**RESUMENES.**

1. **Revisión de elementos conceptuales para la representación de las arquitecturas de referencias de software:** Este estudio se centró en cómo se representa el conocimiento sobre las arquitecturas de referencia de software, que son modelos estándar que ayudan a diseñar sistemas de software. La investigación identificó las formas más comunes en las que se describen estas arquitecturas y también señaló qué elementos son importantes y cuáles presentan deficiencias.
2. **Análisis comparativo de patrones de diseño de software para el desarrollo de aplicaciones móviles de calidad: Una revisión sistemática de la literatura:** Este artículo revisó estudios sobre patrones de diseño de software, enfocados en el desarrollo de aplicaciones móviles de alta calidad. El objetivo fue identificar los estudios más relevantes y luego definir criterios que ayuden a seleccionar los mejores patrones de diseño.

Los resultados del estudio identificaron cinco patrones principales de diseño para desarrollo móvil de calidad y definieron ciertos criterios que permiten analizarlos. Estos criterios pueden usarse como una herramienta para comparar y seleccionar el patrón de diseño más adecuado para las necesidades de un proyecto específico.

1. **Modelado y verificación de patrones de diseño de arquitectura de software para entornos de computación en la nube:** Este artículo presenta una herramienta integral para diseñar arquitecturas de software enfocadas en aplicaciones web. Este entorno ayuda a los arquitectos a crear diseños de calidad al abordar los problemas más comunes en el proceso de diseño.

La herramienta se basa en un metamodelo que identifica y organiza los componentes arquitectónicos más comunes en este tipo de aplicaciones. Además, incluye una herramienta gráfica que permite construir estos diseños visualmente y verifica automáticamente la aplicación correcta de los patrones de diseño, asegurando que se sigan las mejores prácticas.

1. **Arquitectura de software para el sistema de visualización médica Vismedic:** Este trabajo propone una nueva arquitectura de software para el sistema de visualización médica Vismedic, combinando tres estilos arquitectónicos: componentes, capas, y tuberías y filtros. El objetivo es que se mejore la extensibilidad, reusabilidad y reducir dependencias. La arquitectura se basó en el análisis de productos como Voreen, VTK e ITK, e integra plugins para extender funcionalidades. La propuesta fue validada mediante prototipos y análisis (ATAM), confirmando que cumple con los criterios de calidad establecidos.
2. **Recuperación de Arquitecturas de Software: Un Mapeo Sistemático de la Literatura:** Este artículo presenta un estudio sobre cómo se recuperan vistas arquitectónicas de sistemas software, es decir, cómo se obtienen representaciones de la estructura y componentes de un sistema ya existente. Se revisaron estudios previos para entender qué técnicas usan, qué aspectos de la arquitectura recuperan y cómo representan esas vistas recuperadas. Se concluye que sería beneficioso tener un mecanismo unificado para representar estos resultados, lo que facilitaría su interpretación, análisis y reutilización en futuros proyectos.
3. **Patrones de diseño para la construcción de cursos on-line en un entorno virtual de aprendizaje:** El artículo propone patrones de diseño específicos para crear cursos online, integrando un enfoque pedagógico adecuado para la educación virtual. La propuesta fue evaluada en dos fases: primero, se probó la capacidad de los docentes para diseñar cursos usando estos patrones; luego, se evaluó la usabilidad del curso con los estudiantes, quienes encontraron el entorno más atractivo y fácil de usar.
4. **Patrones de Diseño GOF (The Gang of Four) en el contexto de Procesos de Desarrollo de Aplicaciones Orientadas a la Web**: Se establecieron criterios para evaluar procesos de desarrollo formales. Luego, se revisó el código fuente de varios proyectos para identificar qué patrones de diseño se están utilizando. Los resultados muestran que, aunque los patrones de diseño sí se usan en la industria, su aplicación es limitada. Esto se debe a que muchos desarrolladores no conocen estos patrones o no tienen suficiente experiencia para utilizarlos correctamente.
5. **Módulo de recomendación de patrones de diseño para EGPat:** Este trabajo presenta un entorno desarrollado por la Universidad de las Ciencias Informáticas para facilitar el uso de patrones de diseño en recursos educativos digitales. Estos patrones, aunque útiles, a menudo no se utilizan por desconocimiento o dificultad de acceso. El entorno ayuda a diseñar, revisar y recomendar patrones adecuados para resolver problemas específicos en la creación de materiales educativos.
6. **Análisis de los patrones de colaboración del Programa de Doctorado en Documentación e Información Científica:** Este artículo examina cómo colaboraron dos universidades, la Universidad de La Habana y la Universidad de Granada, se estudiaron las redes de colaboración en la coautoría de artículos y en la tutoría de tesis, así como las conexiones entre las instituciones y se descubrió que la mayoría de los artículos eran escritos por varios autores y que las universidades involucradas tuvieron un papel central.
7. **Modelo Teórico para la Identificacion del Antipatron "Stovepipe System" en la Etapa de la Implementacion de una Arquitectura de Software:** Los anti patrones son enfoques que complican más de lo que resuelven. En la Arquitectura de Software, pueden llevar a retrasos o fracasos si la solución elegida no funciona bien. Esto es un problema en grandes organizaciones con sistemas complejos, como las instituciones públicas en Bolivia.
8. **Monitoreo energético en los laboratorios de la Universidad de las Ciencias Informáticas:** Este Artículo propone una solución automatizada usando un sistema informático con la herramienta Nmap Network Mapper. Este sistema automatiza la recolección de datos y genera reportes que ayudan a los directivos a tomar decisiones para cumplir con sus planes energéticos.
9. **Softwares educativos:** Los softwares educativos son programas diseñados para apoyar el proceso de enseñanza y aprendizaje, facilitando la educación, el autoaprendizaje y el desarrollo de habilidades cognitivas.
10. **Tipificación de Dominios de Requerimientos para la Aplicación de Patrones Arquitectónicos:** Este artículo presenta los resultados de una investigación que apoyó el diseño de la arquitectura de software para un proyecto. Se creó un conjunto de requisitos comunes para proyectos web que integran plataformas digitales. Este conjunto se utilizó para relacionar y definir patrones arquitectónicos útiles para elegir el más adecuado en proyectos de desarrollo.
11. **Método ágil híbrido para desarrollar software en dispositivos móviles:** Este trabajo presenta un nuevo método ágil híbrido para desarrollar software en dispositivos móviles. El método se creó en cinco etapas, que incluyeron analizar métodos ágiles existentes, diseñar un enfoque evaluativo, y aplicar este enfoque para desarrollar y probar el nuevo método. Se describen los recursos y herramientas que utiliza el método, los cuales son útiles para desarrolladores y jefes de proyecto.
12. **Modelo de desarrollo para la construcción de aplicaciones móviles educativas:** El modelo propuesto integra estrategias didácticas para mejorar el aprendizaje y se basa en una investigación que analizó su efectividad con docentes de diez escuelas públicas. El modelo permite a los profesores crear aplicaciones móviles útiles para la educación primaria y secundaria, promoviendo prácticas docentes innovadoras.
13. **Desarrollo de una arquitectura de software para el robot móvil Lázaro:** La arquitectura tiene tres niveles: el primero gestiona los componentes básicos del robot, el segundo proporciona librerías para crear aplicaciones de control, y el tercero ofrece una interfaz de usuario con un panel de control y un simulador 3D para monitorear y programar el robot. El artículo también presenta pruebas para verificar cómo funciona esta arquitectura, que fue desarrollada en C#.
14. **Arquitectura de Software para el Soporte de Comunidades Académicas Virtuales en Ambientes de Televisión Digital Interactiva:** Se explican los escenarios que soporta, los componentes que facilitan la interactividad en la televisión digital, y los avances de su implementación en el laboratorio. Además, se detallan dos servicios que la arquitectura soporta: un tablón de mensajes y una sala de chat. La arquitectura permite gestionar y utilizar eficazmente el canal de retorno, integrando servicios flexibles tanto relacionados como no relacionados con el contenido multimedia.
15. **Influencia de la Ingeniería de Software en los Procesos de Automatización Industrial:** Las industrias necesitan sistemas flexibles para adaptarse a los cambios en sus procesos de producción, pero muchos sistemas actuales no ofrecen la agilidad necesaria. La ingeniería de software puede ayudar a mejorar estos sistemas, haciendo que sean más adaptables y capaces de integrar nuevas funcionalidades en la automatización industrial.
16. **Arquitectura de software para el desarrollo de aplicaciones web orientada a micro-servicios en TecNM campus Escárcega:** En el TecNM Campus Escárcega, se usaba un enfoque monolítico para el desarrollo de aplicaciones, pero ahora se está cambiando a micro-servicios para mejorar la productividad y reducir los tiempos de desarrollo. Esto muestra que el enfoque de micro-servicios es más eficiente y se alinea mejor con las tendencias tecnológicas actuales.
17. **Implementación de una solución web y móvil para la gestión vehicular basada en Arquitectura de Aspectos y metodologías ágiles: Un enfoque educativo de la teoría a la práctica:** Este artículo describe cómo se desarrolló una aplicación web y móvil para gestionar el parque automotor de la Universidad Técnica Particular de Loja utilizando metodologías ágiles y tecnologías de código abierto. La versión móvil de la aplicación ofrece las mismas funciones que la web, pero adaptadas para facilitar su uso en teléfonos y tabletas, mejorando la accesibilidad y usabilidad en cualquier momento y lugar.
18. **Patrones de diseño en Java: los 23 modelos de diseño: descripciones y soluciones ilustradas en UML 2 et Java:** Los patrones de diseño en programación son como soluciones predefinidas para problemas comunes que se presentan a menudo cuando estás escribiendo código. Son como recetas que te dicen cómo organizar tu código de manera que sea más claro, fácil de mantener y adaptable.
19. **Patrones Grasp y Anti-Patrones: un Enfoque Orientado a Objetos desde Lógica de Programación:** Este artículo propone una forma de enseñar a usar los patrones GRASP, que ayudan a asignar responsabilidades en el diseño de software, y también a evitar los anti-patrones, que son malas prácticas al programar. Es una metodología que enseña buenas prácticas de programación, con el objetivo de escribir código mejor estructurado y más eficiente, mientras se evita el uso de malas prácticas o errores comunes.
20. **Una Arquitectura para una Herramienta de Patrones de Diseño:** Los entornos de programación deberían tener herramientas que ayuden a usar estos patrones de diseño de manera más sencilla. Están proponiendo una forma de mejorar los programas con herramientas que faciliten el uso de patrones de diseño, apoyándose en una estructura que los haga más manejables y fáciles de aplicar.
21. **El patrón de diseño Modelo-Vista-Controlador (MVC) y su implementación en Java Swing:** Cuando los profesores enseñan programación, se enfrentan a un problema común: muchos estudiantes tienden a escribir código de manera desordenada, como si intentaran resolver todo en un solo bloque enorme. En ese código, mezclan cosas como el procesamiento de datos, colores, mensajes al usuario, animaciones, gráficos, y mucho más. Esto hace que el código sea difícil de entender, complicado de modificar y casi imposible de mantener o mejorar en el futuro.
22. **Propuesta de implementación simple del patrón de diseño modelo vista controlador sobre php sin el uso de frameworks y utilizando programación orientada a objetos:** El objetivo es ofrecer un modelo base para desarrollar aplicaciones escalables, donde un equipo de programadores pueda trabajar fácilmente añadiendo componentes o bibliotecas sin complicaciones.
23. **Arquitectura de software con programación orientada a objeto:** Este artículo habla sobre cómo la Programación Orientada a Objetos (POO) ayuda a desarrollar y mantener programas complejos de manera más sencilla. El artículo concluye que la evolución de la arquitectura de software permite a los profesionales crear soluciones de mejor calidad y más rápido, lo que mejora la satisfacción del usuario.
24. **Lenguajes de Patrones de Arquitectura de Software: Una Aproximación Al Estado del Arte:** Este artículo explora el estado actual de los lenguajes de patrones en la arquitectura de software, desde sus inicios hasta los avances más recientes y su uso en la creación de arquitecturas en diferentes áreas. Permiten crear sistemas de información aplicables a distintos dominios, ofreciendo flexibilidad y extensibilidad en el diseño de software.
25. **Marco de referencia de arquitectura de software para aplicaciones web y móviles:** Este artículo presenta un marco de referencia para desarrollar aplicaciones web y móviles usando software libre y código abierto, con el objetivo de automatizar tareas y mejorar la respuesta de los usuarios.
26. **Arquitectura de software para los actuales sistemas ciberfísicos:** Este artículo propone una arquitectura de software que permita diseñar, implementar y combinar servicios de manera flexible según la demanda. Estos servicios deben cumplir con requisitos específicos de confianza, seguridad, eficiencia, confiabilidad y previsibilidad, todo mientras se ajustan a las capacidades del hardware disponible.
27. **Arquitectura orientada a servicios para software de apoyo para el proceso personal de software:** El software se destaca por ser extensible e independiente, lo que significa que permite agregar nuevas herramientas al proceso de desarrollo sin depender de sistemas operativos o lenguajes de programación específicos.
28. **Diseño web adaptativo para la plataforma educativa ZERA:** El artículo destaca cómo se mejoró ZERA usando técnicas de diseño web adaptativo, como HTML5, el framework de CSS Bootstrap, y adaptadores de imágenes. Después de implementar estas mejoras, se comprobó que la plataforma se adapta bien a diferentes tamaños de pantallas y dispositivos móviles.
29. **Implementación de un framework para el desarrollo de aplicaciones web:** El proyecto propone un framework que facilita separar la lógica de negocios, el acceso a datos y la presentación de información. Esto permitiría generar contenido en diferentes formatos sin complicar o repetir el código, simplificando el proceso y adaptándose a las nuevas necesidades del mercado.
30. **Patrones de multiculturalidad para el diseño De aplicaciones web:** Este artículo examina cómo varía la comunicación multicultural en la web, basándose en teorías culturales de autores como Hofstede y Hall. Se analizan elementos de diseño web en sitios representativos de Argentina, Colombia y España, para finalmente desarrollar un conjunto de patrones de diseño que ayuden a crear aplicaciones web más inclusivas y adaptadas a diversas culturas.
31. **Lineamientos para el diseño de aplicaciones web soportados en patrones GRASP:** El texto propone un proyecto que ayude a los estudiantes a entender mejor los patrones de diseño GRASP, que son útiles en la programación orientada a objetos. Esto les permitirá mejorar sus habilidades para desarrollar software de calidad y con buenas prácticas.
32. **Sobre Comercio electrónico en la WEB 2.0 y 3.0:** Se examina la seguridad informática, cubriendo sus principios, políticas, arquitectura, amenazas y estrategias de protección en la Web. El estudio utiliza varios métodos de análisis para entender el impacto y las perspectivas de este entorno digital, destacando la necesidad de que las empresas adapten nuevas estrategias ante los rápidos cambios en la web.
33. **Arquitectura de software para la construcción de un sistema de cuadro de mando integral como herramienta de inteligencia de negocios:** La Inteligencia de Negocios ofrece herramientas que permiten extraer, limpiar y analizar datos dentro de una empresa, sin importar su tamaño.
34. **Integración de arquitectura de software en el ciclo de vida de las metodologías ágiles:** El texto explica cómo las Metodologías Ágiles se enfocan en el trabajo en equipo, la adaptabilidad y la colaboración, tanto dentro del equipo de desarrollo como con los usuarios finales. Debido a que los requisitos de los proyectos de software suelen cambiar a lo largo del tiempo, las MA evitan planificaciones detalladas al inicio, asumiendo que los cambios son inevitables.
35. **Selección de metodologías ágiles e integración de arquitecturas de software en el desarrollo de sistemas de información:** Se trata sobre la importancia de la Ingeniería de Software en las organizaciones, que depende mucho de los Sistemas de Información y las Tecnologías de la Información. En los últimos años, las Metodologías Ágiles han ganado popularidad porque son más efectivas que las metodologías tradicionales en entornos donde las necesidades y tecnologías cambian rápidamente. Las MA son ideales para proyectos a corto plazo con requisitos en constante evolución.
36. **Representación y razonamiento sobre las decisiones de diseño de arquitectura de software:** Este artículo busca ayudar a los arquitectos de software durante el diseño arquitectónico. Parte de la idea de que estos profesionales reutilizan soluciones conocidas en nuevos proyectos.
37. **Comparación de métodos para la arquitectura del software: Un marco de referencia para un método arquitectónico unificado:** Este artículo trata sobre la importancia de la visión arquitectónica en el desarrollo de software y cómo se han propuesto diferentes métodos para el diseño y evaluación de arquitecturas. Estos métodos se basan en conceptos que pueden ser similares, complementarios o alternativos.