Metodología para desarrollo de sistemas orientados a objetos

¿Por qué analizar y diseñar?

"En resumidas cuentas, la cuestión fundamental del desarrollo del software es la escritura del código. Después de todo, los diagramas son solo imágenes bonitas. Ningún usuario va a agradecer la belleza de los dibujos; lo que el usuario quiere es software que funcione."

Fowler, M & Scott, K.(1999)

La Programación Orientada a Objetos requiere de una metodología de trabajo para poder abordar el diseño y desarrollo de sistemas de manera efectiva y organizada. A continuación se presenta una metodología genérica, sin entrar en detalle en cada una de sus etapas:

Análisis del sistema: En esta etapa se estudia el sistema a desarrollar y se definen los objetivos del mismo. Se deben identificar los requerimientos, restricciones y limitaciones del sistema, y se debe analizar la realidad correspondiente para construir un modelo de análisis que represente de manera efectiva el problema.

Diseño del sistema: Una vez que se tiene un modelo de análisis claro y completo, se procede a diseñar el sistema. En esta etapa se definen las soluciones para cada uno de los requerimientos del sistema, y se establece cómo se van a implementar estas soluciones. Se debe generar una documentación clara y precisa de las clases y métodos que se van a utilizar, utilizando notaciones como UML.

Implementación del sistema: Finalmente, se procede a la implementación del sistema en un lenguaje de programación orientado a objetos. En esta etapa se traducen las soluciones y documentación del diseño a código concreto, generando las clases y métodos necesarios para llevar a cabo las tareas del sistema.

Es importante destacar que estas etapas no son necesariamente lineales ni exclusivas, y en muchos casos se realizan en paralelo o se realizan varias iteraciones en cada una de ellas.

Existen diversas metodologías de desarrollo de sistemas orientados a objetos, cada una con sus propias características y ventajas. Algunas de ellas son:

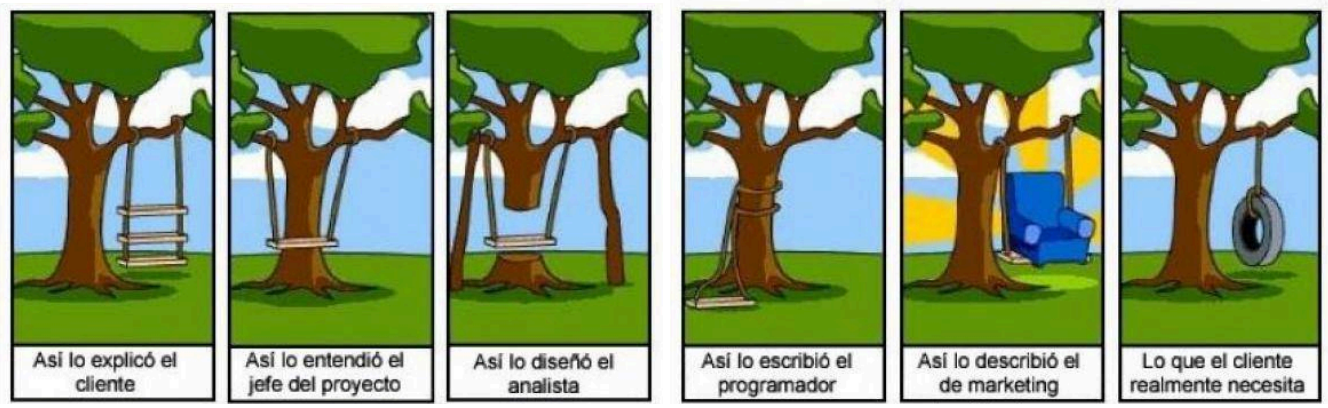
Métodos ágiles: Son metodologías que priorizan la flexibilidad y la adaptabilidad, y buscan generar entregables funcionales en plazos cortos. Algunos ejemplos son Scrum y XP.

Proceso unificado: Esta metodología se basa en una serie de fases iterativas, donde se van construyendo incrementos del sistema hasta alcanzar la versión final. Se utiliza una documentación detallada y se presta especial atención a la calidad del software generado.

Rational Unified Process (RUP): Es una metodología iterativa e incremental que se basa en el uso de casos de uso para modelar los requerimientos del sistema. Se presta especial atención a la calidad del software generado y a la gestión del proyecto.

En este curso utilizaremos una metodología basada en las etapas de requerimientos, análisis, diseño e implementación, que se basa en el proceso iterativo e incremental. En la etapa de requerimientos se definen los objetivos y requerimientos del sistema, en la etapa de análisis se estudia el problema y se construye un modelo de análisis, en la etapa de diseño se definen las soluciones y se genera la documentación necesaria, y finalmente en la etapa de implementación se traduce la documentación a código concreto. Esta metodología permite un desarrollo organizado y estructurado de sistemas orientados a objetos, garantizando una alta calidad del software generado.

Modelo de proceso iterativo e incremental como marco de aplicación.



Puedes ver diversas imágenes originales de *Tree swing cartoon* en el siguiente link. Se recomienda observar la de Tom Gilb (1988).

El modelo de proceso iterativo e incremental es un marco de aplicación para la metodología de desarrollo de sistemas orientados a objetos que se basa en la idea de desarrollar un sistema de manera iterativa, dividiendo el proyecto en pequeñas partes que se construyen e implementan en ciclos cortos y repetitivos. Cada iteración construye un componente funcional del sistema que es probado y validado antes de pasar a la siguiente iteración. El modelo iterativo e incremental es un enfoque popular en la programación orientada a objetos debido a su flexibilidad y capacidad para adaptarse a cambios en los requisitos del sistema.

Otro modelo de desarrollo de sistemas orientados a objetos que puede ser útil de mencionar es el Modelo en Cascada o Waterfall. Este modelo es un enfoque secuencial y lineal en el que cada etapa del proceso de desarrollo debe ser completada antes de pasar a la siguiente. Este enfoque no es tan flexible como el modelo iterativo e incremental y puede ser difícil de adaptar a cambios en los requisitos del sistema. Sin embargo, el modelo en cascada puede ser útil en proyectos donde los requisitos son claros y estables desde el principio.

Otro modelo que podría ser mencionado es el modelo en espiral. Este modelo combina elementos del modelo iterativo e incremental y del modelo en cascada, y se enfoca en la gestión de riesgos. El modelo en espiral es un enfoque más complejo y puede ser más adecuado para proyectos grandes y complejos.

Cabe mencionar que la profundidad en cada uno de estos modelos de desarrollo de sistemas orientados a objetos depende del enfoque y las necesidades específicas del proyecto. En la asignatura de Ingeniería de Sistemas se profundiza en estas metodologías y se estudian herramientas y técnicas adicionales para llevar a cabo proyectos de desarrollo de sistemas.

Actividades

1. En una plataforma de educación en línea se desea crear un programa que permita al estudiante repasar conceptos a través de preguntas de opción múltiple. El programa deberá permitir al usuario crear preguntas de opción múltiple y almacenarlas en una base de datos. Posteriormente, el programa debe permitir al estudiante seleccionar una cantidad determinada de preguntas al azar para responder y visualizar el resultado de la evaluación al final de la sesión. Además, el programa deberá permitir la actualización y eliminación de preguntas existentes en la base de datos.

   Para lograr esto se solicita lo siguiente:

   a) Identificar a los usuarios del sistema y resumir en dos o tres líneas cuál será el dominio del mismo.

   b) Elaborar un cuestionario que permita obtener información detallada y concreta acerca de los requerimientos del programa. Clasificar las preguntas en:

      Preguntas para el usuario que creará las preguntas.

      Preguntas para el estudiante que responderá las preguntas al azar.

   c) Identificar quiénes podrían ser considerados expertos en el dominio del problema planteado y en cuáles aspectos podrían asesorar.

   d) Utilizando el cuestionario elaborado en el punto b, realizar una entrevista a un compañero que represente al usuario creador de preguntas, y en base a las respuestas, elaborar un bosquejo del documento de requerimientos del programa.

2. Crear un documento en grupo resumen, con imágenes, del capítulo 5 del libro Ingeniería de Software de Roger Pressman (7ma edición). *“Comprensión de los requerimientos”*

Próximamente >> Requerimientos y Análisis Inicial.

Como obtener información

Suposiciones que debemos combatir

Documento de Requerimientos

Dominio del Problema

Clases y objetos candidatos

Glosario

Bibliografía.

Fowler, M & Scott, K. (1999). *UML Gota a Gota*. Addison Wesley