadas para el software con anterioridad.

Prueba de integración: éstas se realizan en el ámbito de desarrollo una vez que se han aprobado las pruebas unitarias. Se refiere a la prueba o pruebas de los elementos unitarios que componen el proceso.

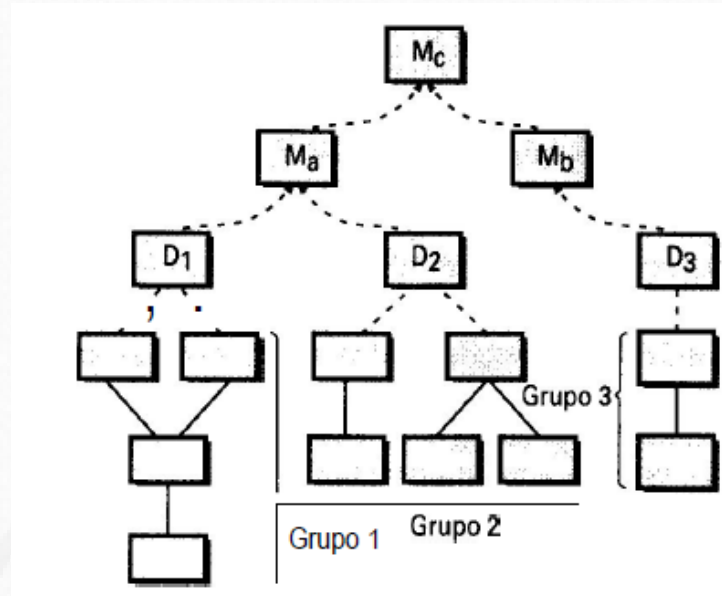


Figura 4. Pressman, R. (2002). Ingeniería de software un enfoque práctico [Integración ascendente.] (p.314). Madrid: McGraw-Hill

Pruebas de validación: ésta se realiza con el fin de verificar el cumplimiento de lo que se estipula sobre el software, si cumple con los fines para los cuales se ha programado, se valida.

Caja blanca: éste es un tipo de prueba que se realiza sobre las funciones internas de un módulo.

Caja negra: se denomina caja negra a aquel elemento estudiado desde el punto de vista de las entradas que se reciben y las salidas o respuestas que se producen.

Caja negra y programación modular: éste es el apartado en el que la programación se divide en módulos dentro del sistema global que es el programa que pretende desarrollar.

Por otro lado existen las pruebas en el ciclo de vida del software, las cuales son:

- Pruebas de aceptación
- Pruebas de integración de sistema
- Pruebas de sistema
- Pruebas de integración de componentes
- Prueba de componente o unitarias

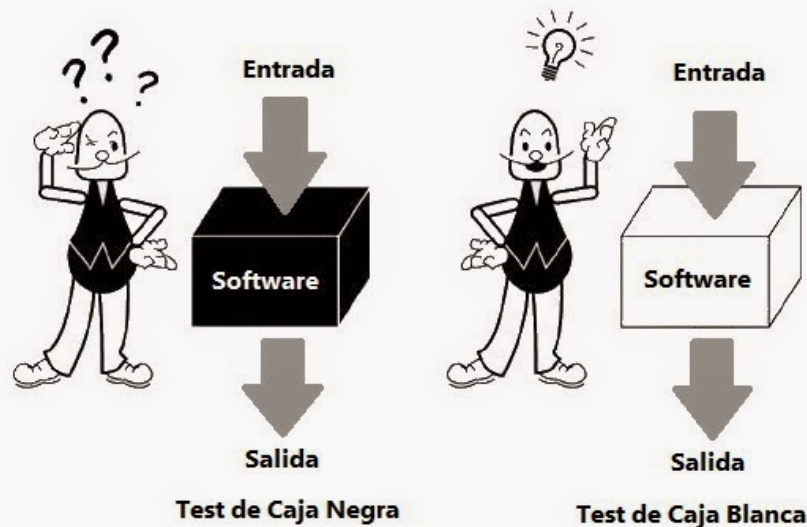


Figura 5. Eleven Paths. (2014). Pruebas para asegurar la calidad del producto software (III) [caja blanca y caja negra.]. Recuperado de: <http://blog.elevenpaths.com/2014/12/qa-pruebas-para-asegurar-la-calidad-del.html>

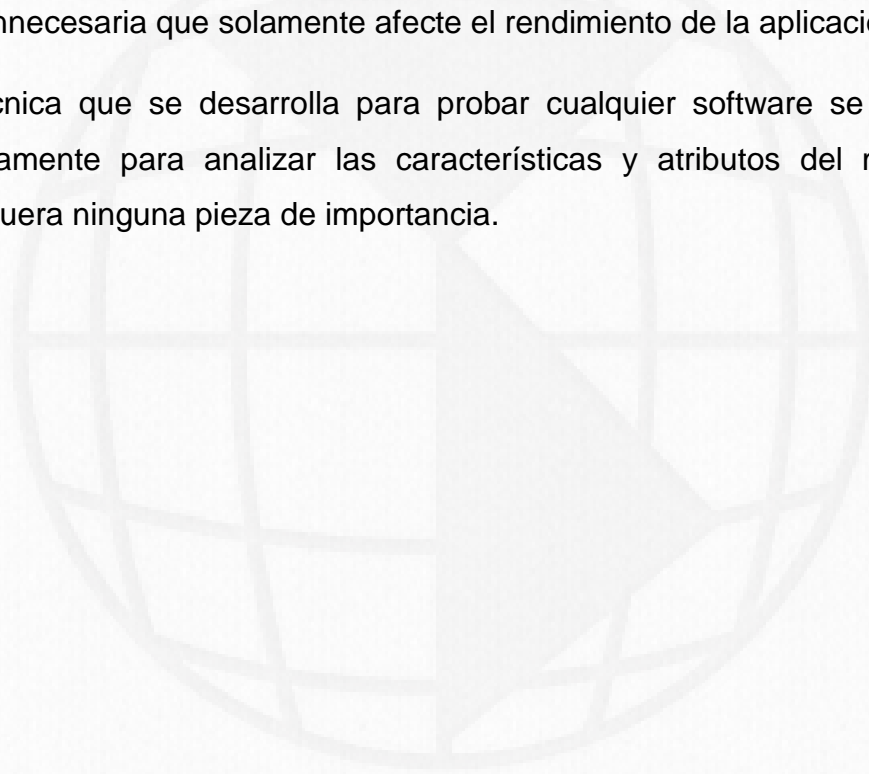
Técnicas de prueba

Las técnicas de prueba de un sistema de software implican varios pasos, dentro de los cuales está la realización de una serie de actividades predispuestas a incorporar errores. Estas técnicas se desarrollan para poder liberar cualquier software al mercado, es la forma en la que se verifica la calidad y que se cumpla con lo que se ofrece.

¿Cuáles son los objetivos dentro de las pruebas?

Los principales objetivos de las pruebas son el lograr determinar errores y el sitio en el que se forman, para poder corregir las líneas de código o eliminar la sintaxis innecesaria que solamente afecte el rendimiento de la aplicación.

Cada técnica que se desarrolla para probar cualquier software se desarrolla específicamente para analizar las características y atributos del mismo, no dejando fuera ninguna pieza de importancia.



Conclusión

Los códigos que se utilizan en la construcción de un software pueden ser diferentes y tener características y atributos que no se consideran por completo, por lo que muchas veces se reutilizan líneas de código y se integran, pues su orden y funcionamiento en otros sistemas garantizan un buen uso de sintaxis y del uso de palabras reservadas.

La codificación es importante, pues se puede hacer en forma escrita en papel para verificar la sintaxis y tener una buena forma de ordenar los elementos principales de la aplicación. Las pruebas que se realizan al software son la forma de verificar mediante los usuarios si este cumple con lo que se ha estipulado en las fases anteriores y se pueden corregir para ofrecer una estabilidad mayor y optimizar los procedimientos sin consumir una gran cantidad de recursos en el sistema que lo alberga.



Para aprender más

Fallas por errores de software.

- (2013). *Fallas en la BMV, por errores de software: Téllez*. Julio 25, 2013, de El Economista. Sitio Web: <http://eleconomista.com.mx/mercados-estadisticas/2013/07/25/fallas-bmv-errores-software-tellez>

-

Software ilegal.

- (2010). *Informe sobre los riesgos y costos del uso de software ilegal*. Diciembre 2, 2010, de Seguridad Informática. Sitio Web: <https://seguinfo.wordpress.com/2010/12/02/informe-sobre-los-riesgos-y-costos-del-uso-de-software-ilegal/>

Actividad de Aprendizaje

Instrucciones:

Con la finalidad de aplicar tus conocimientos adquiridos a lo largo del curso, tendrás que realizar una actividad la cual consiste en realizar como primera actividad una [lista de al menos 6 requerimientos para brindar solución al siguiente problema](#):

PROBLEMA

El cliente es una escuela que cuenta con tres niveles educativos (primaria, secundaria, bachillerato), tiene un promedio de población de 1200 alumnos y 40 profesores. El director escolar solicita un sistema para la biblioteca, que opere de la siguiente manera:

- Identifique al usuario con el perfil adecuado.
- Al realizar las búsquedas, la información se limite dependiendo de su nivel escolar, materia y edad.
- Cuando se realice una búsqueda se pueda mostrar una sugerencia acerca de este tema, por ejemplo, al buscar historia de México el sistema muestre una sugerencia de búsqueda como “Quizá te pueda interesar”, y esta información también debe de estar limitada.
- Tener un perfil “Administrador” para poder ver las estadísticas de búsqueda de los usuarios y ver búsquedas por usuario individualmente.
- Controlar el sistema para que los usuarios no ingresen con la sesión que no les corresponde (no guardar contraseñas).

Este problema le daremos seguimiento a lo largo del curso en diferentes etapas, por lo tanto es importante que realices el desarrollo principal adecuado.

Para esta actividad se tomará en cuenta lo siguiente:

- Título
- Datos personales
- Ortografía y redacción
- Actividad solucionada
- Bibliografía

Bibliografía

- UTEC, *Construcción del software*. De Universidad Tecnológica del Salvador. Sitio Web: <http://biblioteca.utec.edu.sv/siab/virtual/auprides/16039/capitulo%205.pdf>
- Vasconcelos, J. (2000) *Manual de construcción de programas*. Sitio Web: http://www.cyta.com.ar/biblioteca/bddoc/bdlibros/construccion_programas/programar.pdf
- Pressman, R. (2002). *Ingeniería de software .Un enfoque práctico*. Madrid: McGraw-Hill.
- Tuya, J., Ramos, I., & Dolado, J. (2007). *Técnicas cuantitativas para la gestión en la ingeniería de software*. España: Gesbiblo.
- Creus, A. (2005). *Fiabilidad y seguridad. Su aplicación en procesos industriales*. España: Marcombo.