Guest Editor's Introduction

Knowledge-Based Software Engineering

Peter G. Seltfridge, AT&T Bell Laboratories

The AI community has recently seen a resurgence of interest in software engineering, as demonstrated by the recent flurry of professional meetings: the Software Engineering and Knowledge Engineering Conference, the Knowledge-Based Software Engineering Conference, AAAI Workshops on Automating Software Design, and the Workshop on Applying AI to Software Problems at CAIA. As the result of continuing publicity of real-world software failures and through trying to scale up AI prototypes, researchers increasingly recognize that software-engineering problems are both hard and interesting.

There are four primary reasons that software engineering is an interesting area for AI research. First, writing large software systems is a complex activity that requires a great deal of individual and organizational intelligence. This appeals to AI researchers, especially those who are trying to step up from small, "toy problem" domains. Second, an explosion of tools and techniques -- including CASE tools, object-oriented programming methodologies, fourth-generation languages, "visual languages," and a variety of new software development environments -- have failed to solve software-engineering problems. Third, software problems combine issues that have been studied in isolation, such as human interface problems, computer-supported cooperative work, basic AI representation issues, knowledge retrieval and reuse problems, and visualization. Finally, software engineering is critically important. The "software crisis" is ever more acute, and its solution is one of the major technological challenges of our society.

Knowledge-based software engineering (KBSE) is another name for applying AI technology to software problems, emphasizing the fact that creating software is a knowledge-intensive activity. In fact, it is the amount and scope of relevant knowledge that makes software so difficult. Creating a large software system requires knowledge of the domain, the final implementation platform (including hardware and software), the current software process, and the organizational and personnel resources. The KBSE approach assumes that making more knowledge available to individual programmers, teams, and managers will facilitate the timely production of high-quality software.

But KBSE researchers must answer several crucial questions:

What part of the software process is targeted? Producing large software systems is a complex, multistep process that can span years and produce a wealth of artifacts such as user requirements, system specifications, code, testing scenarios, and documentation. While a fully integrated environment is the ultimate goal, any current KBSE research project must carefully target some single part of this process.

What knowledge is applicable and how can it be represented, acquired, and main-
After the appropriate knowledge is represented in the system, maintaining that knowledge base is as critical as maintaining the code base or documentation base. All large systems evolve, and knowledge must be kept up to date.

How can we present the knowledge to developers, teams, and managers to improve the quality, cost, and timeliness of software development? The presentation and integration of the knowledge-based approach into the everyday working world of software engineers is a critical challenge for the KBSE community.

The special series

These first two articles in IEEE Expert's special series on knowledge-based software engineering are derived from work presented at the Sixth Conference on Knowledge-Based Software Engineering (KBSE '91). Twenty-five papers, three panels, and about 12 demonstrations were presented over three days. The two articles in this issue illustrate the scope and promise of KBSE.

The first article describes the Lockheed Environment for Automatic Programming, which provides intelligent assistance to individuals and groups developing large software systems. LEAP represents domain-specific knowledge about reusable components and helps the user access and modify those components to build applications; user-modified components are then entered back into the knowledge base. Users manipulate the components via a graphical representation from which the system automatically generates code (such as in Ada). This article outlines the LEAP development paradigm and the kinds of knowledge that LEAP represents, and describes using the system to develop control software for an autonomous underwater vehicle.

The second article — "Rule Chaining in Marvel: Dynamic Binding of Parameters" — describes a rule-based software development environment (RBDE) that addresses the entire software-engineering process by modeling it as a set of rules. These rules capture a variety of dependencies and constraints between software objects (stored in an object base) and guide the process: As objects are changed, rules fire and enforce dependencies, schedule other work, and maintain consistency between various software objects. In this article, the authors address the proper binding of parameters for rules, which by their nature fire dynamically. The system's performance depends on the bindings of proper objects from the object base. After experimenting with various heuristics, the authors describe a comprehensive approach that solves this problem.

References


Peter G. Selfridge is a member of the technical staff at AT&T Bell Laboratories. He joined the AI Principles Research Dept. in 1986, where he began work in the application of AI techniques to problems in software, particularly problems of understanding and maintaining very large software systems. His other research interests include interactive database environments, data and software visualization, knowledge-base/database integration, and tools for providing software design advice. Selfridge received his PhD in computer vision from the University of Rochester in 1981.

He can be reached at AT&T Bell Laboratories, Room 2B-425, 600 Mountain Ave., Murray Hill, NJ 07974; Internet, pgs@att.research.com