Taller Arquitectura de software GA4-220501095-AA2-EV06

APRENDIZ:

VALENTINA VARGAS SANCHEZ

ANGELA GUALDRON DULCEY

HENRY ANDRES MORALES GARZON

YERMAN AUGUSTO HERNANDEZ SAENZ

INSTRUCTOR

CENTRO DE SERVICIOS FINANCIEROS

SENA- REGIONAL DISTRITO CAPITAL

ANALISIS Y DESARROLLO DEL SOFTWARE

FICHA: 2627062

2023

INTRODUCCIÓN

De manera análoga al campo de la ingeniería y construcción, se puede relacionar la arquitectura de software como lo es el arquitecto en la construcción a la disciplina que se encarga previo a la construcción - codificación, de tener una serie de planos, modelos y detalles que permitan que la construcción de la edificación se dé con una mayor facilidad y se pueda construir la mejor edificación dentro de lo posible. El no crear un diseño preliminar desde etapas tempranas del desarrollo puede limitar severamente el que el producto final satisfaga las necesidades de los clientes. Además, se pueden dar sobrecostos elevadísimos con las correcciones relacionadas con problemas en la arquitectura. Es así que la arquitectura de software juega un papel fundamental dentro del desarrollo y en este taller los aprendices quieren profundizar en los aspectos principales de la arquitectura de software y comprender como ésta se puede elaborar y cuáles son sus componentes principales.

SOLUCION ACTIVIDAD

¿Qué entiende por arquitectura de software?

Cuando se habla de arquitectura de software nos referimos a la construcción del sistema y se crea en etapas tempranas del desarrollo. La arquitectura de software tiene dos propósitos primarios: Velar por la calidad (desempeño, seguridad, modificabilidad), y servir como guía en el proceso de desarrollo. Las decisiones primarias en el desarrollo deben hacerse en etapas tempranas debido a que si no se realizan desde el principio puede causar limitaciones en el producto final y puede generar costos adicionales. Es así como arquitectura de software cumple un papel tan importante en el desarrollo de Software.

¿Cuál es su función?

Una arquitectura de software se selecciona y diseña con base en objetivos (requisitos) y restricciones. Los objetivos son aquellos prefijados para el sistema de información, pero no solamente los de tipo funcional, también otros objetivos como el mantenimiento, la auditoría, flexibilidad e interacción con otros sistemas de información. Las restricciones son aquellas limitaciones derivadas de las tecnologías disponibles para implementar sistemas de información. Unas arquitecturas son más recomendables de implementar con ciertas tecnologías mientras que otras tecnologías no son aptas para determinadas arquitecturas. Por ejemplo, no es viable emplear una arquitectura de software de tres capas para implementar sistemas en tiempo real. 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 implementarle en una arquitectura física, que consiste simplemente en determinar qué computadora tendrá asignada cada tarea.

¿Cómo se elabora la arquitectura de software?

Pese que el patrón que se escoja en una arquitectura son muy distintos, las arquitecturas se determinan por las siguientes etapas:

- **Requerimientos:** Esta etapa se recolecta la información y se documentan los requerimientos que influyen en la arquitectura de la aplicación.
- **Análisis:** Se generar los casos de uso a nivel de borrador hasta tener la información necesaria para cada caso.
- **Diseño:** Aquí se define el uso de tecnologías adecuadas para resolver el problema teniendo en cuenta los patrones de diseño.
- **Documentación:** Con el diseño definido es necesario comunicarlo de manera eficiente y es importante crear documentación que sirva como referencia a todos y sea el marco de trabajo para todos.
- **Evaluación:** Luego de tener la documentación, esta se evalúa par ver con todos los involucrados si hay algo en el diseño que pueda no funcionar y reformarlo, esta evaluación se debería hacer posterior teniendo métricas por ejemplo del rendimiento de la aplicación.

¿Cómo lograr una buena arquitectura?

La Arquitectura de Software se relaciona con aspectos como rendimiento, usabilidad, presupuesto, tecnología e incluso cuestiones estéticas. Para implementarla de manera adecuada es recomendable apoyarse de una metodología de desarrollo.

La forma en la que se definen las directrices de la arquitectura suele depender de las necesidades del negocio, ya que pueden ser muy rígidas o ajustables a un proyecto en específico.

1. Establece sistemas robustos, pero libres de frameworks. Esto permite que la estructura sea estable e independiente.

2. Ajusta la construcción y el uso de la base de datos. La base de datos es la que debe alinearse a tus necesidades de negocio, no al revés.

3. Elige las herramientas necesarias para optimizar procesos. Considera la posibilidad de unificar servicios de automatización de sistemas e integración de aplicaciones, colocándolos de manera estratégica desde un inicio.

4. Auditable y testable. Es importante crear un ambiente donde el código pueda ser evaluado, medido y regulado.

¿Cuáles son los elementos de diseño de una arquitectura de software?

Factores importantes que influyen la aptitud de la arquitectura de software son la planificación de proyectos, el análisis de riesgo, la organización, el proceso de desarrollo, los ciclos de trabajo, el hardware, la garantía de calidad y los requerimientos.

- **Patrones**: Dan una descripción de los elementos y el tipo de relación que tienen junto con un conjunto de restricciones sobre cómo pueden ser usados. Un patrón arquitectónico expresa un esquema de organización estructural esencial para un sistema de software, que consta de subsistemas, sus responsabilidades e interrelaciones. En comparación con los patrones de diseño, los patrones arquitectónicos tienen un nivel de abstracción mayor.
- **Tácticas:** Las tácticas son decisiones de diseño que influyen en el control de la respuesta de un atributo de calidad. A diferencia de los patrones de diseño, las tácticas son soluciones menos detalladas y están enfocadas a atributos de calidad específicos. Se aplican en conjunto con los patrones durante el diseño de la arquitectura de un sistema.
- **Frameworks:** Estos son elementos reutilizables de software que proveen funcionalidades genéricas enfocadas a resolver cuestiones recurrentes. Los frameworks típicamente se basan en patrones y tácticas. Un ejemplo de ello es el framework Swingpara creación de interfaces de usuario en Java, que aplica patrones como Modelo-Vista-Controlador (MVC), Observador y Composite; e incorpora tácticas de modificabilidad y usabilidad tales como la generalización de módulos y la separación de elementos de interfaz de usuario.
- **Modelos de dominio:** Describe las abstracciones con sus atributos y las relaciones entre ellas. Un modelo de dominio es por lo habitual independiente de la solución y, de hecho, un mismo modelo de dominio podría usarse para crear distintas soluciones.
- **Componentes COTS**: Partes de aplicaciones, o bien, aplicaciones completas listas para ser integradas. Este tipo de componentes se incorpora por lo habitual de forma binaria previa configuración mediante algún mecanismo previsto para tal propósito.
- **Servicios web:** Son interfaces que encapsulan sistemas completos, los cuales pueden ser parte de la compañía para la cual se desarrolla el sistema, o bien, pertenecer una empresa distinta.