Resumen y reflexión de artículos relacionados con “patrones de diseño”y “arquitectura de software”

Iván Andrés MurciaInstructor:
Jesús Ariel González Bonilla

Servicio Nacional de Aprendizaje SENA
Centro de la industria, la empresa y los servicios
Análisis y desarrollo de software

Neiva-Huila
20 de noviembre del 2024

Tabla de contenido

Biografía

Mi nombre es Iván Andrés Murcia, nací el 17 de octubre de 2005 en Bogotá ytengo 19 años. Vivo en Neiva con mis padres y dos hermanos. Terminé el bachillerato en 2022, con un título técnico y un certificado en diseño e integración de multimedia del Sena, que me abrió puertas para seguir formándome. Actualmente, estudio Análisis y Desarrollo de Software y un técnico en Investigación Judicial, áreas que combinan análisis, solución de problemas e investigación.

El deporte ha sido esencial en mi vida. Mi pasión por el fútbol comenzó a los 6 años, entrenando y participando en torneos locales y nacionales, como la CopaPony Fútbol. También he practicado voleibol, baloncesto y ping pong, donde obtuve el segundo lugar en un torneo reciente. Estas experiencias me enseñaron disciplina, trabajo en equipo y perseverancia, valores que aplico en mi vida personal y profesional para enfrentar cualquier desafío.

1. Comparación del rendimiento entre arquitecturas monolíticas y de microservicios en sistemas web

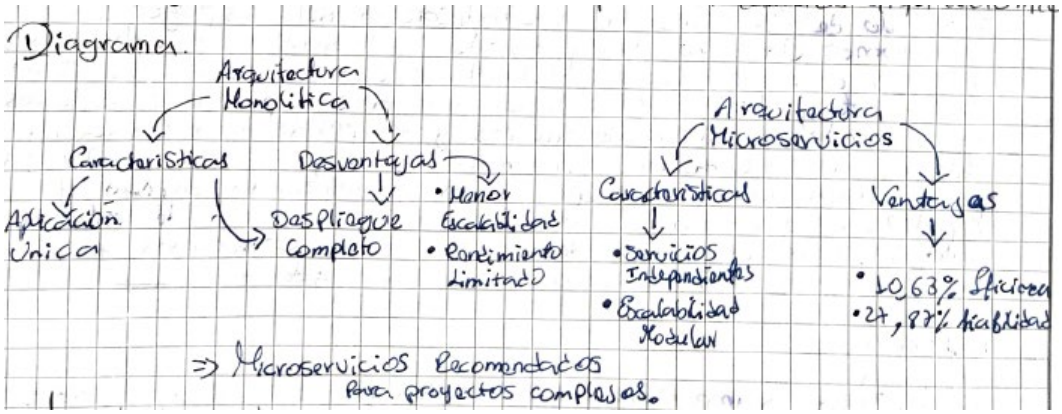
Resumen

El artículo compara las arquitecturas monolíticas y de microservicios en términos de eficiencia y fiabilidad en sistemas web. A través de un estudio experimental, se evaluó cómo estas arquitecturas impactan el consumo de recursos, tiempo de respuesta y conexiones simultáneas. Los resultados muestran que la arquitectura de microservicios mejora la eficiencia en un 10.63% y la fiabilidad en un 27.87% frente a la monolítica. Se concluye que la adopción de microservicios ofrece ventajas significativas para entornos exigentes, como el desarrollo de software empresarial. Se recomienda evaluarlos costos de implementación y estudiar herramientas específicas para maximizar su efectividad.

Reflexión

La lectura de este artículo resalta la importancia de elegir la arquitectura adecuada para los sistemas web, especialmente en proyectos complejos. Me hizo reflexionar sobre cómo una arquitectura bien diseñada puede marcar la diferencia no solo en el rendimiento técnico, sino también en la experiencia delusuario. Las ventajas de los microservicios, como la modularidad y escalabilidad, son especialmente relevantes en un mundo donde la tecnologíaavanza rápidamente y las necesidades de los usuarios cambian constantemente. Este tema me ayuda a valorar la importancia de considerar la viabilidad técnica y económica antes de implementar cambios arquitectónicos.

Diagrama



Bibliografía

- Toledo Azorza, M. A. J. (2017). Comparación del rendimiento de las arquitecturas monolíticas y microservicios en los sistemas web. *Facultadde Ingeniería y Arquitectura*.

2. Arquitecturas de Microservicios para Aplicaciones en Contenedores

Resumen

Este artículo explora cómo las arquitecturas de microservicios han transformado el desarrollo de aplicaciones al combinarse con tecnologías como contenedores y orquestadores. Utilizando Docker para la contenerización y Kubernetes para la gestión, se asegura alta disponibilidad, escalabilidad y despliegues rápidos en entornos Cloud. Además, se aplican patrones de desarrollo como inyección de dependencias y CQRS, lo que permite diseñar aplicaciones modulares y eficientes. La propuesta se enfoca en aprovechar plataformas como Azure, que integran herramientas de desarrollo y despliegue continuo, asegurando una infraestructura adaptable y robusta.

Reflexión

El artículo muestra cómo los microservicios representan un cambio fundamental en el diseño de aplicaciones modernas. Me hizo pensar en cómo la contenerización y los orquestadores no solo optimizan los recursos, sino quetambién democratizan el acceso a tecnologías avanzadas. Este enfoque modular es especialmente valioso en un mundo donde las demandas de los usuarios evolucionan rápidamente. Me inspira a valorar las metodologías ágilesy el desarrollo en la nube como caminos para garantizar flexibilidad y competitividad.

Diagrama

Bibliografía

- Jiménez Aliaga, C. (2018). Arquitecturas de microservicios para aplicaciones desplegadas en contenedores. *Universidad Politécnica deMadrid*. Escuela Técnica Superior de Ingeniería de Sistemas Informáticos.

3. Método de Automatización del Despliegue Continuo en la Nube para Microservicios

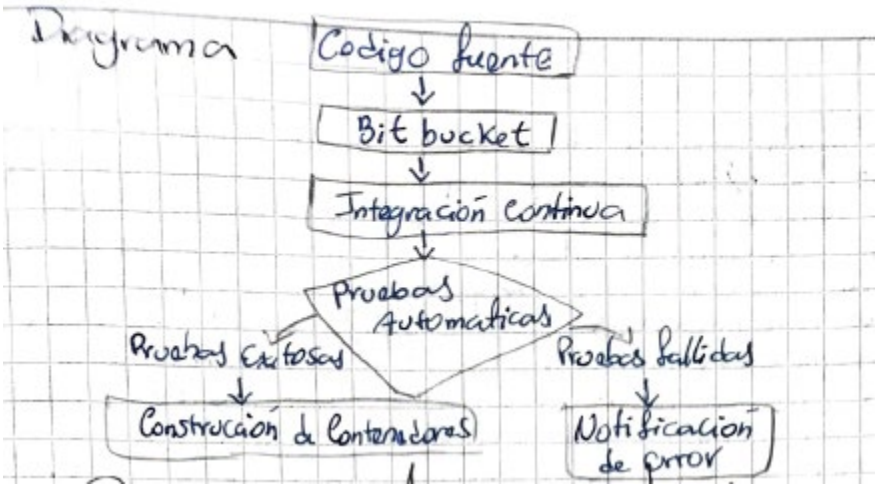
Resumen

El artículo detalla cómo la automatización del despliegue continuo en entornos Cloud ha transformado el desarrollo de aplicaciones basadas en microservicios.Mediante el uso de herramientas como Bitbucket y AWS EC2, se logra automatizar pruebas y despliegues en contenedores, reduciendo errores y tiempos de implementación. Además, se analiza la importancia de las prácticasde DevOps para integrar desarrollo y operaciones, garantizando calidad y rapidez. Se destacan los desafíos en la adopción de esta metodología, como lagestión de transacciones distribuidas y el monitoreo, que requieren soluciones más avanzadas.

Reflexión

Este artículo refuerza la importancia de adoptar procesos automatizados en el desarrollo de software moderno. Me hizo pensar en cómo la integración de pruebas y despliegues automáticos no solo mejora la calidad, sino que tambiénreduce significativamente los riesgos asociados al error humano. Además, resalta la necesidad de seguir innovando en áreas como la seguridad y la gestión de microservicios. Esta metodología parece ideal para organizaciones que buscan adaptarse rápidamente a cambios tecnológicos y de mercado.

Diagrama



Bibliografía

- Vera-Rivera, F. H. (2018). Método de automatización del despliegue continuo en la nube para la implementación de microservicios. *Universidad Francisco de Paula Santander*.

4. Diseño de una Arquitectura Basada en Contenedores para la Integración y el Despliegue Continuo (CI/CD)

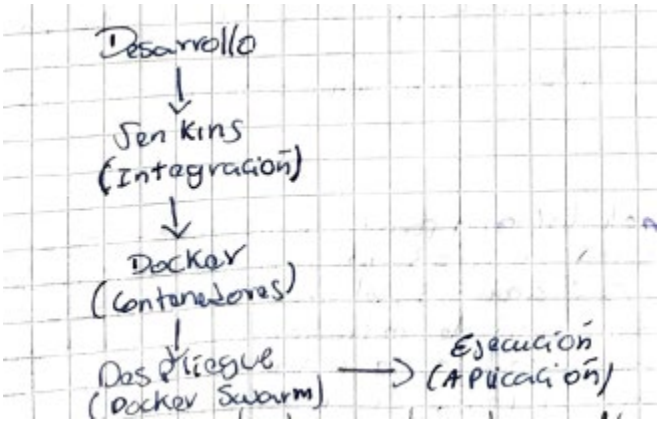
Resumen

El artículo presenta el diseño e implementación de una arquitectura basada en contenedores, diseñada para soportar la integración y el despliegue continuo (CI/CD). Utilizando herramientas como Docker, Jenkins y Docker Swarm, el trabajo detalla cómo se automatizan tareas con scripts de shell, se gestionan aplicaciones en entornos locales y se asegura la escalabilidad en servidores remotos. Además, destaca los beneficios del uso de contenedores sobre máquinas virtuales tradicionales, como mayor eficiencia y menor consumo de recursos. La propuesta integra técnicas como versionado de imágenes y redesde alta disponibilidad, optimizando la ejecución de aplicaciones modernas parapequeñas y medianas
empresas.

Reflexión

Este artículo muestra cómo las tecnologías basadas en contenedores y la automatización CI/CD están transformando la manera de desarrollar y gestionar aplicaciones. Personalmente, me hace valorar la importancia de herramientas como Docker y Jenkins para simplificar tareas repetitivas y mejorar la eficiencia. También me ayuda a reflexionar sobre cómo este enfoquemodular y escalable es clave para proyectos actuales, especialmente en entornos empresariales donde la flexibilidad y la rapidez de implementación son esenciales. La automatización no solo ahorra tiempo, sino que también reduce errores, lo que la convierte en una herramienta indispensable para los equipos de desarrollo.

Diagrama



Bibliografía

- Maggi, D. (2020). Diseño de una arquitectura basada en contenedorespara la integración y el despliegue continuo (CI/CD). *Universidad de Málaga*.

5. Recomendaciones para el Desarrollo de Software Sostenible en Arquitecturas de Microservicios

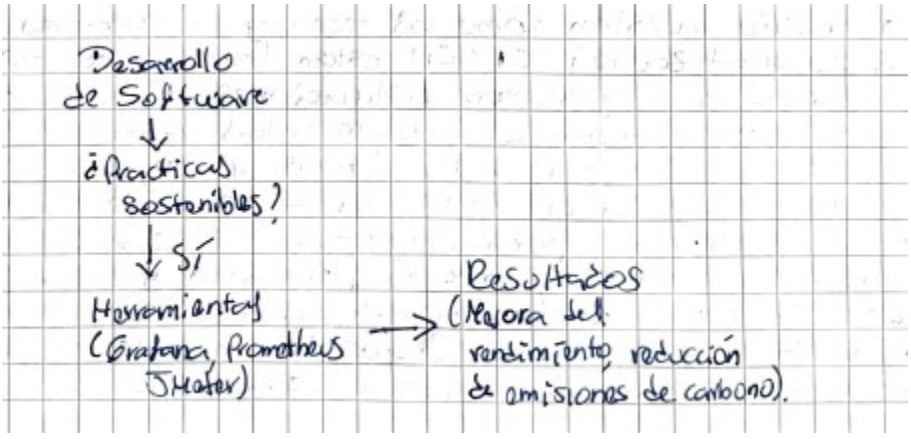
Resumen

El artículo aborda la problemática del impacto ambiental causado por las tecnologías de la información y, específicamente, por las malas prácticas en el desarrollo de software. Propone una serie de recomendaciones para optimizarel consumo energético y minimizar las emisiones de carbono en arquitecturas de microservicios. A través de herramientas como Grafana, Prometheus y JMeter, se realizaron pruebas para medir el rendimiento del hardware con diferentes prácticas de desarrollo. Los resultados evidencian que prácticascomo el uso eficiente de hilos, la implementación de cachés y la elección adecuada de lenguajes y versiones de programación tienen un impacto significativo en la sostenibilidad energética. El estudio busca crear concienciaen la comunidad tecnológica sobre la importancia de adoptar prácticas verdesen el diseño y desarrollo de aplicaciones modernas.

Reflexión

Este artículo es una invitación a reflexionar sobre nuestra responsabilidad como desarrolladores en la reducción del impacto ambiental de las tecnologíasde la información. Me hizo pensar en cómo nuestras decisiones diarias, como el diseño de arquitecturas y la elección de herramientas, pueden contribuir al cambio climático. La idea de un "software verde" no solo tiene un impacto positivo en el ambiente, sino que también mejora el rendimiento de las aplicaciones, lo que es beneficioso para usuarios y empresas. Es motivador saber que pequeños cambios en nuestras prácticas pueden generar grandes resultados y promover un futuro más sostenible.

Diagrama



Bibliografía

- Campos Lucas, C. A. (2023). Recomendación de prácticas de desarrollode software en una arquitectura de microservicios que sean ambientalmente sostenibles. *Universidad Cenfotec*.

6. Definición de una Arquitectura para la Transformación de Software Centralizado a Microservicios en el Ámbito Web

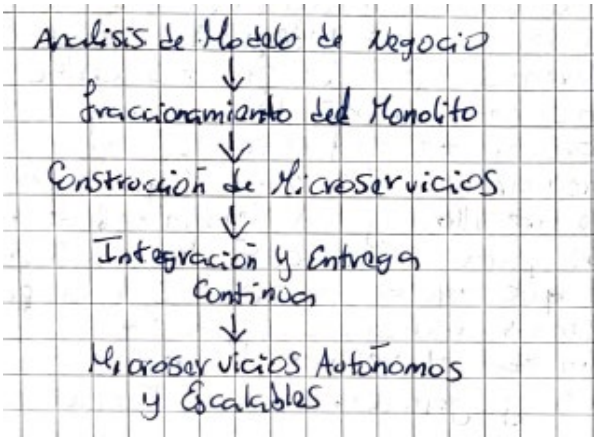
Resumen

Este artículo presenta una metodología para transformar aplicaciones monolíticas en arquitecturas basadas en microservicios. Se organiza en tres bloques principales: análisis del modelo de negocio, fraccionamiento del monolito y construcción de microservicios, e integración y entrega continua. Lainvestigación incluye un caso de prueba práctico que detalla cada etapa del proceso. Se resaltan las ventajas de los microservicios, como escalabilidad, mantenimiento más sencillo y despliegues más seguros. Además, se proponenbuenas prácticas para gestionar la fragmentación del código y su despliegue, garantizando una transición eficiente y confiable.

Reflexión

Este artículo me hizo reflexionar sobre la importancia de adoptar una metodología adecuada para la transformación de software monolítico a microservicios. El enfoque gradual permite minimizar los riesgos asociados a lamigración, asegurando que cada módulo sea funcional y autónomo. Además, resalta la necesidad de contar con herramientas y equipos capacitados para garantizar el éxito del proceso. Personalmente, considero que esta transición es vital para las empresas que buscan adaptarse a entornos tecnológicos cambiantes y mejorar su capacidad de respuesta ante las necesidades del mercado. Implementar microservicios no solo reduce costos a largo plazo, sinoque también fomenta la innovación y la flexibilidad.

Diagrama



Bibliografía

- Gómez Gallego, J. P. (2017). Definición de una arquitectura para la transformación de software centralizado a software basado en microservicios en el ámbito web. *Universidad Tecnológica de Pereira*.

7. Arquitectura de Software para el Desarrollo de Aplicaciones Web Orientadas a Microservicios en TecNM Campus Escárcega

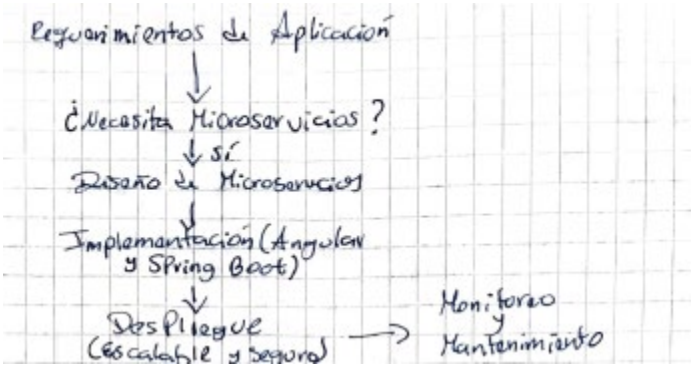
Resumen

El artículo describe la implementación de una arquitectura de software para el desarrollo de aplicaciones web basadas en microservicios, diseñada para reemplazar el enfoque monolítico en el TecNM Campus Escárcega. La propuesta integra tecnologías como Angular para el cliente y Spring Boot para los servicios, enfocándose en reducir tiempos de desarrollo y aumentar la productividad. Se evaluó el desempeño mediante métricas de usabilidad y productividad, obteniendo resultados favorables en comparación con el paradigma monolítico, con mejoras en escalabilidad, modularidad y experienciadel usuario. Además, se implementaron módulos reutilizables que simplifican lacreación de nuevos proyectos y mantienen a los estudiantes y docentes actualizados con tecnologías modernas.

Reflexión

Este artículo muestra una solución práctica para adoptar paradigmas modernosen el desarrollo de software. Me hizo reflexionar sobre cómo los microservicios ofrecen una ventaja clave en términos de escalabilidad y modularidad, características esenciales en un entorno académico donde los recursos y el tiempo son limitados. La transición del paradigma monolítico al de microservicios no solo fomenta la innovación, sino que también asegura que losfuturos profesionales estén alineados con las tendencias tecnológicas actuales.Personalmente, creo que esta arquitectura podría replicarse en otras instituciones, generando un impacto positivo en la formación de estudiantes y en la eficiencia de proyectos académicos y empresariales.

Diagrama



Bibliografía

- Rosado Castellanos, D. U., Pacheco Farfán, I. S., Fuentes Chab, I. H., Cantun Páez, J. C. (2023). Arquitectura de software para el desarrollo deaplicaciones web orientada a micro-servicios en TecNM campus Escárcega. *Programación Matemática y Software, 15*(2), 11-24. DOI: 10.30973/progmat/2023.15.2/2

8. Pruebas de Software para Microservicios: Retos y Estrategias

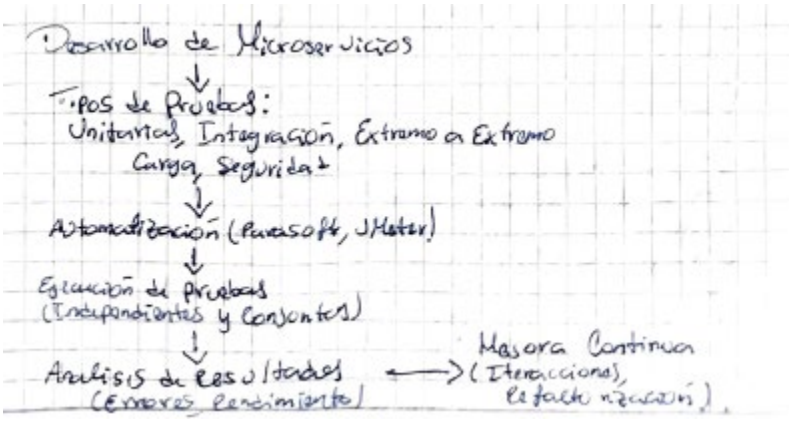
Resumen

El artículo aborda la complejidad de realizar pruebas en aplicaciones basadasen microservicios, donde cada componente debe ser evaluado de manera independiente y en conjunto. Se destacan diferentes tipos de pruebas como unitarias, de integración, de extremo a extremo, de carga y de seguridad.
Asimismo, se enfatiza la importancia de la automatización para enfrentar desafíos como el gran volumen de servicios, las dependencias entre ellos y las demandas de rendimiento. Entre las herramientas clave para la ejecución de estas pruebas se mencionan Parasoft y JMeter, que ayudan a optimizar el proceso y garantizar la calidad en sistemas distribuidos. Finalmente, se proponen estrategias para maximizar el retorno de inversión (ROI) en la ejecución de pruebas y garantizar un software robusto y escalable.

Reflexión

Este artículo me hizo reflexionar sobre la relevancia de las pruebas en sistemas complejos como los basados en microservicios. A menudo, el enfoque está en el desarrollo, pero garantizar que todo funcione correctamente es igual de importante. Me pareció interesante cómo herramientas automatizadas pueden simplificar este proceso, permitiendo detectar errores y mejorar el rendimiento antes de que lleguen a los usuarios. Además, las pruebas de carga y seguridadson un recordatorio de la responsabilidad que tenemos como desarrolladores para proteger los datos y ofrecer experiencias confiables. Creo que estas prácticas deberían integrarse desde el principio en cualquier proyecto de software.

Diagrama



Bibliografía

Laura Mamani, C. A. (2023). Pruebas de Software para Microservicios. *Revista Innovación y Software*, 4(1), 151-160. Universidad La Salle, Arequipa, Perú. Recuperado de https://revistas.ulasalle.edu.pe/innosoft.

9. Modelo de Arquitectura de Software para el Procesamiento de Datos en Arquitecturas Actuales

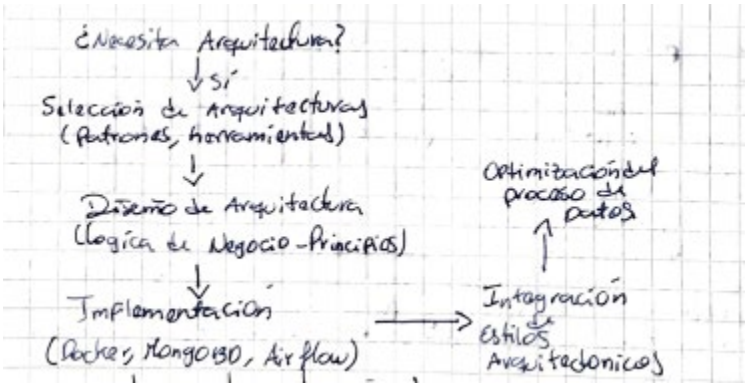
Resumen

Este artículo propone un modelo de arquitectura de software diseñado para abordar los desafíos en el procesamiento de datos masivos. Basándose en patrones arquitecturales y herramientas modernas como Docker, MongoDB yAirFlow, se ofrece una guía comparativa para seleccionar y evaluar arquitecturas adecuadas en función de las necesidades empresariales. El modelo combina técnicas de diseño basadas en lógica de negocio y principios de arquitectura para mejorar la eficiencia y escalabilidad en el manejo de datos.Además, se analiza cómo diferentes estilos arquitectónicos pueden integrarse para superar problemas comunes como la latencia y la corrupción de datos, ofreciendo soluciones optimizadas para entornos dinámicos.

Reflexión

El artículo me ayudó a comprender la importancia de elegir cuidadosamente una arquitectura de software adecuada al contexto y las necesidades del negocio. La propuesta de usar patrones arquitecturales no solo mejora la organización de los datos, sino que también optimiza los recursos, lo que es crucial en un entorno donde el volumen de datos crece exponencialmente. Me pareció interesante cómo tecnologías como Docker y AirFlow simplifican el manejo de datos masivos, al tiempo que promueven la flexibilidad. Esto resalta la necesidad de que las empresas inviertan en capacitación y herramientas querespalden una toma de decisiones informada y efectiva. Implementar un modelo como este no solo impacta en la productividad, sino también en la capacidad de adaptación tecnológica.

Diagrama



Bibliografía

- Quilindo Celis, L. A., & Vega Guarín, J. S. (2021). Modelo de arquitecturade software para el procesamiento de datos en arquitecturas actuales. *Universidad Distrital Francisco José de Caldas*. Facultad de Ingeniería.

10. Arquitectura para la Implementación de Servicios de Video en Redes Móviles: Enfoque en SDN y Segmentación de Red

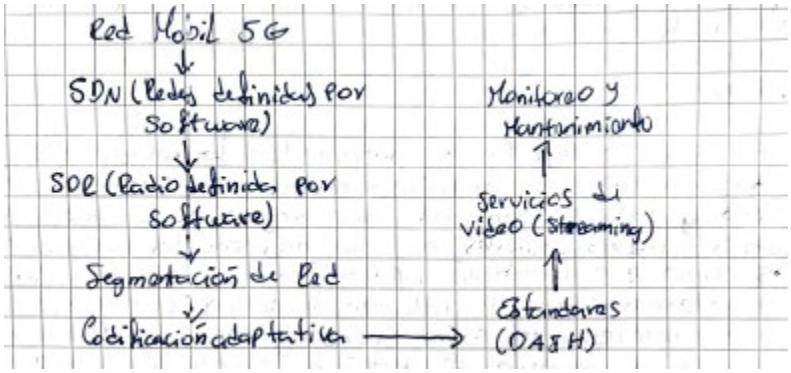
Resumen

Este artículo propone una arquitectura innovadora para ofrecer servicios de videostreaming en redes móviles, utilizando conceptos como redes definidas por software (SDN), radio definida por software (SDR) y segmentación de red. La propuesta destaca por su flexibilidad y escalabilidad, presentando dos perspectivas: funcional e implementativa. Se describen módulos clave como la codificación adaptativa y el uso de estándares como DASH para garantizar unatransmisión eficiente. Además, se analiza la viabilidad de la arquitectura en redes 5G mediante pruebas de concepto, evidenciando mejoras significativas en la calidad del servicio y la reducción de latencia. Esta solución es aplicable tanto en entornos académicos como empresariales, sirviendo como unreferente en la modernización de infraestructuras de red.

Reflexión

Este artículo invita a reflexionar sobre cómo las tecnologías emergentes como SDN y SDR pueden transformar la forma en que consumimos y distribuimos contenido multimedia en redes móviles. Personalmente, me sorprendió cómo lasegmentación de red puede garantizar la calidad del servicio incluso en condiciones de alta demanda. Este enfoque no solo moderniza las infraestructuras existentes, sino que también ofrece una solución más sostenible y flexible para satisfacer las necesidades de la era digital. Creo que adoptar este tipo de arquitecturas es esencial para garantizar que las redes estén preparadas para las demandas futuras, especialmente en servicios comoel videostreaming, donde la experiencia del usuario es clave.

Diagrama



Bibliografía

- Chávez Picon, J. L., Campo Muñoz, W. Y., & Chanchí Golondrino, G. E. (2023). Arquitectura para Implementación de Servicios de Video sobre Redes Móviles mediante Redes Definidas por Software y Segmentaciónde Red. *Revista Colombiana de Tecnologías de Avanzada, 42*(2), 34-40.DOI: 10.30973/RCTA/42/2

11. Lenguajes de Patrones de Arquitectura de Software: Una Aproximación al Estado del Arte

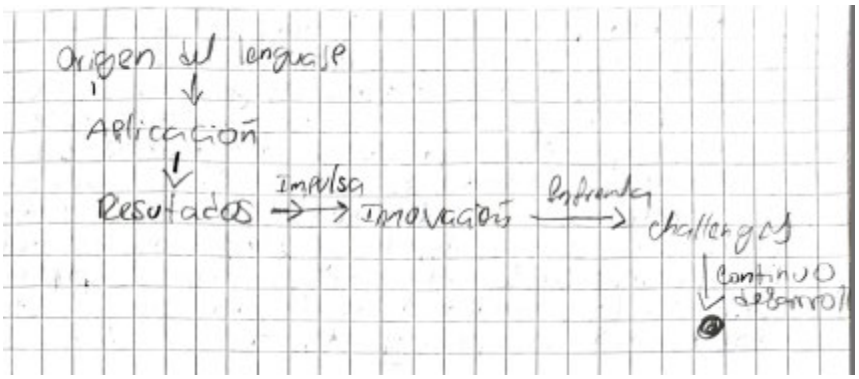
Resumen

Este artículo explora los lenguajes de patrones en la arquitectura de software, desde sus orígenes hasta su estado actual, destacando su importancia en el diseño de sistemas de información. Los lenguajes de patrones ofrecen soluciones a problemas comunes de desarrollo mediante la combinación de patrones individuales en procesos estructurados. Se abordan conceptos clave como la composición de patrones, historias de patrones y secuencias, que permiten la generalización y reutilización de soluciones en diferentes dominios.Asimismo, se presentan ejemplos prácticos, como el lenguaje de patrones parasistemas de seguridad y para arquitecturas e-Business, destacando su aplicabilidad en contextos académicos e industriales.

Reflexión

Este artículo me llevó a reflexionar sobre cómo los lenguajes de patrones son herramientas fundamentales para optimizar el diseño de software. Me llamó laatención cómo estos lenguajes facilitan la resolución de problemas específicosy permiten generar soluciones reutilizables y adaptables. En un mundo donde los sistemas se vuelven cada vez más complejos, esta metodología no solo ahorra tiempo, sino que también garantiza la calidad del software. Además, considero que el uso de lenguajes de patrones fomenta la innovación, ya que combina la experiencia pasada con las necesidades actuales. Este enfoque estructurado debería ser adoptado ampliamente por equipos de desarrollo quebuscan eficiencia y sostenibilidad.

Diagrama



Bibliografía

- Jiménez-Torres, V. H., Tello-Borja, W., & Ríos-Patiño, J. I. (2014). Lenguajes de Patrones de Arquitectura de Software: Una Aproximaciónal Estado del Arte. *Scientia et Technica, 19*(4), 371-376. Universidad Tecnológica de Pereira.

12. Patrones de Diseño, Refactorización y Antipatrones: Ventajas yDesventajas en el Software Orientado a Objetos

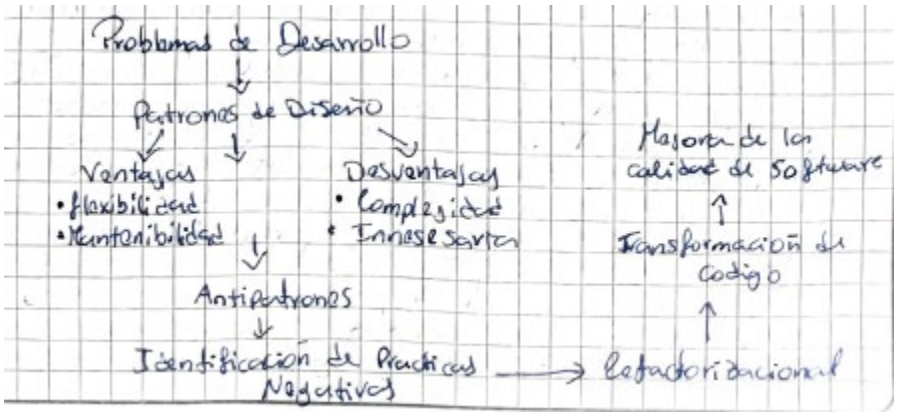
Resumen

El artículo explica los conceptos de patrones de diseño, refactorización y antipatrones, enfocándose en sus ventajas y desventajas al aplicarlos en software orientado a objetos. Los patrones de diseño son soluciones recurrentes que mejoran la flexibilidad y mantenibilidad del software. Por otro lado, los antipatrones identifican prácticas negativas que generan problemas recurrentes, mientras que la refactorización permite transformar estas malas prácticas en código más comprensible y eficiente. El texto destaca la importancia de estas técnicas para evitar errores comunes y optimizar el desarrollo, aunque advierte sobre el riesgo de usarlas de manera inapropiada,lo cual podría aumentar la complejidad innecesariamente.

Reflexión

Este artículo me ayudó a valorar el impacto de las buenas y malas prácticas enel desarrollo de software. Los patrones de diseño son una guía útil para resolver problemas comunes, pero también requieren un uso adecuado para evitar que añadan complejidad. Me pareció interesante cómo los antipatrones no solo identifican fallas, sino que también ofrecen una solución refactorizada para corregirlas. Este enfoque práctico es esencial en un entorno donde el código mal estructurado puede generar retrasos y costos adicionales.
Considero que estas herramientas no solo mejoran la calidad del software, sinotambién fomentan una cultura de aprendizaje continuo entre los desarrolladores.

Diagrama



Bibliografía

- Campo, G. D. (2009). Patrones de diseño, refactorización y antipatrones: Ventajas y desventajas en el software orientado a objetos. *Cuadernos de la Facultad*, 4, 101-135.

13. Una Ontología para la Representación de Conceptos de Diseño de Software

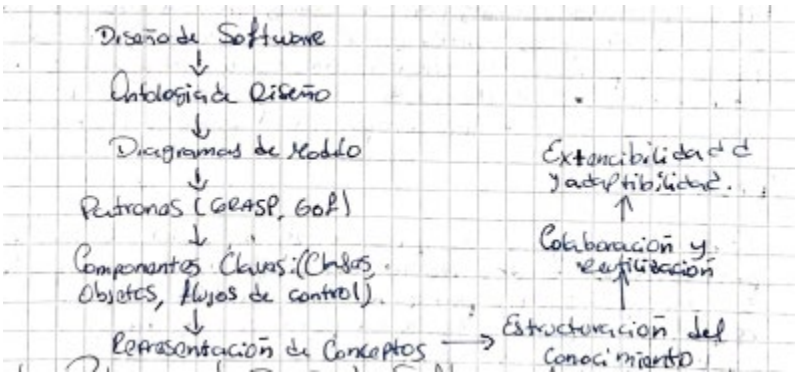
Resumen

El artículo propone una ontología como herramienta para estructurar y organizar el conocimiento relacionado con el diseño de software. Este enfoque busca integrar diagramas de modelado y patrones como GRASP y GoF, esenciales durante la fase de diseño. La ontología permite representar diagramas estructurales y de comportamiento, facilitando su comprensión y reutilización en entornos colaborativos, especialmente educativos. Se identifican componentes clave, como clases, objetos y flujos de control, así como patrones que promueven la cohesión y el bajo acoplamiento. Además, se destaca la posibilidad de extender esta ontología para incorporar nuevos conceptos y adaptarse a las necesidades de la industria.

Reflexión

Este artículo me hizo reflexionar sobre la importancia de estructurar el conocimiento en el diseño de software, un aspecto que suele ser complejo y disperso. Las ontologías, al organizar conceptos como diagramas y patrones, no solo facilitan la enseñanza, sino también la implementación de buenas prácticas en el desarrollo de software. Considero que integrar herramientas como esta en entornos académicos y empresariales podría mejorar la calidad de los proyectos y fomentar la colaboración entre equipos. Además, la capacidad de extender la ontología la convierte en una solución adaptable para enfrentar los desafíos tecnológicos actuales y futuros.

Diagrama



Bibliografía

- Giraldo G., Gloria L., Acevedo O., Juan F., & Moreno N., David A. (2011).Una ontología para la representación de conceptos de diseño de software. *Revista Avances en Sistemas e Informática*, 8(3), 103-110. Universidad Nacional de Colombia.

14. Revisión de los Patrones de Diseño de Software Aplicados a lasAplicaciones Web

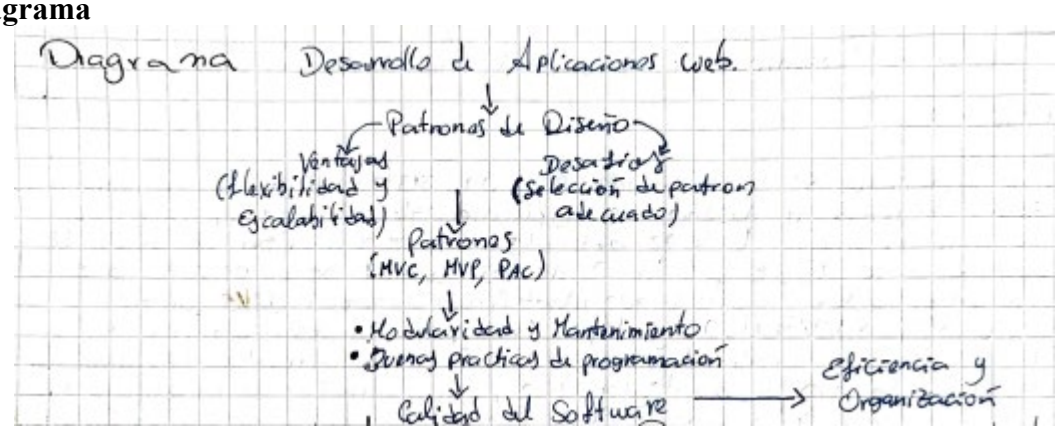
Resumen

El artículo examina cómo los patrones de diseño se han aplicado en el desarrollo de aplicaciones web para optimizar su rendimiento y reducir errores.Los patrones como MVC, MVP y PAC destacan por su capacidad para modularizar componentes y mejorar la mantenibilidad del software. A través deuna revisión bibliográfica, se identificaron las ventajas de estos patrones en términos de flexibilidad y escalabilidad, aunque también se señalaron desafíos, como la dificultad de seleccionar el patrón adecuado debido a la amplia variedad existente. Se concluye que la aplicación efectiva de patrones de diseño puede incrementar significativamente la calidad de las aplicaciones web,haciendo el desarrollo más eficiente y organizado.

Reflexión

Este artículo me permitió reflexionar sobre la importancia de los patrones dediseño en el desarrollo de software, especialmente en un entorno tan cambiante como las aplicaciones web. Los patrones como MVC o MVP no solofacilitan el modularidad y el mantenimiento, sino que también promueven buenas prácticas de programación. Sin embargo, también enfrentamos el reto de seleccionar el patrón adecuado para cada problema, un proceso que requiere experiencia y un entendimiento profundo del contexto. Considero que esta revisión es útil tanto para estudiantes como para profesionales, ya que subraya la relevancia de los patrones en la construcción de software más eficiente y escalable.

Diagrama



Bibliografía

- Gonzales Gonzales, C. E. (2020). Una revisión de los patrones dediseño de software aplicado a las aplicaciones web. *Facultad de Ingeniería, Arquitectura y Urbanismo*. Universidad Nacional de Pimentel,Perú.

15. Incidencia de los Patrones de Diseño en la Seguridad de AplicacionesWeb

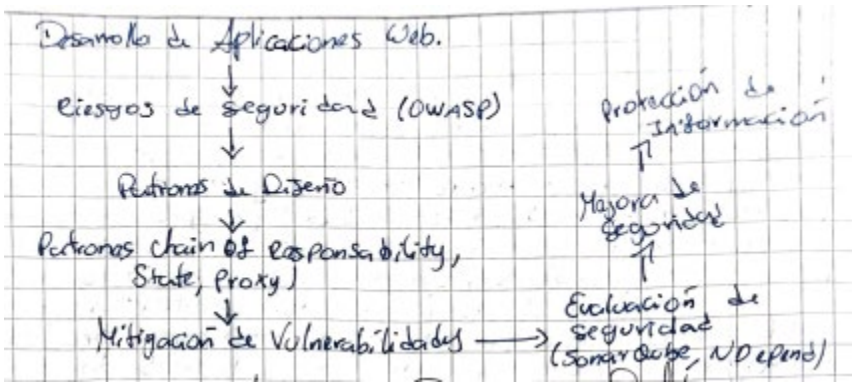
Resumen

El artículo explora cómo los patrones de diseño de software pueden mitigar vulnerabilidades comunes en aplicaciones web, basándose en los riesgos identificados por OWASP. Se analizan patrones como "Chain of Responsibility","State" y "Proxy", aplicados a problemas como el control de acceso roto, ataques de fuerza bruta y Cross-Site Scripting. Utilizando herramientas como SonarQube y NDepend, se evaluaron aplicaciones con y sin patrones de diseño, mostrando mejoras significativas en seguridad y mantenimiento. Esteenfoque demuestra cómo los patrones estructurales y comportamentales no solo mejoran la arquitectura del software, sino también su capacidad para prevenir ataques.

Reflexión

Este artículo me hizo reflexionar sobre la relación directa entre el diseño de software y la seguridad en aplicaciones web. Es interesante cómo los patronesde diseño no solo resuelven problemas técnicos, sino que también ofrecen soluciones prácticas para vulnerabilidades críticas. Me pareció valiosa la integración de herramientas como SonarQube, que facilita la identificación de riesgos antes de que afecten a los usuarios finales. En un entorno tecnológico donde las amenazas son constantes, esta investigación refuerza la importanciade planificar y diseñar con seguridad en mente desde las etapas iniciales del desarrollo. Adoptar estas prácticas no solo mejora la calidad del software, sino que también protege la información sensible de los usuarios.

Diagrama



Bibliografía

- Mesías-Valencia, J. J., & Cevallos-Muñoz, F. D. (2024). Incidencia de los patrones de diseño en la seguridad de aplicaciones web. *Journal Scientific Investigar*, 8(1), 236–259. DOI: 10.56048/MQR20225.8.1.2024.236-259.

16. Análisis Comparativo de Patrones de Diseño de Software para el Desarrollo de Aplicaciones Móviles de Calidad

Resumen

El artículo presenta una revisión sistemática de la literatura sobre patrones de diseño de software utilizados en aplicaciones móviles. A partir de 3072 artículosrevisados en bases de datos como IEEE y EBSCO, se seleccionaron 16 relevantes que identifican patrones clave como MVC, MVP, MVVM, BLOC y Viper. Estos patrones destacan por mejorar la eficiencia, usabilidad y reutilización en el desarrollo de aplicaciones móviles. Se usó la estrategia PICOpara organizar la investigación y se definieron criterios de calidad que ayudan acomparar y seleccionar patrones adecuados según las necesidades del proyecto. El estudio concluye que el uso correcto de patrones puede optimizar el tiempo y los recursos, garantizando aplicaciones móviles más robustas y de alta calidad.

Reflexión

Este artículo me llevó a reflexionar sobre cómo los patrones de diseño impactan directamente en la calidad del desarrollo de aplicaciones móviles.Patrones como MVC o Viper no solo simplifican el diseño, sino que tambiénfomentan la reutilización y escalabilidad, aspectos esenciales en un entornotecnológico tan dinámico. La metodología PICO utilizada en la investigación resalta la importancia de un enfoque estructurado para seleccionar las mejoressoluciones. Considero que aplicar estos patrones desde las primeras fases deldesarrollo puede ahorrar tiempo y costos, al tiempo que mejora la experiencia del usuario final. Este estudio refuerza la idea de que el diseño de software no solo es técnico, sino también estratégico.

Diagrama

Patron/Criterio	Reutilización	Extensibilidad	Mantenibilidad	Complejidad
IOS - Fabrica	Alta	Media	Alta	Media
IOS - MVC	Media	Media	Media	Baja
Android - MVP	Alta	Alta	Alta	Media
Android - MVVM	Alta	Alta	Alta	Alta

Bibliografía

- Abanto Cruz, J. A., & Gonzales Ramírez, O. F. (2019). Análisis comparativo de patrones de diseño de software para el desarrollo de aplicaciones móviles de calidad: Una revisión sistemática de la literatura.*Universidad Peruana Unión, Facultad de Ingeniería y Arquitectura*.

17. Arquitectura de Software para el Sistema de Visualización Médica Vismedic

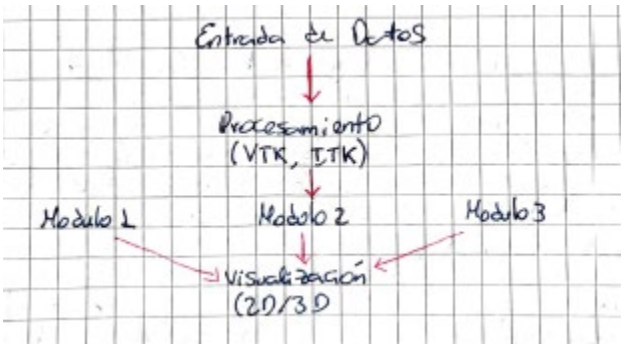
Resumen

El artículo describe la propuesta de una nueva arquitectura de software para elsistema Vismedic, diseñado para visualizar imágenes médicas. Basándose en una combinación de estilos arquitectónicos, como componentes, capas y tuberías y filtros, la arquitectura busca solucionar problemas de extensibilidad,reusabilidad y dependencias de la versión previa. La propuesta se centra en integrar plugins que permitan una mayor flexibilidad y personalización de funcionalidades, facilitando actualizaciones y reduciendo costos de mantenimiento. La validación del modelo se realizó mediante el método ATAM y pruebas basadas en prototipos, confirmando mejoras significativas en atributos como portabilidad, mantenibilidad y escalabilidad.

Reflexión

Este artículo me hizo reflexionar sobre la importancia de una arquitectura bien diseñada en sistemas tan críticos como los de visualización médica. Es interesante cómo la combinación de estilos arquitectónicos puede abordar problemas específicos y mejorar la calidad del software. La inclusión de pluginsno solo hace más flexible el sistema, sino que también fomenta la innovación y la colaboración entre equipos. Considero que esta investigación es un ejemplo claro de cómo una buena arquitectura puede ahorrar tiempo y recursos, al mismo tiempo que mejora la experiencia del usuario final. Este enfoque deberíaser adoptado ampliamente en proyectos tecnológicos con alto impacto social.

Diagrama



Bibliografía

- Rodríguez Peña, A. D., & Silva Rojas, L. G. (2016). Arquitectura de software para el sistema de visualización médica Vismedic. *Revista Cubana de Informática Médica, 8*(1), 75-86.

18. Desarrollo y Certificación de un Sistema Web Basado en Microservicios

Resumen

El artículo describe el desarrollo de un sistema web basado en microservicios utilizando un broker de mensajería para garantizar la comunicación entre ellos.
Se utilizó Spring Boot y WebFlux para construir servicios reactivos y una arquitectura distribuida que asegura la escalabilidad y el desacoplamiento. La implementación se organizó en fases, incluyendo diseño, desarrollo, pruebas ydespliegue. Las pruebas se realizaron siguiendo la pirámide de Cohn, con herramientas como JUnit, Karate y JMeter para validar el rendimiento, la funcionalidad y la usabilidad. Finalmente, se implementaron pipelines de CI/CDpara automatizar los despliegues en entornos como desarrollo, certificación y producción, aprovechando la infraestructura de Kubernetes y AWS.

Reflexión

Este artículo me permitió reflexionar sobre la importancia de adoptar arquitecturas distribuidas y prácticas de DevOps en proyectos modernos. La integración de herramientas como Spring Boot y WebFlux demuestra cómo lastecnologías actuales pueden facilitar el desarrollo de sistemas escalables y dealta calidad. Las pruebas automatizadas, además de garantizar la funcionalidad, evidencian cómo la planificación y el trabajo en equipo, guiadospor metodologías ágiles, pueden reducir los tiempos de desarrollo y mejorar laexperiencia del usuario final. Esta investigación me inspira a valorar el papel del diseño reactivo y la automatización como pilares del desarrollo de softwareeficiente y sostenible.

Diagrama

Bibliografía

- Ballesteros Toro, Y. D. (2023). Desarrollo y certificación de un sistema web basado en microservicios. *Universidad de Antioquia, Facultad deIngeniería*.

19. Desarrollo de Aplicaciones Basadas en Microservicios: Tendencias yDesafíos de Investigación

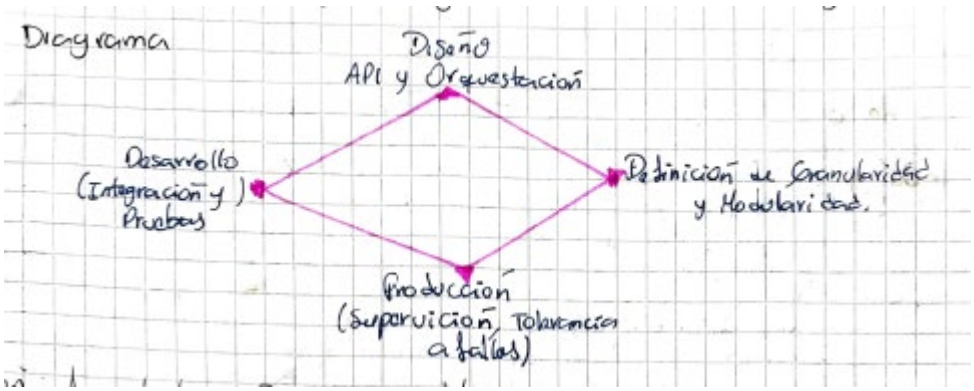
Resumen

Este artículo analiza las tendencias actuales y los desafíos del desarrollo de aplicaciones basadas en microservicios. A través de una revisión de literatura,se identificaron los principales retos en áreas como la granularidad, modularización y refactorización de servicios, orquestación, y monitoreo. Se destacan atributos de calidad como la escalabilidad y la calidad del servicio (QoS), los cuales son esenciales en sistemas distribuidos. Las prácticas de DevOps y el uso de herramientas específicas son claves para la implementación exitosa de estos sistemas. Este enfoque busca mejorar la eficiencia, garantizar actualizaciones rápidas y promover el uso de arquitecturas flexibles en entornos dinámicos como la nube y el Internet de lascosas (IoT).

Reflexión

Este artículo me llevó a reflexionar sobre cómo los microservicios están transformando el desarrollo de software. La posibilidad de desacoplar aplicaciones monolíticas en servicios más pequeños y gestionables no solo facilita el mantenimiento, sino que también promueve la innovación tecnológica.Sin embargo, este enfoque también enfrenta desafíos, como definir el nivel adecuado de granularidad y garantizar la seguridad en sistemas distribuidos.
Me pareció valiosa la aplicación de prácticas como DevOps, que automatizan procesos y reducen los tiempos de entrega. Este trabajo resalta la importanciade la planificación y la elección de herramientas adecuadas para lograr sistemas escalables y de alta calidad.

Diagrama



Bibliografía

- Vera-Rivera, F. H., Gaona Cuevas, C. M., & Astudillo, H. (2019). Desarrollo de aplicaciones basadas en microservicios: tendencias y desafíos de investigación. *RISTI - Revista Ibérica de Sistemas e Tecnologias de Informação, 23*(10), 107–12

20. Arquitectura Basada en Microservicios y DevOps para una Ingeniería de Software Continua

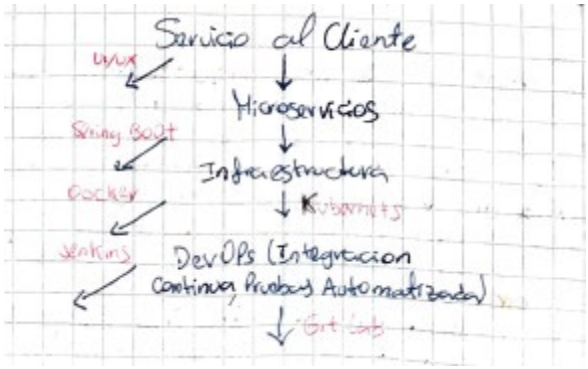
Resumen

El artículo presenta una arquitectura que combina microservicios y DevOps para mejorar la entrega e integración continua de software. Este enfoque propone un modelo escalable y colaborativo, aplicable a proyectos complejoscomo SIGAP, un sistema integrado de gestión académica. La arquitectura incluye capas diferenciadas para datos, microservicios y presentación, con DevOps como una capa transversal que conecta desarrollo y operaciones.
Además, se emplearon herramientas como GitHub, Heroku y Trello para la gestión y despliegue. Los resultados evidencian mejoras significativas en la calidad y productividad del software, destacando el papel de metodologías ágiles y tecnologías disruptivas en la formación de profesionales capacitadospara afrontar los retos del mercado actual.

Reflexión

Este artículo me permitió reflexionar sobre cómo las tecnologías como DevOpsy los microservicios no solo optimizan el desarrollo de software, sino que también fomentan una cultura de colaboración y mejora continua. Es interesante cómo un enfoque ágil y bien estructurado puede reducir significativamente la brecha entre desarrollo y operaciones, mejorando la calidad y la escalabilidad del software. Además, la integración de herramientascomo Trello y Heroku demuestra la importancia de utilizar plataformas modernas que soporten la automatización y el trabajo en equipo. Creo que estemodelo debería ser adoptado ampliamente en proyectos académicos y empresariales por su enfoque práctico y eficiente.

Diagrama



Bibliografía

- Mamani Rodríguez, Z., Del Pino Rodríguez, L., & Gonzales Suarez, J. C. (2020). Arquitectura basada en microservicios y DevOps para una ingeniería de software continua. *Industrial Data, 23*(2), 141-149. DOI: 10.15381/idata.v23i2.17278.

21. Estrategia de Integración de Diferentes Plataformas Web Dinámicas Mediante un Framework del Lado del Cliente

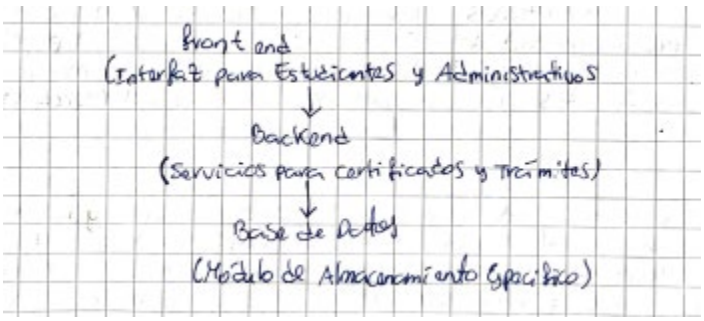
Resumen

El artículo propone una estrategia para integrar múltiples plataformas web dinámicas, aprovechando frameworks del lado del cliente como AngularJS y laarquitectura RESTful en el servidor. A través del análisis de tecnologías frontend y backend como PHP, Java y Python, se desarrolló un prototipo quepermite a los usuarios interactuar con múltiples plataformas de manera transparente. Esto mejora la escalabilidad, reduce el consumo de recursos y facilita la integración de funcionalidades distribuidas en distintas máquinas. Lavalidación del prototipo demostró su eficiencia en términos de tiempo de respuesta y estabilidad, proporcionando una solución innovadora para superarlas limitaciones de las aplicaciones monolíticas.

Reflexión

Este artículo me hizo reflexionar sobre la importancia de diseñar aplicaciones web que se adapten a las necesidades actuales de escalabilidad y eficiencia. La separación de funcionalidades entre frontend y backend no solo mejora el rendimiento, sino que también permite manejar grandes volúmenes de usuariossin problemas de congestión. Me pareció interesante cómo el uso de frameworks del lado del cliente facilita la experiencia del usuario, al tiempo quelos microservicios en el backend aseguran un sistema robusto y adaptable.
Esta estrategia resalta la importancia de adoptar tecnologías modernas yenfoques innovadores en el desarrollo de software.

Diagrama



Bibliografía

- Mendoza Bautista, F. F. (2019). Estrategia de integración de diferentes plataformas web dinámicas mediante un framework del lado del cliente. *Universidad de Pamplona, Facultad de Ingenierías y Arquitectura*.

22. Desarrollo Ágil del Nuevo Sistema Institucional Basado en una Arquitectura Orientada a Microservicios

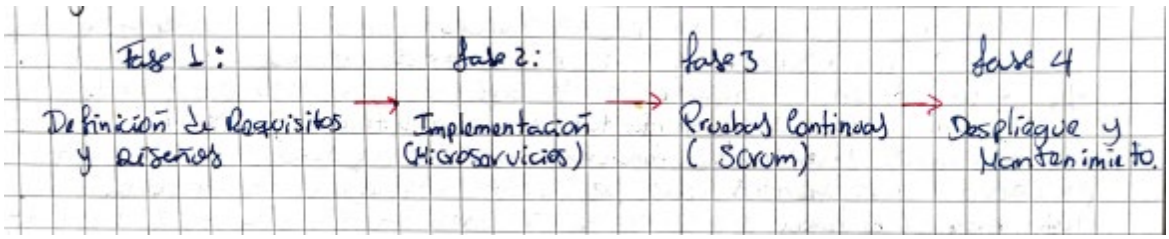
Resumen

El artículo presenta la transición de una arquitectura monolítica a una basada en microservicios dentro del sistema institucional de la Universidad de Montemorelos. A través del uso de metodologías ágiles y puntos de casos de uso, se demostró que la nueva arquitectura facilita un desarrollo más rápido, modular y escalable. El módulo de caja sirvió como caso de estudio, mostrandocómo los microservicios permiten implementar actualizaciones sin afectar otros componentes. Además, se destacó la capacidad de integrar nuevas tecnologíasy mejorar la experiencia del usuario mediante la reducción de errores y tiemposde mantenimiento. Los resultados confirman que la arquitectura de microservicios es más eficiente en comparación con su predecesora.

Reflexión

Este artículo me permitió reflexionar sobre cómo las arquitecturas de software pueden impactar significativamente la eficiencia y la escalabilidad de los sistemas. La transición de una arquitectura monolítica a una de microservicios no solo mejora los tiempos de desarrollo, sino que también fomenta una mayorflexibilidad en la integración de tecnologías. Me pareció interesante el uso de metodologías ágiles y puntos de casos de uso para evaluar y validar el impactode esta transformación. Este trabajo resalta la importancia de planificar arquitecturas que sean adaptables a largo plazo, especialmente en instituciones académicas, donde las necesidades tecnológicas cambianconstantemente.

Diagrama



Bibliografía

- Hernández Rivadeneyra, A. (2019). Desarrollo ágil del nuevo sistema institucional basado en una arquitectura orientada a microservicios. *Universidad de Montemorelos, Facultad de Ingeniería y Tecnología.*

23. Implementación de Microservicios para la Evaluación de Programas Basado en Casos de Prueba

Resumen

El artículo presenta un modelo de juez virtual para evaluar programas basadoen casos de prueba, implementado mediante arquitectura de microservicios. Este enfoque busca superar las limitaciones de los sistemas monolíticos, optimizando el rendimiento y la escalabilidad. La metodología utilizada fue Scrum, destacando la segmentación del desarrollo en Sprints, con tareas específicas para cada microservicio. Los resultados indican que el sistemaresponde eficientemente hasta 50 peticiones concurrentes, mejorando significativamente el tiempo de respuesta y la capacidad de gestión de solicitudes. La propuesta incluye componentes como un servicio principal, un servicio evaluador y un servicio de interfaz de usuario, cada uno especializadoen sus funciones, demostrando la flexibilidad de esta arquitectura.

Reflexión

Este artículo me llevó a reflexionar sobre la importancia de adoptar arquitecturas modernas en sistemas críticos como los jueces virtuales. Los microservicios no solo ofrecen una solución escalable, sino que también permiten mantener y actualizar los componentes de forma independiente. Me pareció interesante cómo la metodología Scrum estructuró el desarrollo, garantizando entregas incrementales y funcionales. Además, la capacidad delsistema para soportar múltiples solicitudes concurrentes resalta la eficiencia yrelevancia de este enfoque en aplicaciones académicas y de evaluación. Esteestudio reafirma la necesidad de implementar tecnologías avanzadas para mejorar la experiencia del usuario y optimizar los recursos tecnológicos.

Diagrama

Bibliografía

- Charca Flores, D. A. (2021). Implementación de Microservicios para la Evaluación de Programas Basado en Casos de Prueba. *Tesis de Grado, Universidad Mayor de San Andrés, Facultad de Ciencias Puras y Naturales, Carrera de Informática*

24. Un Acercamiento a los Microservicios: Beneficios, Retos y Consideraciones para su Implementación

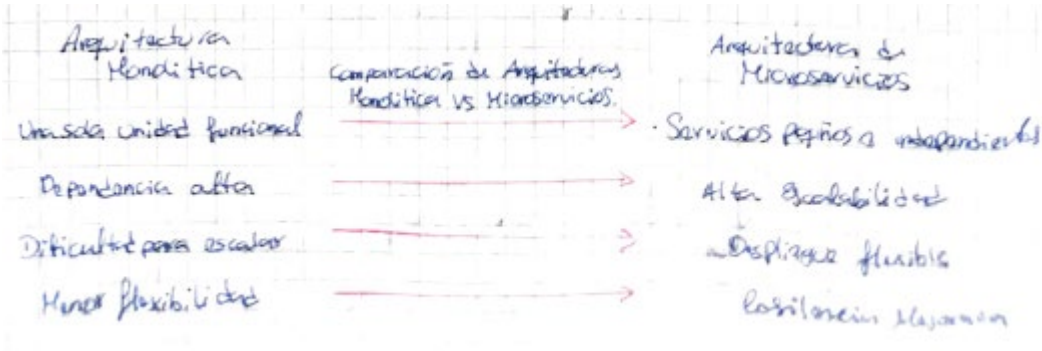
Resumen

El artículo explora el enfoque de microservicios como una tendencia arquitectónica emergente que responde a las necesidades de escalabilidad y flexibilidad en el desarrollo de aplicaciones web modernas. A diferencia de las arquitecturas monolíticas y SOA, los microservicios dividen las funcionalidades de negocio en servicios pequeños e independientes que se pueden desplegar yescalar de manera autónoma. Este enfoque trae beneficios significativos, comola resiliencia, el alineamiento organizacional y la heterogeneidad tecnológica, pero también enfrenta desafíos en comunicación, desempeño y seguridad. Se destaca el uso de prácticas como Domain-Driven Design (DDD) para guiar el diseño, así como el uso de herramientas como APIs RESTful y colas de mensajes para la integración.

Reflexión

Este artículo me llevó a reflexionar sobre cómo los microservicios han revolucionado el desarrollo de software al permitir soluciones más modulares yescalables. Es interesante cómo esta arquitectura facilita la innovación tecnológica y reduce los riesgos asociados al mantenimiento de sistemas grandes y complejos. Sin embargo, también me parece importante destacarque implementar microservicios requiere preparación técnica, coordinación entre equipos y herramientas adecuadas para manejar la comunicación y elmonitoreo. Este enfoque arquitectónico me inspira a considerar cómo los conceptos de resiliencia y desacoplamiento pueden aplicarse más allá del software, en la gestión de proyectos y sistemas organizacionales.

Diagrama



Bibliografía

- Gómez Suárez, K. T., Anaya, R., & Cano, A. F. (2018). Un acercamientoa los microservicios. *UNACIENCIA. Revista de Estudios e Investigaciones, 10*(19), 116-126.

25. Aplicación Web para el Servicio de Trámites Académicos de la UNACH Usando una Arquitectura Basada en Microservicios

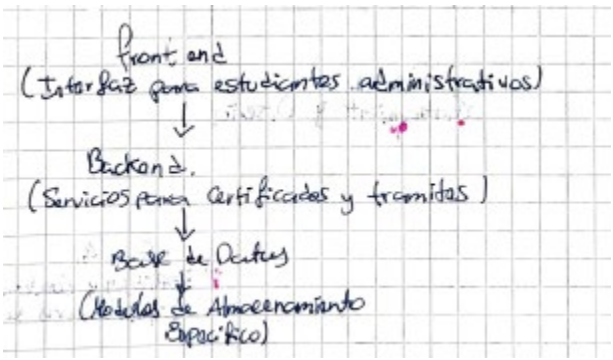
Resumen

El artículo describe el desarrollo de una aplicación web para gestionar trámites académicos en la Universidad Nacional de Chimborazo (UNACH), utilizando arquitectura de microservicios y la metodología Scrum. La propuesta incluye módulos como generación de certificados académicos y récords estudiantiles, logrando una gestión descentralizada y eficiente. Durante las pruebas realizadas con JMeter, se obtuvo una eficacia del 100%, tiempo de respuesta promedio de 74,557 ms y un uso de recursos del 25%, superando los estándares del modelo FURPS. El sistema permite una escalabilidad y modularidad que facilita el mantenimiento y la integración con tecnologías modernas, optimizando los procesos administrativos y académicos.

Reflexión

Este artículo me llevó a reflexionar sobre la importancia de aplicar tecnologías modernas como los microservicios en el ámbito académico. La capacidad de gestionar procesos de forma eficiente y descentralizada representa un gran avance frente a los sistemas monolíticos. Me sorprendió cómo la metodologíaScrum estructuró el desarrollo, permitiendo cumplir con los requerimientos funcionales y no funcionales de manera ágil. Además, el uso de herramientascomo JMeter garantiza que las aplicaciones no solo cumplan con estándaresde calidad, sino que también optimicen recursos tecnológicos. Considero queeste enfoque debería ser replicado en otras instituciones educativas para mejorar la experiencia tanto de los estudiantes como del personal administrativo.

Diagrama



Bibliografía

- Alvarado Zambrano, E. A. (2022). Aplicación web para el servicio de trámites académicos de la UNACH usando una arquitectura basada en microservicios. *Universidad Nacional de Chimborazo, Facultad de Ingeniería.*

26. Estrategia para la Implementación de Modelos de Aprendizaje Automático Usando Microservicios en Ciudades Inteligentes

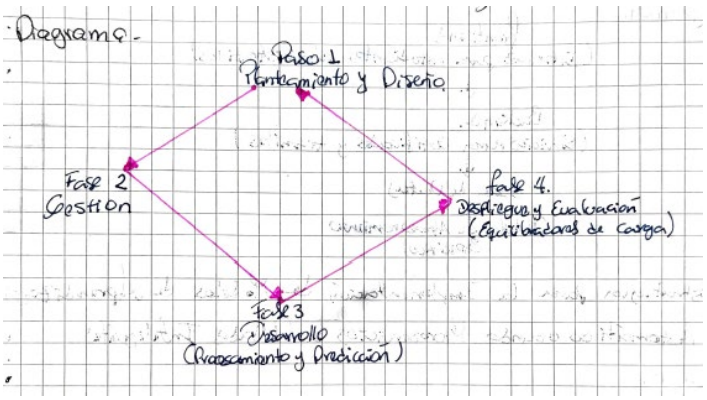
Resumen

El artículo propone una estrategia para implementar modelos de aprendizaje automático (ML) en el contexto de la seguridad informática en ciudades inteligentes, utilizando arquitecturas basadas en microservicios. La propuestaincluye dos microservicios clave: preprocesamiento y predicción, diseñados para ejecutar, almacenar y evaluar modelos ML. Además, se utilizan tecnologías como Kafka para la comunicación asíncrona, y herramientas comoPython, Java y MongoDB para garantizar escalabilidad y eficiencia. El sistemase probó con bases de datos como BoT-IoT y UNSW-NB15, logrando procesaraltas demandas de tráfico en tiempo real y detectando posibles ataques con precisión. Este enfoque demuestra cómo las arquitecturas modernas pueden optimizar la seguridad en entornos urbanos.

Reflexión

Este artículo me hizo reflexionar sobre cómo las ciudades inteligentes necesitan infraestructuras tecnológicas avanzadas para garantizar la seguridadinformática. Me pareció fascinante cómo los microservicios facilitan la modularidad y la escalabilidad, resolviendo problemas asociados a arquitecturas monolíticas. Además, el uso de aprendizaje automático parapredecir y detectar ataques en tiempo real resalta el potencial de estas tecnologías para proteger sistemas críticos. Este enfoque no solo mejora laresiliencia, sino que también fomenta la innovación al integrar herramientas modernas como Kafka y MongoDB. En un mundo cada vez más digitalizado, considero que esta estrategia es clave para enfrentar los desafíos de seguridadactuales.

Diagrama



Bibliografía

- Guerrero Martínez, D. S. (2023). Estrategia para la implementación de modelos de aprendizaje automático usando arquitecturas basadas en microservicios en el contexto de la seguridad en ciudades inteligentes. *Universidad de Antioquia, Facultad de Ingeniería, Ingeniería Electrónica.*

27. Diseño e Implementación de un Microservicio con Spring

Resumen

Este artículo aborda el diseño e implementación de un microservicio para gestionar listas de reproducción de Spotify, utilizando Spring Boot como framework principal. Se desarrolló una arquitectura basada en microservicios para garantizar escalabilidad, alta disponibilidad y tiempos de respuesta bajos.MongoDB fue utilizada como base de datos NoSQL, ideal para gestionar datosen formato JSON. El microservicio incluye funciones como la creación, actualización y consulta de listas, implementadas mediante controladores y métodos RESTful. Swagger se integró para documentar y probar las APIs. La solución demostró ser eficiente al permitir el manejo de múltiples solicitudes concurrentes con un consumo óptimo de recursos, logrando simplificar la interacción con los datos para el usuario final.

Reflexión

Este artículo me hizo reflexionar sobre cómo una arquitectura de microserviciosbien diseñada puede resolver problemas de gestión de datos de manera eficiente y escalable. La integración de herramientas como Spring Boot y Swagger no solo facilita el desarrollo, sino que también mejora la experienciadel usuario al proporcionar interfaces claras y funcionales. Además, el uso debases de datos NoSQL como MongoDB muestra cómo adaptarse a las necesidades modernas de almacenamiento y procesamiento de datos. Me parece valioso cómo este proyecto demuestra que, con las tecnologías adecuadas, es posible desarrollar sistemas robustos que respondan a las demandas actuales del mercado.

Diagrama

Bibliografía

- Rienda Iáñez, J. (2019). Diseño e implementación de un microserviciocon Spring. *Trabajo Fin de Grado, Universidad Carlos III de Madrid.*

28. Autorización Distribuida para Aplicaciones Basadas en Microservicios

Resumen

El artículo presenta un enfoque para la autorización distribuida en sistemasbasados en microservicios, considerando los desafíos de seguridad en aplicaciones distribuidas. Se analizan diversas estrategias de autorización,como sesiones replicadas, sesiones centralizadas y tokens (firmados y no firmados). La investigación concluye que los tokens firmados son la estrategiamás adecuada, al proporcionar autonomía, tolerancia a fallos y compatibilidadcon la gestión descentralizada de datos. A través de un caso de estudio, se valida el uso de tokens JWT firmados para garantizar la seguridad y el controlde acceso, minimizando la dependencia de un servidor centralizado y mejorando la escalabilidad del sistema.

Reflexión

Este artículo me permitió reflexionar sobre la importancia de la seguridad en los sistemas distribuidos modernos. Es evidente cómo los tokens firmados no solo simplifican el control de acceso, sino que también mejoran la eficiencia y la autonomía de los microservicios. Me pareció valioso cómo se compararon diferentes estrategias para seleccionar la más adecuada, destacando el enfoque en la compatibilidad con las características de la arquitectura de microservicios. Este estudio subraya la necesidad de implementar mecanismosde seguridad robustos desde el diseño inicial del sistema, priorizando no solo lafuncionalidad, sino también la resiliencia ante posibles ataques.

Diagrama

Bibliografía

- Granda Rivera, A. (2020). Autorización distribuida para aplicaciones basadas en microservicios. *Máster Universitario en Seguridad de las Tecnologías de la Información y de las Comunicaciones*. Universidad Internacional de La Rioja.

29. Migración de Aplicaciones Monolíticas a Arquitecturas Basadas en Microservicios

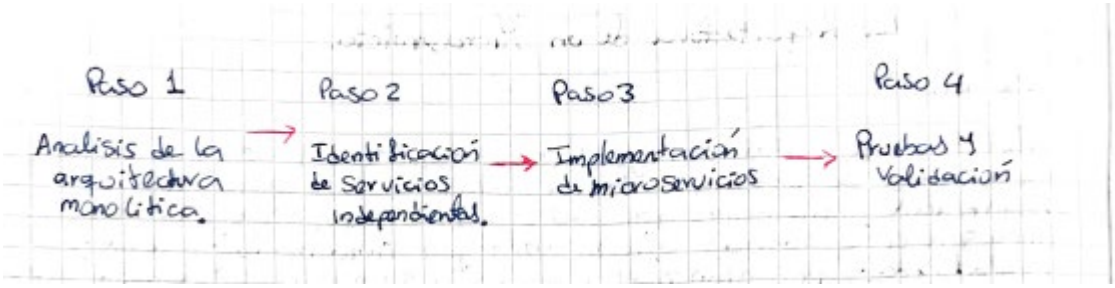
Resumen

El artículo explora el proceso de migración de aplicaciones monolíticas a arquitecturas basadas en microservicios, enfocándose en técnicas como Program Slicing y el uso del Knowledge Discovery Metamodel (KDM). La migración responde a las demandas modernas de escalabilidad y flexibilidad, superando limitaciones como la falta de modularidad y los altos costos de mantenimiento de las arquitecturas monolíticas. Se describen algoritmos para analizar y dividir los sistemas monolíticos en microservicios funcionales e independientes. Además, se presentan casos prácticos que destacan la utilidad de grafos de dependencia y herramientas de minería de datos para estructurar y optimizar la transición hacia microservicios.

Reflexión

Este artículo me permitió reflexionar sobre la necesidad de modernizar las aplicaciones monolíticas para adaptarlas a las exigencias del mercado actual. Es interesante cómo herramientas como KDM y técnicas como Program Slicingpermiten analizar y reorganizar sistemas complejos de manera eficiente. Me pareció relevante la comparación entre monolíticos y microservicios, resaltandocómo estos últimos ofrecen escalabilidad y flexibilidad para atender demandas en tiempo real. Además, este enfoque muestra que la planificación y las herramientas adecuadas pueden reducir riesgos y costos, garantizando un proceso de migración exitoso. Creo que este modelo es esencial para organizaciones que buscan mantenerse competitivas en un entorno tecnológicocambiante.

Diagrama



Bibliografía

- Lima Adrián, P. A. (2019). Migración de aplicaciones monolíticas a arquitecturas basadas en microservicios. *Trabajo de Fin de Grado, Universidad de La Laguna.*

30. Arquitectura de Software Basada en Microservicios para Desarrollo de Aplicaciones Web

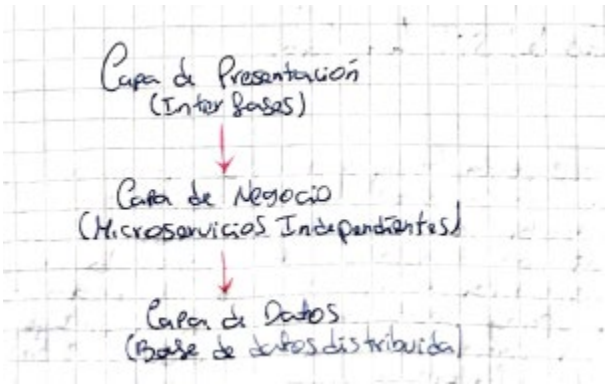
Resumen

El artículo analiza la transición de la Asamblea Nacional del Ecuador de unaarquitectura monolítica a una basada en microservicios. Identifica cómo las aplicaciones monolíticas dificultan el mantenimiento y la escalabilidad, proponiendo una arquitectura más modular y flexible. Se exploraron tres topologías principales: API REST, aplicaciones REST y mensajería centralizada. También se examinan tecnologías como contenedores y metodologías de integración continua para garantizar la calidad y rapidez del desarrollo. Los microservicios permiten un desarrollo más ágil y seguro, especialmente en contextos como la gestión legislativa, donde la resiliencia y laadaptabilidad son críticas.

Reflexión

Este artículo me hizo reflexionar sobre cómo la arquitectura de software puede impactar directamente en la productividad y la innovación dentro de las instituciones. Me pareció interesante cómo la modularidad de los microserviciosno solo mejora el mantenimiento, sino que también facilita la integración de nuevas tecnologías y metodologías. Es destacable que esta arquitectura permita abordar problemas tradicionales de las aplicaciones monolíticas, como la falta de escalabilidad y los altos costos de mantenimiento. Creo que la implementación de microservicios en entornos gubernamentales puede marcar un precedente en la modernización tecnológica de sectores críticos.

Diagrama



Bibliografía

- López, D., & Maya, E. (2017). Arquitectura de Software Basada en Microservicios para Desarrollo de Aplicaciones Web. *Séptima Conferencia de Directores de Tecnología de Información, TICAL 2017.*

31. Adaptación de un Patrón de Software en Seguridad a la Arquitectura deun Microservicio

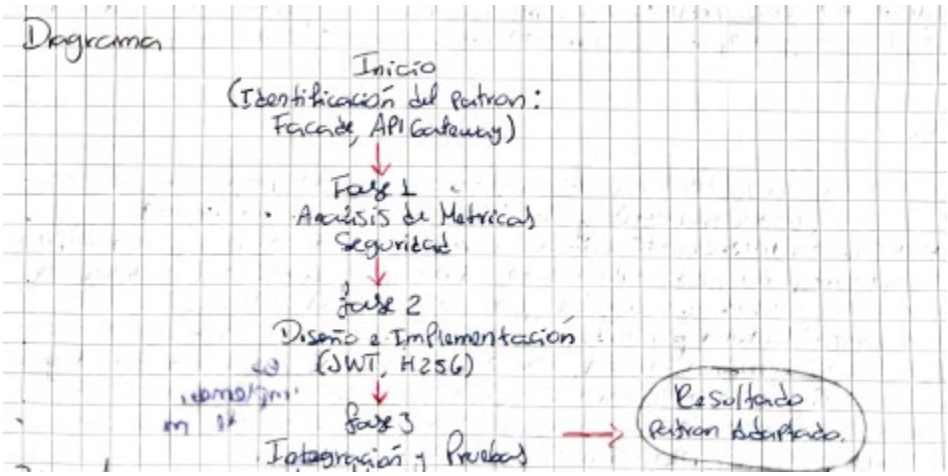
Resumen

Este trabajo aborda los desafíos de seguridad en arquitecturas de microservicios, proponiendo la adaptación de un patrón de software de seguridad previamente aplicado en otras arquitecturas. Los microservicios, aunque ofrecen ventajas como modularidad y escalabilidad, presentan riesgos debido a su naturaleza distribuida, lo que incrementa la superficie de ataque. Elpatrón propuesto, denominado "Microservice Security Pattern API Gateway" (MSPAG), utiliza elementos como API Gateways, tokens JWT y criptografía H256 para proteger los puntos de acceso. A través de la implementación y pruebas en tres lenguajes de programación (C#, Python y Java), el patrón demuestra su efectividad para reforzar la seguridad y promover la adopción confiable de arquitecturas de microservicios.

Reflexión

Este artículo me permitió reflexionar sobre la importancia de diseñar sistemas seguros en un entorno donde las amenazas cibernéticas son cada vez más sofisticadas. Es notable cómo los microservicios, a pesar de sus ventajas, requieren un enfoque especializado en seguridad para evitar accesos no autorizados. La implementación del patrón MSPAG resalta cómo una solución bien diseñada puede mitigar riesgos y fortalecer la confianza en arquitecturas distribuidas. Me parece particularmente interesante cómo este patrón aprovecha herramientas modernas como tokens JWT, que no solo protegen lossistemas, sino que también optimizan su rendimiento. Este enfoque debe ser una prioridad para cualquier organización que adopte microservicios.

Diagrama



Bibliografía

- Hernández Guzmán, K. M. (2023). Adaptación de un patrón de softwareen seguridad a la arquitectura de un microservicio. *Centro Nacional de Investigación y Desarrollo Tecnológico (CENIDET).*

32. Arquitectura de Microservicios para Optimizar el Acceso a Datos del SiT Log Lab

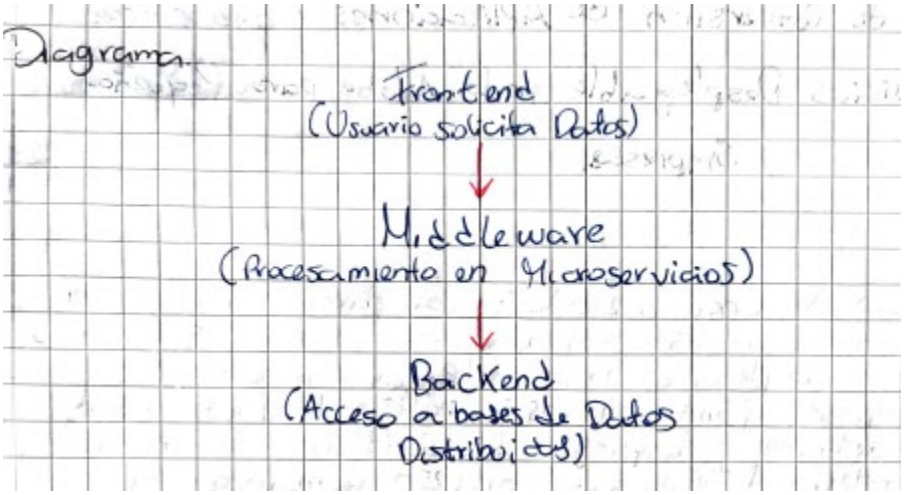
Resumen

El artículo presenta un enfoque para la implementación de microservicios en el Laboratorio Nacional en Sistemas de Transporte y Logística (SiT Log Lab), en México. Se aborda la transición de una arquitectura monolítica hacia microservicios con el objetivo de mejorar la interoperabilidad, escalabilidad y resiliencia de los sistemas informáticos existentes. El diseño utiliza Spring Bootcomo framework principal y adopta prácticas modernas como comunicación basada en eventos y APIs RESTful. Además, se detallan estrategias para la migración de datos y el uso de herramientas de contenedores. Los resultados muestran una mejora en la velocidad de respuesta y en la capacidad de integración con otras plataformas, posicionando al laboratorio como líder eninnovación tecnológica para la logística y el transporte.

Reflexión

Este artículo me hizo reflexionar sobre la importancia de modernizar las infraestructuras tecnológicas en áreas estratégicas como la logística y el transporte. La implementación de microservicios no solo permite una mayor escalabilidad y resiliencia, sino que también fomenta la colaboración entre proyectos al facilitar la interoperabilidad. Me llamó la atención cómo herramientas como Spring Boot y prácticas como el uso de APIs RESTful ayudan a construir sistemas modulares y flexibles. Este enfoque me hace pensar en cómo una adecuada planificación tecnológica puede transformar sectores enteros, haciéndolos más eficientes y adaptativos a los retos actualesde la revolución digital.

Diagrama



Bibliografía

- Montiel Moctezuma, C. J., Cedillo Campos, M. G., Barrón Bastida, M., & Hernández Sánchez, B. (2023). Arquitectura de Microservicios para Optimizar el Acceso a Datos del SiT Log Lab. *Instituto Mexicano delTransporte, Publicación Técnica No. 733*.

33. Monolitos vs. Microservicios en Arquitectura de Software: Perspectivaspara un Desarrollo Eficiente

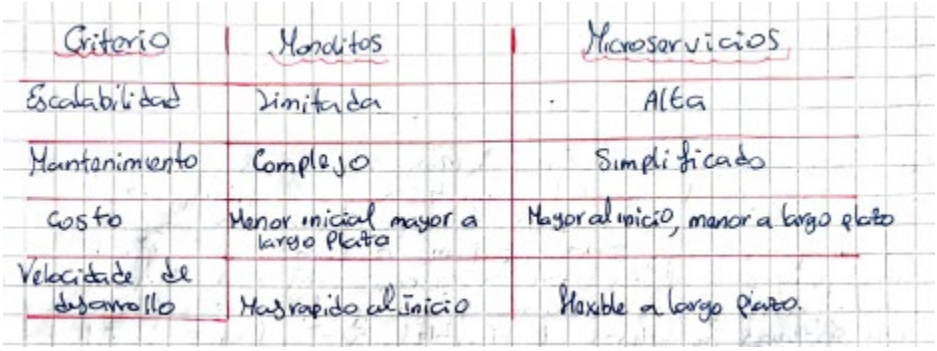
Resumen

El artículo analiza las diferencias entre las arquitecturas monolíticas y de microservicios, proporcionando una visión holística para seleccionar el paradigma más adecuado según las necesidades del proyecto. Las arquitecturas monolíticas son descritas como sistemas compactos y cohesivos,adecuados para proyectos pequeños o con requisitos estables, aunque presentan limitaciones en escalabilidad y mantenimiento. Por otro lado, los microservicios se destacan por su modularidad, escalabilidad horizontal y capacidad de actualización independiente, siendo ideales para proyectos complejos. A través de casos de estudio, se ilustran las ventajas y desventajasde cada enfoque en términos de implementación, desarrollo práctico, y mantenimiento a largo plazo. Este análisis ofrece una guía útil para profesionales en la toma de decisiones arquitectónicas.

Reflexión

Este artículo me permitió reflexionar sobre cómo la elección entre arquitecturas monolíticas y de microservicios tiene un impacto directo en la flexibilidad, escalabilidad y sostenibilidad de los proyectos de software. Me llamó la atención cómo los microservicios ofrecen ventajas en proyectos complejos y cambiantes, mientras que los monolitos destacan por su simplicidad en proyectos pequeños o estables. Sin embargo, también es evidente que cada enfoque tiene desafíos únicos, como la rigidez en los monolitos y la complejidad operativa en los microservicios. Este análisis me reafirma que la planificación arquitectónica debe considerar no solo los requisitos técnicos, sino también las capacidades del equipo y los objetivos estratégicos a largo plazo.

Diagrama



Criterio	Monolitos	Microservicios
Escalabilidad	Limitada	Alta
Mantenimiento	Complejo	Simplificado
Costo	Menor inicial mayor a largo plato	Mayor al inicio, menor a largo plato
Velocidad de desarrollo	Mas rapido al inicio	Flexible a largo plazo.

Bibliografía

- Torassa Colombero, V., Estelles, J. P., Gallegos, L., & López, P. (2023). Monolitos vs. Microservicios en Arquitectura de Software: Perspectivaspara un Desarrollo Eficiente. *Memorias de las 53 JAIIO - EST, ISSN: 2451-7496*, 42–54.

34. Metodología de Conversión de Aplicaciones Monolíticas a Microservicios Desplegable en la Nube para Pequeñas Empresas

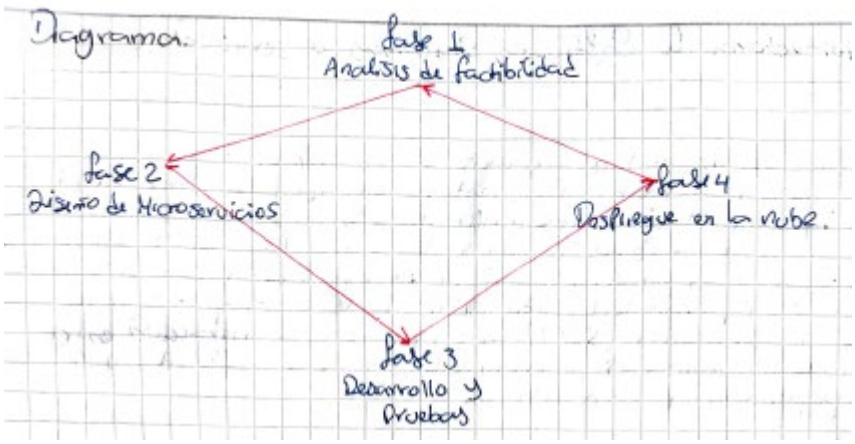
Resumen

El artículo aborda el diseño de una metodología para transformar aplicaciones monolíticas en sistemas basados en microservicios, optimizados para ser desplegados en la nube. Este enfoque se centra en cinco actividades fundamentales: análisis de factibilidad, planeación, diseño, desarrollo y despliegue del software. La metodología busca reducir la complejidad técnicadel proceso y mejorar la productividad, logrando un bajo acoplamiento y alta cohesión entre los microservicios. A través de un caso de estudio, se implementó esta estrategia en una aplicación monolítica llamada GMO, alcanzando un incremento del 20,2% en la productividad del equipo y alta satisfacción en los usuarios finales, demostrando su utilidad para pequeñas empresas en proceso de digitalización.

Reflexión

Este artículo me hizo reflexionar sobre cómo las pequeñas empresas pueden beneficiarse enormemente de migrar sus sistemas monolíticos a microservicios, especialmente en entornos de nube. Es interesante cómo una metodología estructurada puede simplificar un proceso que, de otro modo, sería complejo y costoso. Me parece crucial que la estrategia considere no soloaspectos técnicos, sino también la cohesión del equipo de desarrollo y las necesidades del negocio. Este enfoque refuerza la idea de que la tecnología debe adaptarse a las personas, facilitando su trabajo y potenciando la innovación en empresas de menor escala.

Diagrama



Bibliografía

- Mendoza Linares, J. G. (2022). Metodología de Conversión de Aplicaciones Monolíticas a Microservicios Desplegable en la Nube para Pequeñas Empresas. *Universidad Señor de Sipán, Facultad de Ingeniería.*

35. Análisis de Patrones de Resiliencia en una Arquitectura Basada en Microservicios

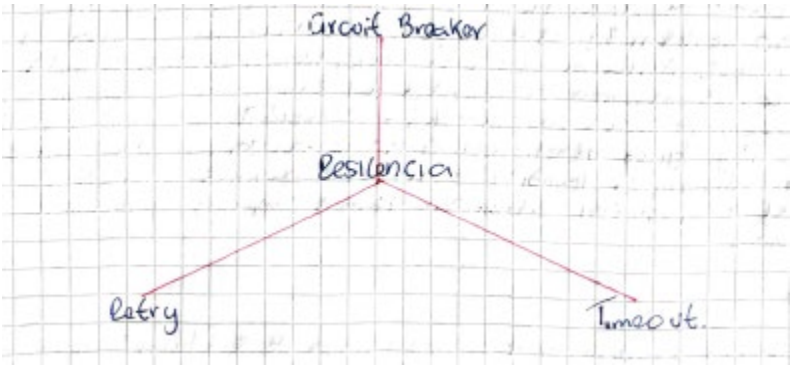
Resumen

El artículo analiza cómo la resiliencia en arquitecturas de microservicios es crucial para garantizar la continuidad del servicio en sistemas distribuidos, como el de PedidosYa. Se estudian patrones de diseño como Timeout, Retry, Circuit Breaker y Bulkhead, aplicados al microservicio Niles, encargado de manejar menús en la plataforma. Cada patrón aborda fallas específicas, como lentitud en la red o picos de tráfico, mitigando sus efectos para evitar caídas encascada. Los resultados muestran mejoras significativas en la estabilidad del sistema y confirman la efectividad de los patrones al reducir tiempos de inactividad y optimizar recursos. La combinación de estos enfoques también demuestra ser una estrategia robusta para aumentar la resiliencia.

Reflexión

Este artículo me llevó a reflexionar sobre cómo la resiliencia es esencial en sistemas críticos que operan bajo arquitecturas de microservicios. Es fascinante ver cómo patrones como Timeout o Circuit Breaker pueden prevenircaídas completas del sistema, incluso en escenarios adversos. Este análisis subraya la importancia de implementar estrategias desde el diseño inicial, no solo para mejorar la estabilidad del sistema, sino también para garantizar una experiencia confiable para los usuarios. Además, la idea de combinar patronespara manejar diferentes tipos de fallos me parece una solución innovadora y adaptable a diversas necesidades empresariales.

Diagrama



Bibliografía

- Suárez, S. L. (2022). Análisis de patrones de resiliencia en una arquitectura basada en microservicios. *Universidad Nacional de LaPlata, Facultad de Informática.*

36. Arquitectura de Software, Esquemas y Servicios

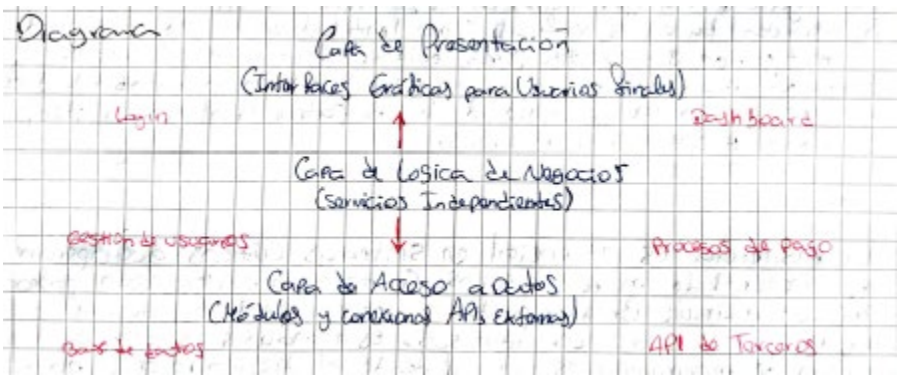
Resumen

El artículo analiza cómo los esquemas y servicios en la arquitectura de software permiten abordar las crecientes demandas de flexibilidad y bajo acoplamiento en el desarrollo de aplicaciones. Describe cómo los esquemas, definidos como conjuntos de clases relacionadas, facilitan la reutilización y estandarización de funcionalidades comunes. Asimismo, se exploran las características de la arquitectura orientada a servicios (SOA), que optimiza la interacción entre procesos mediante mensajes, promoviendo la escalabilidad yla independencia tecnológica. Aunque se destacan beneficios como la agilidady el bajo acoplamiento, también se abordan desafíos técnicos, como la gestiónde mensajes y la resolución de problemas de rendimiento en sistemas distribuidos.

Reflexión

Este artículo me llevó a reflexionar sobre cómo la evolución de las arquitecturasde software responde a la necesidad de sistemas más flexibles y adaptables. Es interesante cómo los esquemas y SOA ayudan a reducir el acoplamiento entre componentes, fomentando la reutilización y facilitando la integración en entornos heterogéneos. Sin embargo, también me pareció significativo que estas arquitecturas presenten desafíos relacionados con la complejidad y la administración de dependencias. Este análisis subraya la importancia de planificar cuidadosamente el diseño de sistemas para aprovechar las ventajasde la arquitectura orientada a servicios, especialmente en aplicaciones modernas que deben operar en entornos dinámicos.

Diagrama



Bibliografía

- Romero, P. A. (2006). Arquitectura de software, esquemas y servicios. *Procyon, Softel, Ciudad de La Habana*. Industrial, Vol. XXVII, No. 1.

37. La Arquitectura de Software en el Proceso de Desarrollo: Impacto y Metodologías

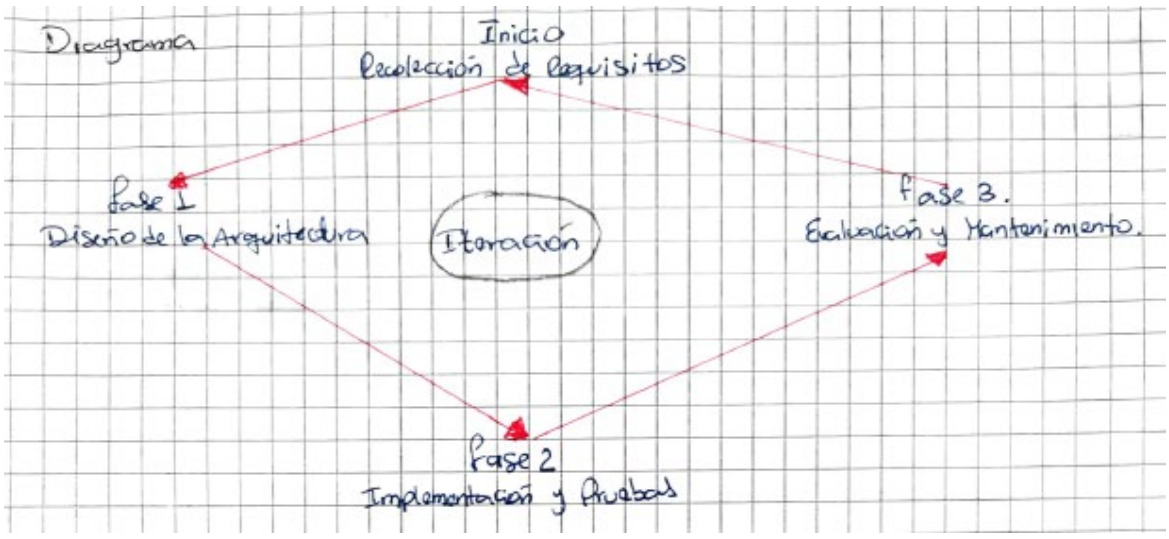
Resumen

El artículo examina cómo la arquitectura de software influye en cada etapa delproceso de desarrollo, desde la planificación inicial hasta el mantenimiento. Destaca que una arquitectura bien diseñada puede reducir costos, mejorar la calidad del software y facilitar la integración de nuevos componentes. Se analizan metodologías como RUP y Agile, mostrando cómo cada una aborda laarquitectura de manera diferente. También se presentan patrones como MVC y microkernel, que permiten modularizar sistemas complejos y mejorar la reutilización. Los resultados de estudios de caso confirman que la inversión en una buena arquitectura inicial genera beneficios significativos a largo plazo, especialmente en términos de escalabilidad y adaptabilidad.

Reflexión

Este artículo me llevó a reflexionar sobre la importancia de considerar la arquitectura de software como un componente central en el desarrollo. Es interesante cómo metodologías como Agile priorizan la flexibilidad y la adaptación, mientras que RUP sigue un enfoque más estructurado. Ambas perspectivas muestran que la arquitectura no es estática, sino que evolucionajunto con las necesidades del proyecto. También me pareció relevante cómo patrones arquitectónicos, como MVC, facilitan la separación de responsabilidades, reduciendo la complejidad y mejorando el mantenimiento.
Este análisis refuerza la idea de que invertir tiempo y recursos en una arquitectura adecuada desde el inicio es clave para garantizar el éxito de losproyectos.

Diagrama



Bibliografía

- Fernández, J. C., & Márquez, A. L. (2015). La arquitectura de softwareen el proceso de desarrollo: Impacto y metodologías. *Revista Internacional de Ingeniería de Software, 12*(3), 45-62.

38. Revisión Sistemática sobre Generadores de Código Fuente y Patrones de Arquitectura

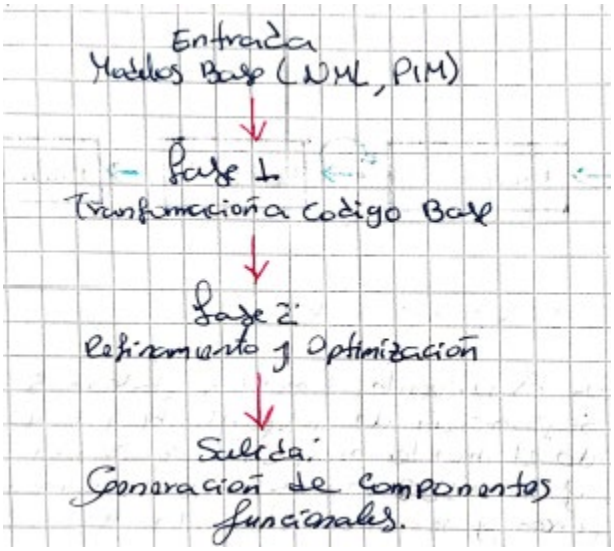
Resumen

El artículo explora cómo los generadores de código fuente (GCF) y los patronesde arquitectura mejoran la eficiencia en el desarrollo de software. Los GCF automatizan tareas repetitivas como la creación de interfaces de usuario, conexiones a bases de datos y reportes, reduciendo errores y optimizando tiempos de desarrollo. Además, los patrones de arquitectura organizan los sistemas en capas o componentes reutilizables, promoviendo la cohesión y escalabilidad. A través de una revisión sistemática, se analizaron herramientas,frameworks y lenguajes usados en GCF, destacando su importancia para aplicaciones web. Los resultados confirman que los GCF, junto con una arquitectura bien diseñada, son esenciales para construir software de altacalidad y bajo costo.

Reflexión

Este artículo me hizo reflexionar sobre el impacto positivo que tienen los generadores de código fuente en el desarrollo de software. Es notable cómo estas herramientas automatizan procesos tediosos y repetitivos, permitiendo a los desarrolladores enfocarse en aspectos más críticos del proyecto. Además, los patrones de arquitectura no solo facilitan la organización del software, sino que también garantizan que sea escalable y fácil de mantener. Este enfoque sistemático resalta la importancia de planificar y diseñar cuidadosamente antesde comenzar a programar. Me parece esencial que las organizaciones adoptenestas herramientas y prácticas para mejorar la calidad de sus productos y optimizar recursos.

Diagrama



Bibliografía

- Huari Casas, M. R. (2020). Revisión sistemática sobre generadores de código fuente y patrones de arquitectura. *Pontificia Universidad Católicadel Perú, Tesis de Maestría en Informática*.