DAVID MAURICIO FLOREZ QUINTERO



Soy un estudiante de Tecnólogo en Análisis y Desarrollo de Software, con una profunda pasión por la tecnología y la programación. Mi objetivo es aplicar mis conocimientos para ofrecer soluciones innovadoras, óptimas y escalables a los desafíos que surgen en el campo del desarrollo de software. Me esfuerzo por seguir las mejores prácticas de programación, garantizando así la alta calidad en cada proyecto que emprendo.

Mis habilidades en liderazgo, comunicación y proactividad me han permitido no solo crecer como profesional, sino también enfrentar y superar obstáculos con eficacia. Estoy comprometido con la mejora continua, convencido de que tanto las personas como los sistemas siempre tienen potencial para evolucionar y perfeccionarse. Con una mentalidad orientada a la excelencia, busco contribuir de manera significativa al desarrollo de soluciones que no solo satisfagan las necesidades actuales, sino que también sean preparadas para el futuro.

1. Revisión de elementos conceptuales para la representación de las arquitecturas de referencias de software

El artículo trata la representación y conceptualización de arquitecturas de referencias de softwares ARS evalúa distintos lenguajes de descripción arquitectónicas ADL utilizados para modelar.

1. Lenguajes de Patrones de Arquitectura de Software: Una Aproximación Al Estado del Arte

Mostrar el estado de una arquitectura de software llamada lenguaje de patrones desde sus orígenes los avances conceptuales y su aplicación en la construcción de arquitecturas de software en diferentes dominios de aplicación

1. Especificación de la arquitectura de software.

La investigación examina los componentes de una arquitectura de software y su interrelación mediante patrones arquitecturales. Destaca cómo los avances tecnológicos y el crecimiento de plataformas móviles y web han transformado el diseño de infraestructuras para manejar tareas complejas, tiempos de entrega reducidos, procesamiento de datos y seguridad. El artículo ofrece una visión general de los componentes de la arquitectura de software actual, sus interpretaciones y su contexto, abordando cómo los patrones arquitecturales ayudan a organizar y estructurar estos elementos para soportar el crecimiento y la magnitud de las ideas.

1. Ingeniería De Software Y Computación En La Nube.

Artículo que habla sobre la computación en la nube explica los servicios con infraestructura la infraestructura como servicio almacenamiento capacidad de procesamiento y transferencia en la red paas plataforma como servicio en esta capa del usuario tiene acceso a plataformas del desarrollo.

1. Distinguiendo entre SaaS y SOA.

Artículo donde nos explica la diferencia entre estos términos saas es un modelo de entrega de software mientras que el segundo es un modelo para la construcción de un software eso es una arquitectura orientada en servicios colección de servicios que interactúan entre sí a través de interfaces estándares.

1. Paradigmas en la construcción de software.

El artículo habla sobre la importancia de la modelación en la construcción de software confiable y de calidad. Se analizan varios paradigmas de construcción, incluyendo imperativo, funcional, de modelamiento de datos, declarativo y orientado a objetos.

1. Atributos de Calidad y Arquitectura del Software.

Documentar una arquitectura de software implica usar vistas específicas: Módulos (estructura estática), Componentes y conectores (estructura dinámica), y Allocation (relación con el entorno externo). Cada vista puede adoptar diferentes estilos arquitectónicos que afectan los atributos de calidad del sistema.

1. Integración de arquitectura de software en el ciclo de vida de las metodologías ágiles.

Las metodologías Ágiles (MA) priorizan el trabajo en equipo, adaptabilidad y colaboración, adaptándose a requisitos cambiantes para entregar beneficios rápidamente. En contraste, la Arquitectura de Software (AR) se basa en decisiones tempranas que afectan significativamente el desarrollo del sistema, lo que ha llevado a la percepción de que MA y AR son incompatibles. Sin embargo, en los últimos cinco años ha emergido el concepto de "Arquitectura Ágil" (AA), que busca integrar MA y AR al enfocarse en los "Requisitos Significantes para la Arquitectura" (RSA). Esta investigación explora cómo combinar efectivamente AR con MA, con énfasis en la captura y gestión de los RSA.

1. Una Arquitectura para una Herramienta de Patrones de Diseño.

Los patrones de diseño son cruciales para construir software orientado a objetos, y los entornos de programación deben ofrecer herramientas que faciliten su uso. Para integrar estos patrones de manera eficiente, se necesita una arquitectura flexible. Este trabajo presenta una arquitectura basada en patrones diseñada para facilitar el manejo de patrones de diseño y describe una herramienta desarrollada para evaluar esta arquitectura.

1. Arquitectura De Software Académica Para La Comprensión Del Desarrollo De Software En Capas

Es una arquitectura donde las capas representan un aspecto lógico definido colaboradores hacia el sistema punto ejemplo de capas relacionadas con el usuario reglas lógicas persistencia de datos capas GUI capa controladora capa lógica capa orm

1. Marco de Trabajo para Seleccionar un Patrón Arquitectónico en el Desarrollo de Software.

El texto aborda problemas comunes en el diseño de software relacionados con el desacoplamiento de componentes como la lógica empresarial, la interfaz de usuario, la navegación y la arquitectura de la información, los cuales afectan la calidad del producto final. Para abordar estos problemas, se propone un framework para elegir el patrón de arquitectura más adecuado en función del contexto de la aplicación.

1. Introducción a la Arquitectura de Software.

El artículo define la arquitectura de software como una descripción abstracta del sistema que va más allá del diseño detallado, enfocándose en la estructura general y las relaciones entre componentes a un nivel superior de abstracción. A diferencia del diseño, que se centra en la implementación concreta, la arquitectura utiliza elementos abstractos y conectores ricos en semántica para mediar la comunicación entre componentes. Este enfoque resalta la importancia de los conectores como elementos esenciales de primer nivel, siguiendo las ideas de Shaw y Garlan, que consideran la arquitectura como una disciplina que integra tanto los componentes como las interacciones entre ellos de manera fundamental para la estructura del sistema.

1. Principio del formulario
2. Final del formulario
3. Enseñanza de Arquitectura de Software en Modalidad Blended Learning: Análisis de la Percepción de Estudiantes de Maestría

El artículo examina la percepción de los estudiantes sobre el curso de Arquitecturas de Software Ágiles ofrecido en la Universidad de los Andes, el cual utiliza la modalidad de Blended Learning. Mediante una encuesta anónima aplicada a estudiantes de los períodos 2015-2018, se encontró que el 39,7% de los encuestados consideraron que aprendieron mejor en esta modalidad en comparación con los cursos presenciales. Además, el uso de problemas de la industria fue el aspecto más valorado por los estudiantes. Estos hallazgos subrayan la efectividad del Blended Learning en la enseñanza de la arquitectura de software y la importancia de integrar problemas reales en el proceso educativo.

1. Por qué el software para empresas se dirige hacia SOA: La Arquitectura Orientada a Servicios es el nuevo Santo Grial de los grandes jugadores del mercado IT.

El artículo destaca la creciente adopción de la Arquitectura Orientada a Servicios (SOA) en la industria del software, respaldada por estudios y proyecciones de consultoras líderes como IDC y Forrester Research. La SOA, que facilita la integración de sistemas y aplicaciones dispares mediante el uso de servicios, está siendo considerada la evolución natural de los servicios Web. Según Gartner, para 2006, más del 60% del mercado de servicios profesionales de TI se basará en estándares de servicios Web, y para 2008, el 80% de los proyectos de desarrollo adoptarán SOA

1. Arquitectura Orientada Al Servicio Service Oriented Architecture (Soa).

El artículo examina la Arquitectura Orientada a Servicios (SOA) como respuesta a la creciente complejidad de los procesos de negocio y la necesidad de integración en las organizaciones. Se detallan las capas y componentes de SOA, así como la definición de servicios web, incluyendo sus características, protocolos y estándares necesarios para su implementación.

1. Modelado y Verificación de Patrones de Diseño de Arquitectura de Software para Entornos de Computación en la Nube.

El trabajo presenta un entorno integral interno de diseño para arquitecturas de software enfocadas en aplicaciones web. Este entorno aborda problemas comunes en el diseño, ofreciendo módulos que asisten a los arquitectos en la creación de diseños de alta calidad.

1. Arquitectura basada en Microservicios y DevOps para una ingeniería de software continua.

Los microservicios y DevOps representan dos enfoques arquitectónicos clave en el desarrollo de software. Los microservicios se centran en construir aplicaciones a través de servicios independientes, escalables y colaborativos, capaces de adaptarse a entornos complejos. DevOps, por su parte, es un paradigma que promueve la entrega e integración continua de software, fomentando una cultura colaborativa y ágil que reduce la brecha entre desarrollo y operaciones. Este trabajo propone una arquitectura que combina microservicios y DevOps para lograr una ingeniería de software continua, aplicada en un caso de estudio con equipos de estudiantes de cursos de software. El resultado es un conjunto de aplicaciones desarrollado con tecnologías líderes y un enfoque innovador.

1. Software Architecture.

A medida que aumentan el tamaño y la complejidad de los sistemas de software, el diseño, la especificación y el análisis de la estructura global del sistema se convierten en cuestiones críticas. Estos problemas estructurales abarcan la organización del sistema en componentes, estructuras de control globales, protocolos de comunicación, sincronización y acceso a datos, asignación de funcionalidades, composición de elementos de diseño, distribución física, escalabilidad, rendimiento y evolución. Este es el nivel de diseño de la arquitectura de software.

1. A Systematic Review of the Project Management Information Systems in Different Types of Construction Projects 1

Los sistemas de información de gestión de proyectos (PMIS) son herramientas de software diseñadas para ayudar a los gerentes a seguir el progreso de los proyectos desde su diseño hasta su ejecución. Estos sistemas deben adoptar los métodos teóricos de gestión de proyectos, ofrecer asistencia en diferentes fases del ciclo de vida del proyecto y tener un alcance adecuado.

1. MOMMIV: Modelo para descomposición de una arquitectura monolítica hacia una arquitectura de microservicios bajo el Principio de Ocultación de Información

Los cambios tecnológicos actuales y la necesidad de actualizar aplicaciones legadas a plataformas modernas han impulsado la adopción de arquitecturas de microservicios. Sin embargo, este proceso presenta desafíos como la complejidad de extraer la lógica de negocio, los problemas derivados de modificar bases de datos relacionales y la posible duplicación o acumulación de funcionalidades en los microservicios.

Bibliografía

1. Palmero, M. A. S., Martínez, N. S., & Grass, O. Y. R. (2019). Revisión de elementos conceptuales para la representación de las arquitecturas de referencias de software. *Revista Cubana de Ciencias Informáticas*, *13*(1), 143–157. <http://scielo.sld.cu/scielo.php?pid=S2227-18992019000100143&script=sci_arttext>
2. *Scientia Et Technica*. (n.d.). Redalyc.org. Retrieved September 7, 2024, from <https://www.redalyc.org/pdf/849/84933912003.pdf>
3. *Vista de Especificando una arquitectura de software*. (n.d.). Edu.co. Retrieved September 7, 2024, from <https://revistas.udistrital.edu.co/index.php/tia/article/view/18076/19933>
4. Barrios, J. J. S., & Moreno, Y. (2016). Ingeniería de Software y computación en la nube: Conceptos básicos que deben ser enseñados en el curso de Ingeniería de Software I, Software Engineering and Cloud Computing.: Edición Semestral | Volumen 4, Número 2 |Julio - Diciembre de 2016. *Revista Plus Economía*, *4*(2), 33–40. <https://revistas.unachi.ac.pa/index.php/pluseconomia/article/view/32>
5. *Distinguiendo entre SaaS y SOA*. (2013, August 9). Juanjo’s Blog. <https://jjegonzalezf.wordpress.com/2013/08/08/distinguiendo-entre-saas-y-soa/>
6. (N.d.). Unirioja.Es. Retrieved September 7, 2024, from <https://dialnet.unirioja.es/servlet/articulo?codigo=4797401>
7. Bastarrica, M. C. (n.d.). *Atributos de Calidad y Arquitectura del Software*. Upm.Es. Retrieved September 7, 2024, from <http://www.grise.upm.es/rearviewmirror/conferencias/jiisic04/Tutoriales/tu4.pdf>
8. Navarro, M. E., Moreno, M. P., Aranda, J., Parra, L., Rueda, J. R., & Pantano, J. C. (2017). Integración de arquitectura de software en el ciclo de vida de las metodologías ágiles. XIX Workshop de Investigadores en Ciencias de la Computación (WICC 2017, ITBA, Buenos Aires).
9. (N.d.). Researchgate.net. Retrieved September 7, 2024, from <https://www.researchgate.net/profile/Jesus-Molina-10/publication/221595302_Ana_Arquitectura_para_una_Herramienta_de_Patrones_de_Diseno/links/58454a0208ae61f75dd6b3d4/Ana-Arquitectura-para-una-Herramienta-de-Patrones-de-Diseno.pdf>
10. with:, P. in C. (n.d.). *Arquitectura de software académica para la comprensión del desarrollo de software en capas*. Econstor.Eu. Retrieved September 7, 2024, from <https://www.econstor.eu/bitstream/10419/130825/1/837816424.pdf>
11. (N.d.). Edu.Co. Retrieved September 7, 2024, from <https://login.bdigital.sena.edu.co/login?url=https://www.proquest.com/scholarly-journals/marco-de-trabajo-para-seleccionar-un-patr%C3%B3n/docview/2562269745/se-2?accountid=31491>
12. (N.d.-b). Edu.Ar. Retrieved September 7, 2024, from <https://www.fceia.unr.edu.ar/~mcristia/Introduccion_a_la_Arquitectura_de_Software.pdf>
13. (S/f). Proquest.com. Recuperado el 9 de septiembre de 2024, de <https://www.proquest.com/docview/2385756012/FF4FAD4A986F4262PQ/11?accountid=31491&sourcetype=Scholarly%20Journals>
14. (S/f). Proquest.com. Recuperado el 9 de septiembre de 2024, de <https://www.proquest.com/docview/2692654293/854F70D8CAAF475DPQ/1?accountid=31491&sourcetype=Newspapers>
15. (N.d.-b). Edu.Co. Retrieved September 7, 2024, from <https://login.bdigital.sena.edu.co/login?url=https://www.proquest.com/scholarly-journals/arquitectura-orientada-al-servicio-service/docview/2972697988/se-2?accountid=31491>
16. (N.d.-a). Edu.Co. Retrieved September 7, 2024, from <https://login.bdigital.sena.edu.co/login?url=https://www.proquest.com/scholarly-journals/modelado-y-verificación-de-patrones-diseño/docview/2343015882/se-2?accountid=31491>
17. (N.d.-b). Redalyc.org. Retrieved September 7, 2024, from <https://www.redalyc.org/journal/816/81665362014/81665362014.pdf>
18. Garlan, D. (n.d.). *Software Architecture*. Cmu.edu. Retrieved September 7, 2024, from <https://kilthub.cmu.edu/articles/journal_contribution/Software_Architecture/6609593/files/12101711.pdf>
19. (N.d.-c). Proquest.com. Retrieved September 7, 2024, from <https://www.proquest.com/docview/2918346989/FF4FAD4A986F4262PQ/4?accountid=31491&sourcetype=Scholarly%20Journals>