ASPECT-ORIENTED PROGRAMMING

Applying AspectJ to J2EE Application Development
by Nicholas Lesiecki, pp. 24–32. This article focuses on the application of AspectJ to the development of a Java 2 Enterprise Edition Web application for Video Monitoring Services of America (VMS). The project used aspects to cleanly modularize concerns ranging in scope from auxiliary (error handling) to application-specific (shopping basket price calculation) to framework-level (object relationship management). VMS saw benefits resulting from the aspect-oriented implementation of these concerns in the areas of code size, understandability, and reduced defects.

Dynamic Adaptation of the Squid Web Cache with Arachne
by Marc Ségura-Devillechaise, Jean-Marc Menaud, Nicolas Loriant, Thomas Fritz, Rémi Douence, Mario Südholt, and Egon Wächter, pp. 34–41. Three adaptations of the Squid Web cache show how the authors use their Arachne system for AOP: correcting a security hole, introducing prefetching, and extending a protocol. Arachne’s expressive aspect language, especially its notion of sequence aspects, allows for concise modularization, and its dynamic weaver lets developers perform such adaptations without a perceptible performance overhead.

Unraveling Crosscutting Concerns in Web Services Middleware
by Bart Verhecke, Wim Vanderperren, and Viviane Jonckers, pp. 42–50. Current approaches to integrate Web services are inflexible, affecting the short-term adaptability and long-term evolution of the service, network, and business environment. To enable the development of more flexible and robust applications, the Web Services Management Layer defines middleware for the dynamic integration, selection, composition, and client-side management of Web services in service-oriented architectures. WSML uses dynamic aspect-oriented programming to solve several crosscutting concerns in SOAs. An industrial case study shows WSML used in the context of broadband service provisioning.

Modular Software Design with Crosscutting Interfaces
by William G. Griswold, Kevin Sullivan, Yuanyuan Song, Macneil Shonle, Nishit Tewari, Yuanyang Cai, and Hridesh Rajan, pp. 51–60. Aspect-oriented programming languages such as AspectJ offer new mechanisms for decomposing systems into modules and composing modules into systems. Common ways of using these mechanisms couple aspects to complex, changeable implementation details, which can compromise modularity. The crosscut programming interface (XPI) can significantly improve modularity in the design of programs employing AspectJ-style AOP. The use of XPIs doesn’t limit the use of existing AOP mechanisms or require new ones, and the approach appears to generalize to other languages.

Discovering Early Aspects
by Elisa Baniassad, Paul C. Clements, João Araújo, Ana Moreira, Awais Rashid, and Bedir Tekinerdöğan, pp. 61–70. Traditionally, aspect-oriented software development has focused on the software life cycle’s implementation phase: aspects are identified and captured mainly in code. But aspects are evident earlier in the life cycle, such as during requirements gathering and architecture development. The authors’ integrated approach to working with early aspects lets you identify them and exploit them throughout the software development life cycle.

FEATURES

Portable C/C++ Code for Portable XML Data
by Zhaoqing Wang and Harry H. Cheng, pp. 76–81. Using C/C++ to develop XML applications is challenging because the code developed in one platform can’t readily run in another. Ch is a cross-platform embeddable C/C++ interpreter. Ch XML packages for the Gnome libxml2 XML library and Oracle XML Developer’s Kit for C/C++ let developers execute XML applications written in C/C++ interpretively without compilation across different platforms.

Mining Software Repositories for Model-Driven Development
by Yuefeng Zhang and Dhaval Sheth, pp. 82–90. Software developers traditionally use data repositories for record keeping. A statistical process control method for mining software repositories with uncertainty in a model-driven development environment creates a new, active use for these repositories—predicting and planning various aspects of model-driven development projects.

Helping Small Companies Assess Software Processes
by Christiane Gresse von Wangenheim, Alessandra Anacleto, and Clênio F. Salviano, pp. 91–98. Developing and applying guidance for ISO/IEC 15504-conformant process assessments in small software companies, the authors enhanced existing assessment methods by considering the organization as a whole. They tailored the assessment to address business factors as well as process maturity and growth.