VIM: Taming Software with Hardware
pp. 21-25
Mark Halpern

The programming world generally agrees that software should never be written more than once. The author proposes something more radical: Much software doesn’t need to be written even once—particularly software devoted to the task of memory management.

By giving programmers so much physical memory that they can proceed as if they were working with infinite memory, virtual infinite memory could make all the software dedicated to this task unnecessary. If successfully implemented, this would give applications a practically limitless amount of real memory.

Solving Awari with Parallel Retrograde Analysis
pp. 26-33
John W. Romein and Henri E. Bal

In awari, a two-person game of pure skill, players sow stones into pits on a board. The game’s rules define how to capture stones, and the player who captures the most wins the game.

For more than a decade, researchers have studied computerized techniques to play awari. The authors have now solved the game by determining the score of 889,063,398,406 board positions and the game by determining the score of 889,063,398,406 board positions and the game by determining the score of 889,063,398,406 board positions and the game by determining the score of 889,063,398,406 board positions and the game by determining the score of 889,063,398,406 board positions.

Proposed orchestration and choreography standards must meet several technical requirements that address the language for describing the process workflow and the supporting infrastructure.

Turning Software into a Service
pp. 38-44
Mark Turner, David Budgen, and Pearl Brereton

The authors explore the concept of software as a service, which envisions a demand-led software market in which businesses assemble and provide services when needed to address a particular requirement. The SaaS vision focuses on separating the possession and ownership of software from its use.

Delivering software’s functionality as a set of distributed services that can be configured and bound at delivery time can overcome many current limitations constraining software use, deployment, and evolution.

Web Services Orchestration and Choreography
pp. 46-52
Chris Peltz

The terms orchestration and choreography describe two aspects of emerging standards for creating business processes from multiple Web services. The two terms overlap somewhat, but orchestration refers to an executable business process that can interact with both internal and external Web services. Orchestration always represents control from one party’s perspective. This distinguishes it from choreography, which is more collaborative and allows each involved party to describe its part in the interaction.

Proposed orchestration and choreography standards must meet several technical requirements that address the language for describing the process workflow and the supporting infrastructure.

CPXe: Web Services for Internet Imaging
pp. 54-62
Timothy Thompson, Rick Weil, and Mark D. Wood

The Common Picture eXchange environment is a highly interoperable service delivery framework that leverages the Web services paradigm to give providers access to an expanded market and offer consumers a broad range of digital imaging services. Multiple providers can register their services in a central directory and precisely characterize their offerings using an extensible catalog and order model as well as a comprehensive dictionary of terms. A service locator mechanism lets consumers easily select vendors offering the products and features they desire.

Ontology-Mediated Integration of Intranet Web Services
pp. 63-71
Tse-Ming Tsai, Han-Kuan Yu, Hsin-Te Shih, Ping-Yao Liao, Ren-Dar Yang, and Seng-chiong T. Chou

ML-based Web services standards have simplified integration with open Internet protocols and given machines an enhanced ability to communicate. Developers are striving to tap into the depth of mutual understanding that can be culled from the Web’s diverse content to provide a vast increase in available services.

Dealing with this flood of options will require sweeping automation. To meet this challenge, the authors built their smart office task automation framework—SOTA—using Web services, an ontology, and agent components.

Using Machine-Learning Methods for Musical Style Modeling
pp. 73-80
Shlomo Dubnov, Gerard Assayag, Olivier Lartillot, and Gill Bejerano

Constructing a musical theory from examples presents an intellectual challenge that could foster a range of new creative applications. Thus, the authors sought to apply machine-learning methods to the problem of musical style modeling. Their work has produced examples of musical generation and applications to a computer-aided composition system. Using statistical and information-theoretic tools that analyze musical pieces, they seek to capture some of the regularity apparent in the composition process. The resulting models can be used for inference and prediction, and to generate new works that imitate the great masters’ styles.