Unidad III: Arquitecturas de software

La Arquitectura del Software es el diseño de más alto nivel de la estructura de un sistema. Una Arquitectura de Software, también denominada Arquitectura lógica, consiste en un conjunto de patrones y abstracciones coherentes que proporcionan el marco

La arquitectura de software define, de manera abstracta, los componentes que llevan a cabo alguna tarea de computación, sus interfaces y la comunicación entre ellos. Toda arquitectura debe ser implementable en una arquitectura física, que consiste simplemente en determinar qué computadora tendrá asignada cada tarea.

3.1 Descomposición modular

La descomposición modular es un método de diseño proporciona un mecanismo sistemático para descomponer el problema en su problemas, reducirá la complejidad de todo el problema consiguiendo de esta manera una solución modular efectiva.

El diseño modular propone dividir el sistema en partes diferenciadas y definir sus interfaces. Sus ventajas: Claridad, reducción de costos y reutilización

Los pasos a seguir son:

1. Identificar los módulos
2. Describir cada módulo
3. Describir las relaciones entre módulos

Después de elegir la organización del sistema en su totalidad, debemos decir como descomponer los subsistemas en módulos.

No existe una distinción rígida entre la organización del sistema y la descomposición modular. Sin embargo, los componentes de los módulos son normalmente más pequeños, lo que permite usar estilos alternativos de descomposición.

Los módulos se descomponen normalmente de varios componentes del sistema simples. Hay dos estrategias para descomponer un subsistema en módulos:

1. **Descomposición orientada a objetos:** donde se descompone un sistema en un conjunto de objetos que se comunican.
2. **Descomposición orientada a flujos de funciones:** donde se descompone el sistema en módulos funcionales que aceptan datos y los transforman en datos de salida.

Estilos de descomposición modular

En la aproximación Orientada a Objetos, los módulos son objetos con estado privado y operaciones definidas sobre ese estado.

En el modelo de Flujos de Funciones, los módulos son transformaciones funcionales.

Los programas secuenciales son más fáciles de diseñar, implementar, verificar y probar que los sistemas en paralelo.

3.2 Patrones de Diseño

Los patrones de diseño son la base para la búsqueda de soluciones a problemas comunes en el desarrollo de software y otros ámbitos referentes al diseño de interacción o interfaces.

Un patrón de diseño resulta ser una solución a un problema de diseño. Para que una solución sea considerada un patrón debe poseer ciertas características. Una de ellas es que debe haber comprobado su efectividad resolviendo problemas

similares en ocasiones anteriores. Otra es que debe ser reutilizable, lo que significa que es aplicable a diferentes problemas de diseño en distintas circunstancias.

Objetivo de los patrones

- Proporcionar catálogos de elementos reusables en el diseño de sistemas software.
- Evitar la reiteración en la búsqueda de soluciones a problemas ya conocidos y solucionados anteriormente.
- Formalizar un vocabulario común entre diseñadores.
- Estandarizar el modo en que se realiza el diseño.
- Facilitar el aprendizaje de las nuevas generaciones de diseñadores condensando conocimiento ya existente.

3.3 Arquitectura de dominio específico

El reto para el diseño es diseñar el software y hardware para proporcionar características deseables a los sistemas distribuidos y, al mismo tiempo, minimizar los problemas propios a estos sistemas. Es necesario comprender las ventajas y desventajas de las diferentes arquitecturas de sistemas distribuidos.

Existen dos modelos de dominio específico:

1. Modelos genéricos: que son abstracciones de varios sistemas reales.
2. Modelos de referencia: que son modelos abstractos y describen a una clase mayor de sistemas.

Estos modelos arquitectónicos se denominan *arquitecturas de dominio específico.*

1. ***Modelos genéricos.***
2. ***Modelos de referencia.***

No hay, desde luego una distinción rígida entre estos tipos de modelos. Los modelos genéricos también pueden servir como modelos de referencia. Hacemos aquí la distinción entre ellos debido a que los modelos genéricos pueden reutilizase directamente en un diseño. Los modelos de referencia se usan normalmente para comunicar conceptos del dominio y comparar o evaluar posibles arquitecturas.

Hay dos tipos de modelos arquitectónicos obtenidos a partir de varios sistemas reales.

a) ***Modelos genéricos.*** Son abstracciones obtenidas a partir de varios sistemas reales. En capsulan las características principales de estos sistemas. Por ejemplo, en sistemas de tiempo real, podría haber modelos arquitectónicos genéricos de diferentes tipos de sistemas tales como sistemas de recolección de datos o sistemas de monitorización. Se estudian varios modelos genéricos.
b) ***Modelos de referencia.*** Son más abstractos y describen una clase más amplia de sistemas. Constituyen un modo de informar a los diseñadores sobre la estructura general de esta clase de sistemas. Los modelos de referencia normalmente se obtienen a partir de un estudio del dominio de la aplicación.

3.4 Diseño de software de arquitectura multiprocesador

A pesar de las grandes mejoras acaecidas en monoprocesadores para algunas aplicaciones no es suficiente. La solución pueden ser los sistemas multiprocesadores debido a que es la solución más sencilla, natural y con mejor coste-prestaciones. Además, las mejoras en microprocesadores cada vez más son complejas; cada avance implica crecer en complejidad, potencia y superficie; quizás se lenta pro es una clara mejora en el software, que permite explotar el paralelismo.

Existen dos factores claves para la extensión de multiprocesadores:

1. Flexibilidad: El mismo sistema puede usarse para un único usuario incrementado el rendimiento en la ejecución de una única aplicación o para varios usuarios y aplicaciones en un entorno compartido.
2. Coste-rendimiento: Actualmente estos sistemas se basan en procesadores comerciales, por lo que su coste se ha reducido drásticamente.

3.5 Diseño de software de arquitectura

Cliente – Servidor

La **arquitectura cliente-servidor** es un modelo de aplicación distribuida en el que las tareas se reparten entre los proveedores de recursos o servicios, llamados servidores, y los demandantes, llamados clientes. Un cliente realiza peticiones a otro programa, el servidor, quien le da respuesta. Esta idea también se puede aplicar a programas que se ejecutan sobre una sola computadora, aunque es más ventajosa en un sistema operativo multiusuario distribuido a través de una red de computadoras.

En esta aproximación, y con el objetivo de definir y delimitar el modelo de referencia de una arquitectura Cliente/Servidor, debemos identificar los componentes que permitan articular dicha arquitectura, considerando que toda aplicación de un sistema de información está caracterizada por tres componentes básicos:

- Presentación/Captación de Información
- Procesos
- Almacenamiento de la Información

3.6 Diseño de software de arquitectura distribuida

Arquitectura distribuida es Un sistema lógico ordenado que se encarga de conectar varios ordenadores. Este sistema conecta servidor - cliente en tal forma que se pueda distribuir información en forma recíproca.

Sistema cuyos componentes hardware y software que están en ordenadores conectados en red, se comunican y coordinan sus acciones mediante el paso de mensajes, para logro de un objetivo. Se establece la comunicación mediante un protocolo prefijado por un esquema cliente-servidor.

Características:

1. **Concurrencia**
2. **Carencia de reloj global**
3. **Fallos independientes de los componentes**

3.7 Diseño de software de arquitectura de tiempo real

Un sistema de tiempo real es un sistema que debe responder a estímulos externos en un período definido y finito de tiempo. La corrección de las respuestas ofrecidas no sólo depende de la respuesta generada, sino también del tiempo empleado en generarla. Una respuesta no generada en el tiempo estipulado, dependiendo del sistema, puede ser considerada como no válida en absoluto.