Arquitectura de Software en el proyecto Experiencias Significativas

Catalina Cometa Fierro
SENA, Neiva, Huila, Colombia — catalina.cometa@soy.sena.edu.co

***Resumen*—La Arquitectura de Software es uno de los principios fundamentales para la construcción de sistemas eficientes, escalables y mantenibles. En este artículo se expone cómo estos conceptos fueron aplicados en el desarrollo del proyecto "Experiencias Significativas", una plataforma diseñada para registrar, evaluar y visualizar actividades significativas dentro de una institución educativa.**

***Index Terms*—arquitectura de software, capas, servicios, modularidad, experiencias significativas**

I. Introducción

La Arquitectura de Software es esencial en el proceso de sistemas. No se trata solo de líneas de código, sino de cómo se organizan las piezas, cómo se comunican entre sí y cómo todo esto permite que una aplicación funcione de manera segura, confiable y escalable.

El proyecto "Experiencias Significativas" está pensado para ayudar a los docentes de una institución educativa a organizar y evaluar sus actividades formativas y administrativas. Para su construcción se aplicaron conceptos fundamentales de la Arquitectura de Software, como la organización por capas, el uso de servicios y el diseño orientado a componentes.

Se aplicaron principios fundamentales:

- **Organización por capas**: permite separar responsabilidades y tener un orden.
- **Uso de Servicios**: facilita la comunicación entre componentes.
- **Diseño orientado a componentes**: asegura flexibilidad y reutilización.

II. Trabajos relacionados

La literatura sobre Arquitectura de Software ha establecido principios fundamentales para la construcción de sistemas robustos y mantenibles.

Kruchten [1] propone el modelo 4+1 vistas, estableciendo un marco conceptual para documentar y comunicar la arquitectura de sistemas complejos. Cardacci [2] presenta una arquitectura académica para la comprensión del desarrollo por capas, destacando la importancia de la separación de responsabilidades.

Fernández [3] discute los fundamentos de la arquitectura de software y su impacto en la calidad del producto final. Jimenez-Torres et al. [4] realizan una aproximación al estado del arte de los lenguajes de patrones arquitectónicos, proporcionando un marco de referencia para la toma de decisiones.

Navarro et al. [5] abordan la integración de la arquitectura de software en metodologías ágiles, mientras que Romero [6] analiza esquemas y servicios en la arquitectura moderna. Vera [7] explora la relación entre arquitectura de software y programación orientada a objetos, y Cambarieri et al. [8] presentan una implementación guiada por el dominio.

III. Desarrollo

III-A. Aplicación de los conceptos en el proyecto "Experiencias Significativas"

extbf1. Arquitectura por Capas en el Backend

Una de las decisiones más importantes fue la implementación de una arquitectura por capas en el backend, siguiendo buenas prácticas que destacan la necesidad de una estructura interna y organizada.

El sistema se construyó con diferentes niveles:

- **Controladores**: reciben las solicitudes de los usuarios y envían una respuesta.
- **Servicios**: donde se conecta la lógica de negocio, es decir, las reglas que dan el funcionamiento del sistema.
- **Repositorios**: responsables de comunicarse con la base de datos y garantizar que la información fluya de manera clara y eficiente.
- **Entidades**: permiten manejar los datos de manera ordenada y coherente, evitando confusiones y facilitando la interacción entre capas.

Esto asegura que el sistema sea más claro, fácil de mantener y preparado para crecer en el futuro.

IV. Implementación del software

El sistema .Experiencias Significativas"se implementó siguiendo una arquitectura por capas que separa las responsabilidades y facilita el mantenimiento del código.

IV-A. Arquitectura del sistema

La arquitectura implementada organiza el código en capas claramente definidas: controladores para manejar las peticiones, servicios para la lógica de negocio, repositorios para el acceso a datos, y entidades para representar los modelos del dominio.

Figura 1: Diagrama de arquitectura por capas implementada en el sistema.

IV-B. Fragmento de código

Ejemplo de implementación de una función con tipado estático:

```python
def suma(a: int, b: int) -> int:
    """Suma dos numeros enteros.

    Args:
        a: Primer operando
        b: Segundo operando

    Returns:
        La suma de a y b
    """
    return a + b
```

IV-C. Tecnologías utilizadas

La selección de tecnologías se basó en criterios de escalabilidad, mantenibilidad y facilidad de uso, priorizando herramientas que facilitaran la implementación de la arquitectura por capas.

V. Resultados

El uso de una arquitectura por capas y la aplicación de principios de modularidad y servicios en el proyecto "Experiencias Significativas" permitió observar mejoras claras en la organización y mantenibilidad del sistema.

- **Claridad y orden**: La separación en controladores, servicios, repositorios y entidades facilitó la comprensión y el mantenimiento del código.
- **Facilidad para agregar nuevas funcionalidades**: Gracias a la modularidad, fue posible extender el sistema sin afectar otras partes.
- **Reducción de errores**: La estructura clara ayudó a detectar y corregir errores rápidamente.
- **Escalabilidad**: El sistema quedó preparado para crecer y adaptarse a nuevas necesidades.

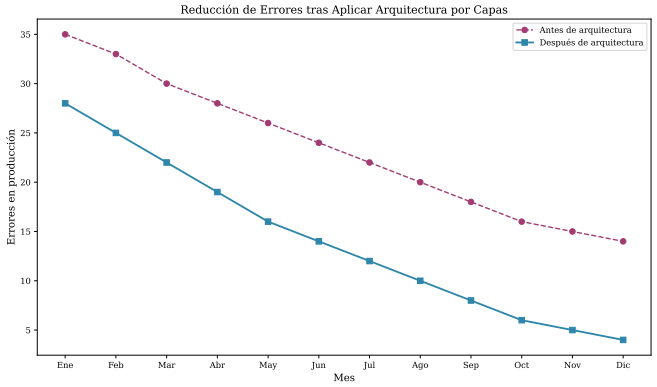

Figura 2: Reducción de errores en producción tras aplicar arquitectura por capas.

Por ejemplo, al implementar nuevas formas de visualización de actividades, solo fue necesario modificar el componente correspondiente, sin afectar la lógica de negocio ni la base de datos. Además, la comunicación entre servicios permitió integrar futuras funcionalidades como reportes automáticos o análisis de datos.

VI. Discusión

La experiencia con el proyecto "Experiencias Significativas" confirma que aplicar principios de Arquitectura de Software aporta beneficios tangibles en proyectos educativos y de gestión.

extbfVentajas:

- La organización por capas permitió separar responsabilidades y facilitó el trabajo colaborativo.
- El uso de servicios y componentes hizo posible la integración de nuevas funcionalidades sin grandes cambios en el sistema.
- La claridad en la estructura redujo el tiempo de resolución de problemas y mejoró la calidad general del software.

extbfLimitaciones y retos:

- La implementación inicial requiere mayor planificación y diseño.
- Es necesario que todo el equipo comprenda y respete la estructura definida para evitar desorden a futuro.

En conclusión, la arquitectura aplicada no solo mejoró el desarrollo y mantenimiento, sino que también sentó las bases para la evolución futura del sistema. Se recomienda a otros equipos considerar estos principios desde el inicio de sus proyectos para obtener resultados similares.

VII. Conclusión

La Arquitectura de Software es la base que asegura que los sistemas sean robustos y fáciles de mantener. Entre estos conceptos se encuentra la modularidad, la separación por capas, la orientación a servicios y el diseño estructurado, todos ellos fundamentales para dar forma al proyecto "Experiencias Significativas".

Apéndice

- **Datos**: fuente, versión, licencias, anonimización.
- **Código**: repositorio, commit hash, instrucciones de ejecución.
- **Entorno**: SO, versión de compiladores, dependencias, semillas.
- **Procedimiento**: pasos exactos para replicar resultados.
- **Resultados**: tablas/figuras generadas automáticamente en `build/`.

Referencias

[1] P. Kruchten, «The 4+1 View Model of Architecture,» *IEEE Software*, vol. 12, n.º 6, págs. 42-50, 1995.

[2] D. G. Cardacci, «Arquitectura de software académica para la comprensión del desarrollo de software en capas,» Serie Documentos de Trabajo, inf. téc. 574, 2015.

[3] L. F. Fernández, «Arquitectura de software,» *Software Guru*, vol. 2, n.º 3, págs. 40-45, 2006.

[4] V. H. Jimenez-Torres, W. Tello-Borja y J. I. Rios-Patiño, «Lenguajes de patrones de arquitectura de software: una aproximación al estado del arte,» *Scientia et technica*, vol. 19, n.º 4, págs. 371-376, 2014.

[5] M. E. Navarro, M. P. Moreno, J. Aranda, L. Parra, J. R. Rueda y J. C. Pantano, «Integración de arquitectura de software en el ciclo de vida de las metodologías ágiles,» en *XIX Workshop de Investigadores en Ciencias de la Computación (WICC 2017, ITBA, Buenos Aires)*, sep. de 2017.

[6] P. Á. Romero, «Arquitectura de software, esquemas y servicios,» *Ingeniería Industrial*, vol. 27, n.º 1, pág. 1, 2006.

[7] J. B. V. Vera, «Arquitectura de software con programación orientada a objeto,» *Polo del Conocimiento: Revista científico-profesional*, vol. 8, n.º 12, págs. 1497-1515, 2023.

[8] M. Cambarieri, F. Difabio y N. García Martínez, «Implementación de una Arquitectura de Software guiada por el Dominio,» en *XXI Simposio Argentino de Ingeniería de Software (ASSE 2020)-JAIIO 49 (Modalidad virtual)*, 2020.