**ARTÍCULO 1**

**Título:** Refactorización Orientada a Aspectos de Aplicaciones Empresariales

**Nombre de la revista o congreso donde se ha publicado:** Sociedad Argentina de Informática (SADIO) → Jornadas Argentinas de Informática e Investigación Operativa (JAIIO) → 40 Jornadas Argentinas de Informática e Investigación Operativa (JAIIO) → XII Argentine Symposium on Software Engineering (ASSE 2011)

**Dirección web donde localizarlo:** <https://sedici.unlp.edu.ar/handle/10915/125480>

**Autores:**[Zunino, Guillermo](https://sedici.unlp.edu.ar/discover?filtertype=author&filter_relational_operator=equals&filter=Zunino,%20Guillermo) | [Vidal, Santiago](https://sedici.unlp.edu.ar/discover?filtertype=author&filter_relational_operator=authority&filter=http://voc.sedici.unlp.edu.ar/node/60900) | [Marcos, Claudia A.](https://sedici.unlp.edu.ar/discover?filtertype=author&filter_relational_operator=authority&filter=http://voc.sedici.unlp.edu.ar/node/53294)

**Fecha de publicación:** 2011

**Resumen:**

El documento "Refactorización orientada a aspectos para aplicaciones empresariales" detalla la importancia de simplificar el mantenimiento y la evolución de las aplicaciones empresariales, especialmente las aplicaciones complejas que requieren una gestión eficiente de características y requisitos. Destaca la necesidad de implementar mecanismos de modularización más allá de la orientación a objetos para abordar de manera efectiva las diversas cuestiones que afectan la estructura de estas aplicaciones.

En este contexto, se destaca la relevancia del marco Spring y su módulo de Programación Orientada a Aspectos (AOP) como una herramienta importante para una separación más clara y efectiva de las preocupaciones en las aplicaciones comerciales. El uso de AOP junto con Spring le permite organizar y administrar mejor los aspectos transversales que pueden afectar múltiples componentes de su aplicación.

Este artículo presenta dos estudios de caso que aplican técnicas de refactorización para abordar preocupaciones generales en aplicaciones comerciales. Describe el proceso de migración iterativa. Esto incluye identificar inquietudes que existen en el código, aplicar refactorizaciones compatibles con Spring/AOP y analizar los resultados obtenidos al modularizar estas inquietudes.

También analizamos el trabajo relacionado en el área de la transición de sistemas orientados a objetos a sistemas orientados a aspectos y destacamos la importancia de estrategias de refactorización efectivas para transiciones exitosas. Destaca la relevancia de utilizar herramientas y enfoques apropiados para mejorar la estructura y la capacidad de mantenimiento de las aplicaciones comerciales, lo que contribuirá a la eficiencia y la calidad de las aplicaciones a lo largo del tiempo.

**ARTÍCULO 2**

**Título:** Semantic Code Refactoring for Abstract Data Types

**Nombre de la revista o congreso donde se ha publicado:** Proceedings of the ACM on Programming LanguagesVolume 8Issue POPLArticle No.: 28pp 816–847

**Dirección web donde localizarlo:** <https://dl.acm.org/doi/abs/10.1145/3632870>

**Autores:**Shankara Pailoor, Yuepeng Wang, Işıl Dillig

**Fecha de publicación:** 2024

**Resumen:**

El texto examina la refactorización de software como un proceso crucial para mejorar la calidad del software al cambiar su estructura interna sin afectar su funcionamiento externo. La dificultad de encontrar una secuencia ideal de refactorizaciones es resaltada, y se menciona cómo las técnicas de búsqueda, utilizadas en el ámbito de la ingeniería de software basada en búsqueda (SBSE), pueden enfrentar este reto. Se presenta el concepto de Refactoring Basado en Búsqueda (SBR) como un campo en desarrollo que emplea métodos de búsqueda para llevar a cabo refactorizaciones de software, resaltando su expansión y utilidad en múltiples desafíos de ingeniería de software. El objetivo de la revisión sistemática del artículo es ofrecer una perspectiva detallada sobre las características y tendencias del SBR, destacando áreas de investigación y contribuciones importantes en este ámbito

**ARTÍCULO 3**

**Título:** Estrategias y herramientas para depuración de código en el back-end

**Nombre de la revista o congreso donde se ha publicado:** Vol. 1 Núm. 1 (2023): Cuadernos Técnicos Universitarios de la DGTIC

**Dirección web donde localizarlo:** <https://cuadernos.tic.unam.mx/index.php/cua/article/view/24/41>

**Autores:**Luz María Ramírez Romero

**Fecha de publicación:** 2023

**Resumen:**

El artículo "Estrategias y herramientas para depuración de código en el back-end" ofrece una visión detallada sobre la importancia crucial de depurar el código en el proceso de desarrollo de sistemas de información. Destaca la relevancia de garantizar la calidad de los sistemas liberados en producción, libres de errores que puedan afectar la experiencia del usuario y el cumplimiento de los objetivos del sistema.

Se mencionan diversas estrategias y herramientas que los programadores pueden emplear para llevar a cabo la depuración de manera efectiva. Entre ellas se incluyen cambios en metodologías como la aplicación de Scrum, el uso de Inteligencia Artificial para identificar rutinas de código probadas y la refactorización del código. Además, se resalta la utilidad de técnicas como el Test Driven Development (TDD) para automatizar pruebas de módulos individuales, así como el uso de herramientas específicas de Laravel/Livewire, como Tinker y la barra de depuración de Laravel, y extensiones integradas al editor de código Visual Studio Code.

El artículo también hace hincapié en la importancia de las herramientas tradicionales de depuración, que permiten a los desarrolladores revisar y verificar líneas de código de forma parcial antes de integrarlas al código principal. Se subraya que cualquier enfoque, técnica o herramienta relacionada con la depuración de código debe ser considerada para garantizar la confiabilidad del sistema desarrollado y superar la llamada Crisis del Software.

En resumen, el documento proporciona una guía valiosa para los desarrolladores sobre cómo abordar la depuración de código en el back-end, destacando la necesidad de contar con estrategias sólidas y herramientas adecuadas para asegurar la calidad de los sistemas de información [T1], [T2], [T4].