



Introducción

- Usualmente las tareas realizadas como parte del desarrollo de un software son modeladas durante el Ciclo de Vida de Desarrollo del Software. (*Software Development Lifecycle*)
- El ciclo comienza al identificar los requisitos y termina con la verificación de dicho software para asegurarse de cumpla con los mismos.
- El ciclo de desarrollo del Software no existe por si solo. El mismo es parte de ciclo mas grande, el ciclo de vida de un producto.

Introducción

- Dentro de la vida de un producto ocurren varias etapas como lo es el mantenimiento.
- En un mantenimiento se pueden corregir defectos del producto y/o adaptarse a nuevos requisitos.
- Muchas veces un producto puede ser tan sencillo como un componente de software aislado, pero puede llegar a incluir múltiples desarrollos de componentes de software.

Introducción

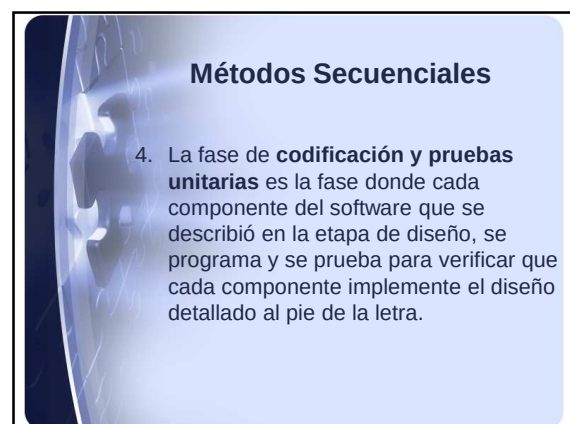
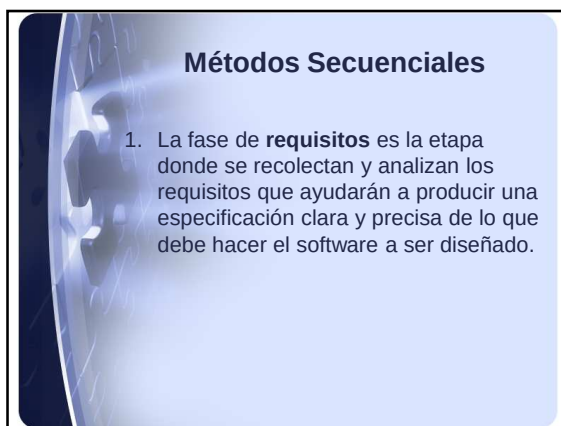
- Existen varios modelos que se pueden utilizar como guía en el ciclo de desarrollo de un software.
- Un detalle en común que tienen **TODO**s los modelos es que en algún punto se tiene que probar el software.
- Como parte de esta presentación vamos a enseñarle algunos de los modelos de desarrollo utilizados con énfasis en las pruebas.

Métodos Secuenciales

- El ciclo de desarrollo comienza con la identificación y recolección de los requisitos del software, y termina en la verificación formal de que el software diseñado cumple con estos requisitos.
- Tradicionalmente los modelos utilizados para el desarrollo de software han sido secuenciales. Varios modelos que siguen este patrón son el modelo V (*V lifecycle model*) y el modelo en cascada (*waterfall lifecycle model*).

Métodos Secuenciales

- Existen muchas variaciones de los modelos V y de cascada donde se añaden diferentes etapas y cambian las fronteras de cada etapa. Sin embargo, las siguientes etapas encajan en la vida diaria de muchos desarrolladores:



Pruebas Unitarias

- Las pruebas unitarias son las pruebas donde el **programador** valida y prueba por medio de pruebas individuales de cada componente, si el código puede ser utilizado.
- Existen diversas herramientas para ayudar al programador a realizar las pruebas. En el caso de Java existen un conjunto de librerías llamado JUnit.

Métodos Secuenciales

5. La fase de **integración del software** es la fase donde se integran los componentes previamente probados para ver como éstos interactúan para formar un componente de software completo.

Métodos Secuenciales

6. La fase de **integración del sistema** es una donde el componente de software diseñado se integra al producto completo y se prueba.

Métodos Secuenciales

7. Por último la fase de **Pruebas de Aceptación** (*User Acceptance Test*) es donde las pruebas son realizadas en conjunto con el cliente para asegurarse que el software cumple con los requisitos especificados.

Métodos Secuenciales

- Las especificaciones serán productos de las primeras tres fases del modelo. Las otras cuatro fases que envuelven diferentes pruebas de distintos niveles y requieren que se especifique como se van a realizar las pruebas como prerrequisito.

Métodos Progresivos

- Los modelos antes descritos representan modelos idealizados del desarrollo de software. Sin embargo, al pasar el tiempo se comenzaron a desarrollar otros modelos para adaptarse a diferentes necesidades.
- Por ejemplo, existen soluciones que son diseñadas para negocios cambiantes y muchas veces los requisitos son volátiles.
- Otras veces simplemente existen exigencias de tiempo más agresivas donde se requiere crear un sistema interino debido a que tomaría mucho tiempo tener el sistema completo funcionando.

Métodos Progresivos

- Un problema muy común en el desarrollo de software es que el mismo necesita ser utilizado rápidamente pero tomaría demasiado tiempo para desarrollarlo completamente.
- La solución a este dilema es llegar a un punto medio donde el itinerario de entrega y la funcionalidad puedan proveer entregables intermedios con funcionalidad reducida pero que sirvan como cimientos para construir el resto de la funcionalidad del software.

Métodos Progresivos

- Muchas veces se denominan ciertas metas ("milestones") o fases para reducir el riesgo de diseñar un software completamente y costear toda la fase de desarrollo.
- Los nombres usuales que se le dan estas metodologías de desarrollo de software son el método de **desarrollo progresivo** o **implementación por fase**.

Métodos Progresivos

- En un ciclo de desarrollo progresivo cada fase individual de desarrollo sigue su propio ciclo de desarrollo.
- Usualmente el número de fases depende del producto o solución que se esta desarrollando.

Métodos Progresivos



Métodos Progresivos

- Cada entregable de software debe pasar una prueba de aceptación para verificar que cumple con todas las partes relevantes del total de requisitos.
- Las pruebas y la integración de cada fase requiere tiempo y esfuerzo así que llega un punto donde añadir más fases de desarrollo se convierte en contra productivo.
- Podría incrementar el costo y el tiempo de entrega que son muy importantes a la hora de poder producir un software funcional para una implementación temprana.

Métodos Progresivos

- El software que se produce en una fase temprana puede que nunca sea utilizado. Sin embargo, puede servir como un prototipo.
- En un prototipo usualmente se utilizan atajos para proveer una manera de validar los requisitos claves y que el cliente apruebe las partes críticas del diseño.
- Dichos atajos pueden ser en áreas como documentación y posiblemente pruebas.

Métodos Progresivos

- Cuando se toman atajos es esencial descartar el prototipo e implementar la próxima fase desde el principio.
- Esto se debe a que debido a la pobre calidad que usualmente tiene un prototipo, puede que los cimientos del mismo no sirvan para un desarrollo continuo.

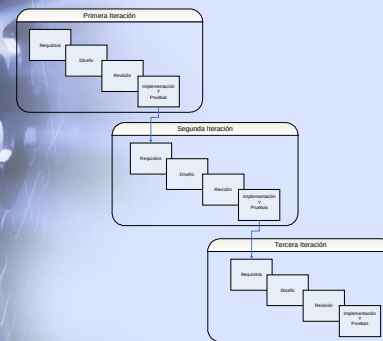
Métodos de Desarrollo Iterativos

- Los modelos iterativos de desarrollo de software no comienzan usualmente con una especificación de requerimientos completa.
- En su lugar, el desarrollo comienza con la implementación de solamente una parte del software.

Métodos de Desarrollo Iterativos

- Esta implementación luego será utilizada para identificar requisitos futuros.
- Este proceso es repetido hasta producir una nueva versión del software para cada ciclo del modelo.

Métodos de Desarrollo Iterativos



Métodos de Desarrollo Iterativos

1. La fase de requerimientos es la fase donde se recogen los requerimientos del software y se analizan. Cada iteración va añadiendo a los requisitos hasta poder tener especificación del software final.

Métodos de Desarrollo Iterativos

2. La fase de diseño es la fase en la cual se utilizan los requisitos para realizar un diseño. Este nuevo diseño puede ser nuevo o una extensión del diseño realizado en iteración anterior.

Métodos de Desarrollo Iterativos

3. La fase de **implementación y pruebas** es cuando el software es programado e integrado para ser probado.

Métodos de Desarrollo Iterativos

4. Una fase de **revisión** en la cual el software es evaluado y los requisitos implementados en esa iteración son aprobados y/o cambios y nuevos requisitos son propuestos.

Métodos de Desarrollo Iterativos

- Para cada ciclo del modelo una decisión se debe tomar sobre si el software que se produjo de este ciclo se va a descartar, o si se va a tomar como base para el próximo ciclo. Esto es conocido en inglés como *incremental prototyping*.
- Eventualmente se llega a un punto donde todos los requisitos son completados y el software puede ser entregado, o resulta imposible mejorar el software como es requerido y se comienza desde el principio.

Métodos de Desarrollo Iterativos

- El método iterativo puede ser comparado con hacer software por aproximación utilizando la analogía de los métodos matemáticos.
- El éxito dependerá de cuán rápido convergen a una solución.

Métodos de Desarrollo Iterativos

- Uno de los problemas primordiales de este método es que las iteraciones nos pueden llevar a una solución que no sea práctica, o que el número de iteraciones incremente sin control y se extienda demasiado el proyecto.
- La clave para el uso exitoso del método iterativo es la validación continua de los requisitos y la verificación, que incluye las pruebas, de cada versión contra los requisitos de cada modelo.

Métodos de Desarrollo Iterativos

- Las primeras tres fases del modelo iterativo son una forma abreviada de los modelos tradicionales V y de cascada.
- Cada ciclo del modelo produce un software que requiere que se le corran pruebas unitarias, pruebas de integración, pruebas de integración de sistemas y pruebas de aceptación.
- Mientras el software evoluciona a través de los ciclos, las pruebas se deben repetir y verificar cada versión del software.

Mantenimiento

- Cuando un software es completado exitosamente, usualmente forma a pasar parte de un producto y entra en una fase de mantenimiento.
- Durante la fase de mantenimiento, se realizan modificaciones para corregir errores y para adaptarse a nuevos requisitos.
- Como en el desarrollo inicial, las modificaciones seguirán un ciclo de desarrollo pero no necesariamente utilizando el mismo modelo que se utilizó en el desarrollo original.

Mantenimiento

- A través de la fase de mantenimiento las pruebas del software deben ser repetidas, modificadas y extendidas.
- El esfuerzo de revisar y repetir las pruebas subsecuentemente forma parte de los costos de desarrollar y mantener el software.

Mantenimiento

- El término **pruebas de regresión** es utilizada para referirse a la repetición de pruebas exitosas previamente para asegurarse de que el software no ha sufrido nuevos cambios inesperados.

Resumen y Conclusión

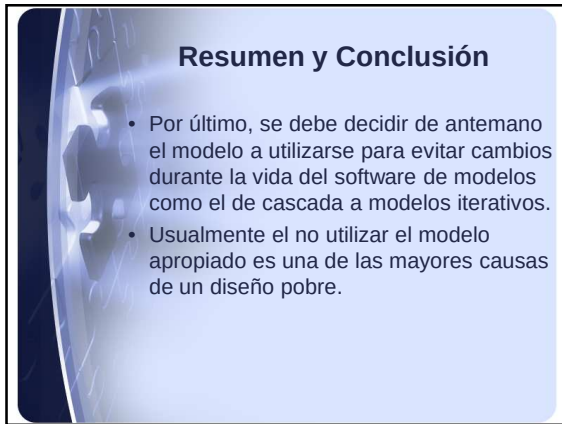
- Independientemente del ciclo o modelo utilizado para el desarrollo de software, el mismo debe pasar por una serie de pruebas.
- El software debe ser probado para asegurarnos de la eficiencia y calidad del mismo durante el proceso original y luego de cada cambio subsecuente realizado al mismo.

Resumen y Conclusión

- La práctica de realizar pruebas se convierte en mas crítica al utilizar modelos como lo son el progresivo y el iterativo.
- Esto se debe a que ese tipo de desarrollo necesita mayor control de calidad que con los métodos tradicionales.

Resumen y Conclusión

- Las **pruebas de regresión** son muy importantes para el mantenimiento de un software.
- Es muy fácil realizar cambios sin darnos cuentas de los efectos secundarios que puedan ocurrir en el mismo.
- La facilidad con que se puedan automatizar dichas pruebas influye grandemente en el costo de mantenimiento.



Resumen y Conclusión

- Por último, se debe decidir de antemano el modelo a utilizarse para evitar cambios durante la vida del software de modelos como el de cascada a modelos iterativos.
- Usualmente el no utilizar el modelo apropiado es una de las mayores causas de un diseño pobre.