Universidad Estatal a Distancia

Cátedra de Ciencias Exactas y Naturales

Curso:

Ingeniería del Software

Código:

3300

Tema:

Los fundamentos del diseño

Año:

2017

Los Fundamentos del diseño

A través de la historia de la ingeniería del software ha evolucionado un conjunto de conceptos fundamentales de diseño de software, aunque el grado de interés en cada concepto ha variado con los años, han pasado la prueba del tiempo ofreciendo cada uno al ingeniero de software fundamentos sobre el cual pueden aplicarse métodos de diseño más elaborados.

El diseño de Software juega un papel importante en el desarrollo de software lo cual permite al ingeniero de software producir varios modelos del sistema o producto de que se va a construir el mismo que forman una especie de plan de la solución de la aplicación. Estos modelos pueden evaluarse en relación con su calidad y mejorarse antes de generar código, de realizar pruebas y de que los usuarios finales se vean involucrados a gran escala. El diseño es el sitio en el que se establece la calidad del software.

El software no es el único campo donde el diseño se encuentra inmiscuido. En general podemos ver el diseño como una forma para resolución de problemas. El problema sin solución definitiva es interesante en términos de comprensión del diseño. Un número de otras nociones y conceptos son también de interés en la comprensión del diseño en su sentido general, objetivos, limitaciones, alternativas, representaciones y soluciones.

Dentro del modelo de diseño es necesario que las clases de diseño colaboren con alguna otra.

Es una medida de la interconexión entre los módulos de la estructura de un programa. Depende de la complejidad de la interfaz entre los módulos, el punto en el que se entra o se hace referencia al módulo y qué datos pasan a través de la interfaz. Intentamos conseguir el menor nivel posible de acoplamiento. Las conexiones sencillas entre los módulos hacen que el software sea más fácil de entender y menos dado al efecto ola.

Acoplamiento: La fuerza de las relaciones entre los módulos.

Acoplamiento de datos: Está subordinado al módulo y se accede a él por medio de una lista convencional de argumentos a través de la cual se pasan los datos.

Acoplamiento de marca: Cuando en vez de argumentos simples se pasa una porción de la estructura de datos se pasa por la interfaz del módulo.

Acoplamiento de control: Se pasa un indicador de control (una variable que controla las decisiones en el módulo subordinado).

Acoplamiento externo: Cuando los módulos están atados a un entorno externo al software. Por ejemplo, las I/O y los dispositivos.

Acoplamiento común: Varios módulos hacen referencia a un área global de datos.

Acoplamiento de contenido: Un módulo hace uso de datos o de información de control mantenidos dentro de los límites de otro módulo. Cuando se realiza una bifurcación hacia la mitad de otro módulo.

Una clase de diseño cohesiva tiene un conjunto de responsabilidades pequeño y enfocado, y aplica atributos y métodos de manera sencilla de implementar dichas responsabilidades.

Cohesión: Como están relacionados los elementos que conforman un módulo.

Es una extensión natural del concepto de ocultamiento de la información. Un módulo con cohesión realiza una sola tarea dentro de un procedimiento de software, requiriendo poca interacción con los procedimientos que se realizan en otras partes del programa. Un módulo con cohesión debería hacer una sola cosa.

Hay un número m de módulos que resultarían en un costo de desarrollo mínimo, pero no tenemos la sofisticación necesaria para predecir m con seguridad

- Encapsulación/Ocultar Información

Mediante la agrupación y empaquetado de los elementos y los detalles internos de una abstracción, haciendo que estos detalles sean inaccesibles.

- Separación de la interfaz y la aplicación

La separación de la interfaz y la aplicación implica la definición de un elemento especificando una interfaz pública, conoce a los clientes, aparte de los detalles de cómo se realiza el componente.

- Suficiencia, integridad y primitivismo.

El desarrollo de un sistema con gran cantidad de software requiere que este sea visto desde diferentes perspectivas. Diferentes usuarios (usuario final, analistas, desarrolladores, integradores, jefes de proyecto...) siguen diferentes actividades en diferentes momentos del ciclo de vida del proyecto, lo que da lugar a las diferentes vistas del proyecto, dependiendo de qué interese más en cada instante de tiempo.

La arquitectura es el conjunto de decisiones significativas sobre:

- La organización del sistema
- Selección de elementos estructurales y sus interfaces a través de los cuales se constituye el sistema.
- El Comportamiento, como se especifica las colaboraciones entre esos componentes.
- Composición de los elementos estructurales y de comportamiento en subsistemas progresivamente más grandes.
- El estilo arquitectónico que guía esta organización: elementos estáticos y dinámicos y sus interfaces, sus colaboraciones y su composición.
- Estructuras Arquitectónicas y Puntos de Vista.

Muchas notaciones y lenguajes existen para representar el diseño de artefactos de software. Algunos describen un diseño estructural organizado, otros representan el inicio del software. Estas notaciones son generalmente usadas durante un diseño natural y se pueden usar durante ambos casos. Una representan notaciones que son usadas en el contexto de específicos métodos en las estrategias de diseño y métodos de sub áreas, pero estas categorías son categorizadas en notaciones para describir la estructura estática y la dinámicas vistas.

La construcción de software causa el volumen más alto de elementos de configuración que tienen que ser dirigidos en un proyecto de software (archivos fuente, el contenido, las pruebas, etcétera) típicamente. Por lo tanto, el área de conocimiento de construcción de software también es conectado con el área de conocimiento de administración de configuración de software.

Debido a que la construcción de software depende de herramientas y métodos, es probablemente que el área de conocimiento que está más vinculada es la de las herramientas de ingeniería de software y lo métodos.

Mientras la calidad de software es importante en todas las áreas de conocimiento, código es el último en liberarse en un proyecto de software, por lo tanto, la calidad de software también es vinculado con la construcción de software.