# Arquitectura de Software, Patrones de Diseño y Metodologías Ágiles

Yordy Erik Nuñez Pineda
Servicio Nacional de Aprendizaje SENA
Email: yordierik05@gmail.com

***Resumen*—Este artículo analiza la intersección entre arquitectura de software, patrones de diseño y metodologías ágiles como herramientas clave para enfrentar los desafíos del desarrollo de sistemas complejos. A través de una revisión de múltiples enfoques, se destacan los beneficios de integrar metodologías ágiles como SCRUM, XP y TDD con arquitecturas robustas, como microservicios, arquitectura hexagonal y el modelo 4+1. También se aborda el papel de los patrones de diseño en la estructuración de sistemas sostenibles y reutilizables. Mediante un análisis comparativo de estudios y casos prácticos, el artículo propone un marco conceptual que combina estas disciplinas para fomentar la escalabilidad, calidad y adaptabilidad en proyectos de software modernos, con aplicaciones particulares en sistemas distribuidos, móviles y empresariales.**

***Abstract*—This article analyses the intersection between software architecture, design patterns and agile methodologies as key tools to meet the challenges of developing complex systems. Through a review of multiple approaches, the benefits of integrating agile methodologies such as SCRUM, XP and TDD with robust architectures such as microservices, hexagonal architecture and the 4+1 model are highlighted. The role of design patterns in structuring sustainable and reusable systems is also addressed. Through a comparative analysis of studies and case studies, the article proposes a conceptual framework that combines these disciplines to foster scalability, quality and adaptability in modern software projects, with particular applications in distributed, mobile and enterprise systems.**

## I. Introducción

El desarrollo de software ha evolucionado significativamente en las últimas décadas, impulsado por la necesidad de sistemas más rápidos, seguros y adaptables. La arquitectura de software, los patrones de diseño y las metodologías ágiles han emergido como pilares fundamentales en esta transformación. Sin embargo, integrarlos de manera efectiva sigue siendo un desafío para muchos equipos de desarrollo.

El propósito de este artículo es sintetizar los conocimientos obtenidos de diversos estudios relacionados con arquitectura de software, patrones de diseño y metodologías ágiles, ofreciendo un marco conceptual que combine estas disciplinas para enfrentar los retos de proyectos modernos.

## II. Arquitectura de Software

La arquitectura de software define la estructura y comportamiento de un sistema. Modelos como el 4+1 de Kruchten y la arquitectura hexagonal enfatizan la modularidad, escalabilidad y adaptabilidad. Por otro lado, los microservicios, aplicados ampliamente en entornos distribuidos, permiten dividir aplicaciones en pequeños componentes independientes que se comunican mediante APIs. Estos enfoques facilitan el desarrollo y mantenimiento de sistemas complejos.

## III. Patrones de Diseño

Los patrones de diseño, como los descritos por la "Gang of Four" (GoF), ofrecen soluciones probadas a problemas comunes de diseño. Patrones como el Observador, el MVC y la Factoría Abstracta han sido fundamentales en la estructuración de sistemas sostenibles. Además, los patrones complementarios y secuenciales permiten mejorar la interacción entre módulos, especialmente en sistemas móviles y distribuidos.

## IV. Metodologías Ágiles

Las metodologías ágiles, como SCRUM y XP, priorizan la entrega continua de valor mediante iteraciones rápidas. Sin embargo, estas suelen relegar la planificación arquitectónica inicial, lo que puede comprometer la calidad estructural del sistema. El desarrollo dirigido por pruebas (TDD) permite validar componentes desde las primeras etapas del proyecto, asegurando calidad y cohesión.

## V. Documentación y Calidad

La documentación de la arquitectura es crucial para mantener la trazabilidad y claridad en proyectos complejos. Modelos como "Vistas y más allá" del SEI y el estándar IEEE 1471 destacan la importancia de crear vistas adaptadas a las necesidades de los stakeholders, asegurando que el diseño cumpla con atributos de calidad como rendimiento, escalabilidad y mantenibilidad.

## VI. Metodología

Para integrar estas disciplinas, se propone un marco que sigue las siguientes etapas:

1. **Diseño Inicial de Arquitectura:** Utilización del modelo 4+1 y principios de arquitectura hexagonal para establecer una estructura preliminar.
2. **Descomposición Modular:** Aplicación de patrones de diseño para definir componentes y relaciones.
3. **Implementación Ágil:** Uso de SCRUM con un Sprint 0 para documentar y validar la arquitectura inicial.
4. **Pruebas Iterativas:** Implementación de TDD para garantizar la calidad desde el inicio.
5. **Adaptación a Contextos Específicos:** Uso de microservicios para sistemas distribuidos y capas para entornos móviles.

## VII. Resultados y Discusión

La integración de arquitecturas bien documentadas con metodologías ágiles y patrones de diseño proporciona múltiples beneficios:

- **Escalabilidad:** Los microservicios permiten que las aplicaciones crezcan de manera modular y flexible.
- **Mantenibilidad:** La documentación de vistas y el uso de patrones de diseño aseguran la claridad en las relaciones entre componentes.
- **Calidad:** El uso de TDD y patrones bien testeados garantiza un código más robusto y sostenible.
- **Adaptabilidad:** La arquitectura hexagonal y el diseño basado en casos facilitan la adaptación del sistema a nuevos requerimientos.

Sin embargo, también se identificaron desafíos, como la necesidad de equilibrar la agilidad con la planificación arquitectónica y la gestión de la complejidad en sistemas distribuidos.

## VIII. Conclusiones

Este artículo demuestra que la integración de arquitectura de software, patrones de diseño y metodologías ágiles es clave para el éxito de proyectos modernos de software. La combinación de principios arquitectónicos con prácticas ágiles no solo mejora la calidad, sino que también aumenta la flexibilidad y la eficiencia en el desarrollo.

Como trabajo futuro, se propone evaluar este marco en proyectos reales de diferentes industrias, así como explorar la integración de herramientas de automatización y DevOps para potenciar aún más su aplicabilidad.

## Referencias


[1] Blas, M. J., Leone, H. P., & Gonnet, S. M. (2019). Modelado y Verificación de Patrones de Diseño de Arquitectura de Software para Entornos de Computación en la Nube.

[2] Sanabria, F. R., & Rodríguez, S. V. (2021). Evaluación de una Arquitectura de Software. Prospectiva, 19(2).

[3] Cristiá, M. (2021). Una Teoría para el Diseño de Software.

[4] Vera, J. B. V. (2023). Arquitectura de software con programación orientada a objeto. Polo del Conocimiento: Revista científico-profesional, 8(12), 1497-1508.

[5] Cambarieri, M., Difabio, F., & García Martínez, N. (2020). Implementación de una arquitectura de software guiada por el dominio. In XXI Simposio Argentino de Ingeniería de Software (ASSE 2020)-JAIIO 49 (Modalidad virtual).

[6] Guarin, J. S. V. (2023). Especificando una arquitectura de software: Software architecture specification. Tecnología Investigación y Academia, 11(2), 170-181.

[7] Blancarte, O. (2016). Introducción a los Patrones de Diseño. México, México DF.

[8] Prieto, C. A., & Madrid, D. A. (2022). Buenas prácticas en la construcción de software: Best practices in software construction. Tecnología Investigación y Academia, 10(2), 149-166.

[9] Carignano, M. C. (2016, June). Representación y razonamiento sobre las decisiones de diseño de arquitectura de software. In XVIII Workshop de Investigadores en Ciencias de la Computación (WICC 2016, Entre Ríos, Argentina).

[10] Pérez, J. S., Cuesta, C. E., & Ossowski, S. (2010). El papel de las tecnologías del acuerdo en la definición de arquitecturas de software adaptativas. In Simposio Argentino de Ingeniería de Software (ASSE 2010)-JAIIO 39 (UADE, 30 de agosto al 3 de septiembre de 2010).

[11] Valbuena, D. C. Modelo de gestión de servicios PKI.

[12] López, D., & Maya, E. (2017). Arquitectura de Software basada en Microservicios para Desarrollo de Aplicaciones Web.

[13] González-Castaño, L. E., Marroquín-Soto, S. V., Maturana-González, G. V., & Manjarrés-Betancur, R. A. (2021). Coffee Challenge: Un juego para la enseñanza de patrones de diseño de software. Revista Politécnica, 17(33), 34-46.

[14] Capel, M. I., Grimán, A. C., & Garví, E. Calidad Ágil: Patrones de Diseño en un contexto de Desarrollo Dirigido por Pruebas.

[15] Garnero, A. B., & Horenstein, N. Utilización de antipatrones y patrones en el análisis de software.

[16] Marticorena, R., López, C., García Osorio, C. I., & Pardo, C. (2002). Aprendizaje práctico de patrones de diseño en asignaturas de programación de nivel III.

[17] Moreno, J. C., Marciszack, M. M., & Groppo, M. A. (2020). Patrones de Usabilidad Temprana en el Modelo Conceptual. AJEA, (5).

[18] Rojas, M., & Montilva, J. (2011). Una arquitectura de software para la integración de objetos de aprendizaje basada en servicios web. In Ninth LACCEI Latin American and Caribbean Conference. Engineering for a Smart Planet, Innovation, Information Technology and Computational Tools for Sustainable Development.

[19] Paez, A. H., Falcón, J. D., & Cruz, A. A. P. (2018). Arquitectura de software para el desarrollo de videojuegos sobre el motor de juego Unity 3D. Revista de I+ D tecnológico, 14(1), 54-64.

[20] González, J. E. F., & García, C. A. O. Propuesta de diseño de arquitectura de software para la gestión, análisis y procesamiento de datos de neurociencia.

[21] Martín, Y. E. (2012). Arquitectura de software. Arquitectura orientada a servicios. Serie Científica de la Universidad de las Ciencias Informáticas, 5(1), 1-10.

[22] Sánchez, A. B., Parejo, J. A., Estrada-Torres, B., Márquez-Chamorro, A. E., del-Río-Ortega, A., & Segura, S. (2023). Aprendiendo arquitectura software a partir de proyectos de código abierto en GitHub.

[23] Cortez, A. A., & Naveda, C. A. Una propuesta de implementación para especificaciones de patrones de comportamiento.

[24] Navarro, M. E., Moreno, M. P., Aranda, J., Parra, L., Rueda, J. R., & Pantano, J. C. (2017, September). Selección de metodologías ágiles e integración de arquitecturas de software en el desarrollo de sistemas de información. In XIX Workshop de Investigadores en Ciencias de la Computación (WICC 2017, ITBA, Buenos Aires).

[25] Astorga, J. A. I., Tirado, J. L. O., Ramírez, R. U. R., Mendoza, J. C. L., & Garzón, A. P. (2019). Clasificación de los patrones de diseño idóneos en programación Android. Revista Digital de Tecnologías Informáticas y Sistemas, 3(1).

[26] Vivas, H. L., Cambarieri, M. G., García Martínez, N., Muñoz Abbate, H., & Petroff, M. (2013). Un marco de trabajo para la Integración de Arquitecturas de Software con Metodologías Ágiles de Desarrollo.

[27] Granada, E. Z., Rodríguez, L. E. S., Montoya, C. E. G., & Uribe, C. A. C. (2014). Arquitecturas de software para entornos móviles. Revista de Investigaciones Universidad del Quindío, 25(1), 20-27.

[28] Gómez, O. Documentando la arquitectura de software.

[29] Moreno, A. M., & Sánchez-Segura, M. (2003, November). Patrones de Usabilidad: Mejora de la Usabilidad del Software desde el Momento Arquitectónico. In JISBD (pp. 117-126).

[30] Roqué Fourcade, L. E., & Arakaki, L. (2015, May). Soporte a la actividad de diseño basado en patrones de diseño. In XVII Workshop de Investigadores en Ciencias de la Computación (Salta, 2015).

[31] Lund, M. I., Chavez, S. B., Ormeño, E. G., Martin, A., & Matturro, G. MODELO DE CASOS DE USO. SU APLICACIÓN EN UN ENTORNO DE DESARROLLO GLOBAL DE SOFTWARE.

[32] Rodríguez, I. Y. A., & Cortes, E. E. P. R. Implementación de MVCC para la generación de Aplicaciones Web en tiempos de entrega reducidos.

[33] Arias-Orezano, J. F., Reyna-Barreto, B. D., & Mamani-Apaza, G. (2021). Repercusión de arquitectura limpia y la norma ISO/IEC 25010 en la mantenibilidad de aplicativos Android. TecnoLógicas, 24(52), 226-241.

[34] Guevara, W. J. T. Metodología para la enseñanza del Desarrollo de Software Modular.