DISEÑO DE SOFTWARE

AUTOR: HELLEN CUBERO

NOVIEMBRE: 2020

San Marcos

Introducción

Antes de iniciar con el proceso de construcción de software es indispensable, diseñar el programa a desarrollar, así como definir la arquitectura o estructura de software y los patrones que pueden implementarse para solventar todos los problemas encontrados, de manera que busca escribir el problema y la solución al problema encontrado.

La presente lectura describe aspectos importantes acerca del diseño de software, la arquitectura del software y los patrones de diseño, así como la referencia a tipos de patrones y ejemplos en los que estos son funcionales.

Además permite comprender la necesidad de realizar el diseño de los requerimientos que conforman un sistema de información.

Contenido

Contexto y aspectos clave

En el contexto del diseño de software algunos autores como Pressman (2005), destaca los siguientes conceptos y aspectos importantes:

"El diseño de software agrupa el conjunto de principios, conceptos y prácticas que llevan al desarrollo de un sistema o producto de alta calidad."(p.183)

"El objetivo del diseño es producir un modelo o representación que tenga resistencia, funcionalidad y belleza". (p.183)

"El diseño de software siempre debe comenzar con el análisis de los datos, pues son el fundamento de todos los demás elementos del diseño. Una vez obtenido el fundamento, se obtiene la arquitectura. Sólo entonces deben realizar otros trabajos del diseño." (p.184)

"El diseño de software comienza una vez que se han analizado y modelado los requerimientos, es la última acción de la ingeniería de software dentro de la actividad de modelado y prepara la etapa de construcción (generación y prueba de código)." (p.184)

Por otra parte, Ruíz & Gónzalez (s.f), definen el diseño de software como la actividad del ciclo de vida del software en la cual se analizan los requisitos para producir una descripción de la estructura interna del software, que sirva de base para su construcción. La salida es un conjunto de modelos y artefactos que registran las principales decisiones adoptadas.
Para obtener más información, haga clic en el enlace

Generalmente la fase de diseño produce un diseño de datos, un diseño arquitectónico, un diseño de interfaz y un diseño procedimental.
El **diseño de datos** se encarga de transformar el modelo de dominio de la información creado durante el análisis.

El **diseño de arquitectónico** define las relaciones entre los principales elementos estructurales del programa.
El **diseño de interfaz** describe cómo se comunica el software consigo mismo, con los sistemas que operan con él y con los operadores que lo emplean.
El **diseño procedural o procedimental** transforma elementos estructurales de la arquitectura del programa en una descripción de los componentes del software. (Universidad de las Américas Puebla, s.f).

Arquitectura de software

La arquitectura del software hace referencia a la estructura de organización de los componentes de un programa (módulos), la forma en la que estos interactúan y la estructura de los datos que utilizan. Sin embargo, en un sentido más amplio, los componentes se generalizan para que representen los elementos de un sistema grande y sus interacciones. (Pressman, 2005)

"Bass, Clements y Kazman (2003) La arquitectura de un programa o sistema de cómputo es la estructura o estructuras del sistema, lo que comprende a los componentes de software, sus propiedades externas visibles y las relaciones entre ellos." (Pressman, 2005, p.207).

Según Pressman (2005), la arquitectura no es el software operativo. Es una representación que permite:

- Analizar la efectividad del diseño para cumplir los requerimientos establecidos.
- Considerar alternativas arquitectónicas en una etapa en la que hacer cambios al diseño todavía es relativamente fácil.
- Reducir los riesgos asociados con la construcción de software.

Según Shaw y Garlan (s.f), el conjunto de propiedades que forman parte de la arquitectura del software son las siguientes:

- **Propiedades estructurales:** este aspecto es la representación del diseño

arquitectónico define los componentes de un sistema (módulos, objetos, filtros, entre otros) y la manera en la que están agrupados e interactúan con otros.

- **Propiedades extrafuncionales**: la descripción del diseño arquitectónico debe abordar la forma en la que la arquitectura del diseño satisface los requerimientos de desempeño, capacidad, confiabilidad, seguridad y adaptabilidad, así como otras características del sistema.
- **Familias de sistemas relacionados**: el diseño arquitectónico debe basarse en patrones repetibles que es común encontrar en el diseño de familias de sistemas similares. El diseño debe tener la capacidad de reutilizar bloques de construcción arquitectónica.(Pressman, 2005, p.190)

Según Ruíz & González (s.f), algunas arquitecturas de software utilizadas en sistemas de información son las siguientes:

- **Capas:** organiza el sistema en un conjunto de capas, cada una provee una serie de servicios a las capas superiores usando los de las capas inferiores.
- **Repositorio compartido:** los datos comunes a los subsistemas se almacenan en una base de datos central o repositorio.
- **Cliente - servidor:** estructura un sistema distribuido en:
    - ***Servidores*** autosuficientes que proveen servicios (impresión, gestión de datos, entre otros).
    - ***Clientes*** que invocan dichos servicios.

Patrones de diseño

Los patrones de diseño brindan soluciones a una serie de problemas comunes que se presentan en el desarrollo de software. Según Larman (2004) un patrón de diseño es una descripción de un problema y su solución, a la cual se le da un nombre y se puede aplicar a nuevos conceptos. Es decir un patrón provee una solución aceptada a un problema común y una terminología para distinguir esa solución. (Rodríguez, Montenegro & Salazar, 2012, p.45)

Tipos de patrones de diseño

Según Ruíz & González (s.f), las principales clases de patrones de diseño, son las siguientes:

- **Creacionales**: se centran en resolver problemas acerca de cómo crear instancias de las clases del sistema.
- **Estructurales:** Se enfocan en como las clases y objetos se combinan para formar estructuras mayores.
- **De comportamiento:** Se enfoca en ser un mediador entre las interacciones que realizan los componentes de un sistema.

Algunos patrones de diseño definidos según Rodríguez, Montenegro & Salazar (2012), son los siguientes:

- **Polimorfismo:** se utiliza cuando existen comportamientos que varían según el tipo de objeto y se desea que el diseño sea extensible. Básicamente se define un nombre para un servicio o comportamiento y se implementan diferentes versiones de este servicio o comportamiento en diferentes objetos.

- **Fábrica o factoría:** Consiste en crear un objeto que encapsule y se encargue de la creación de objetos. Por ejemplo cuando se desea un reproductor multimedia que soporte diferentes formatos, para cada tipo de formato existe una implementación distinta que posee la lógica para decodificar y reproducir casa formato. En este caso se crea un objeto fábrica que dependiendo el tipo de archivo a reproducir cree el objeto correspondiente.

- **Singleton:** se utiliza cuando existe un objeto del cual debe existir únicamente una instancia y es deseable un acceso global a esa instancia. Un ejemplo de su uso, es el acceso al componente del sistema de posicionamiento global (GPS) de un dispositivo móvil.

- **Indirección:** consiste en crear un objeto intermedio que maneje la comunicación entre dos componentes. Se utiliza cuando se requiere una responsabilidad que

involucra dos componentes, pero que asignar esa responsabilidad a cualquiera de los dos componentes disminuiría su reusabilidad.

- **Observador:** se utiliza cuando un objeto genera un evento o realiza un cambio de estado y varios componentes deben reaccionar a este de forma distinta. El patrón consiste en definir una interfaz que implementa todos los objetos que estén interesados en un evento. Por ejemplo, un reproductor de música, si se está escuchando música y desconecta los audífonos el sistema genera un evento que recibe tanto la interfaz, como el controlador del reproductor.

Para obtener más información acerca de los patrones de diseño, haga clic en el enlace

Conclusiones y recomendaciones

A modo de conclusión se logra deducir que la importancia de realizar la arquitectura de un sistema, radica en que permite realizar modificaciones al menor costo posible, sin disminuir la calidad del sistema.

Además gracias al diseño del sistema, se logra identificar si los cambios a realizar tendrán un nivel de complejidad alto o bajo y las implicaciones que tendrá en el sistema como tal.

También se logra concluir que los patrones de diseño constituyen en cierta manera una estandarización para resolver problemas comunes en el desarrollo de software.

Como recomendación, para obtener una mayor comprensión de este tema, es importante leer los enlaces referidos en cada apartado.

Referencias bibliográficas

Montenegro, I., Rodríguez, L., Salazar, G. (2012). Uso de patrones de diseño de software: un caso práctico. Recuperado de https://citic.ucr.ac.cr/sites/default/files/recursos/usopatrones_de_diseno_de_software1.pdf

Ruíz, F., González, M. (s.f). Diseño de software. Recuperado de https://www.ctr.unican.es/asignaturas/is1/is1-t04-trans.pdf

Pressman. (2005). Ingeniería del software un enfoque práctico. Recuperado de http://cotana.informatica.edu.bo/downloads/ld-Ingenieria.de.software.enfoque.practico.7ed.Pressman.PDF

Universidad de las Américas Puebla. (s.f). Capítulo 2: ingeniería de Software, Análisis y Diseño. Recuperado de http://catarina.udlap.mx/u_dl_a/tales/documentos/lis/fuentes_k_jf/capitulo2.pdf

San Marcos

www.usanmarcos.ac.cr

San José, Costa Rica