*The Knowledge Engineering Review*, Vol. 7:4, 1992, 367–369

# Knowledge-Based Software Engineering

W. LEWIS JOHNSON
*USC/ Information Sciences Institute, 4676 Admiralty Way, Marina del Rey, CA 90292-6695*

## 1. Introduction

The 7th Annual Knowledge-Based Software Engineering Conference was held at the McLean Hilton at Tysons Corner, in McLean, Virginia, on Sept. 20-23, 1992. This conference was sponsored by Rome Laboratory and held in cooperation with the IEEE Computer Society, ACM SIGART and SIGSOFT, and the American Association for Artificial Intelligence (AAAI).

The focus of the KBSE conferences is the application of artificial intelligence and knowledge-based techniques to software engineering problems. This includes techniques for constructing, representing, reasoning about, understanding and adapting software artifacts and processes. The conferences are concerned with all activities related to software, including project planning, domain modeling, requirements acquisition, specification, design, coding, documentation, understanding, reuse, evolution, testing and maintenance; providing intelligent tools that can perform these activities, support humans in performing them or cooperate with humans in performing them.

In 1983 RADC (now, Rome Laboratory) published a report calling for the development of a knowledge-based software assistant (KBSA), which would employ artificial intelligence techniques to support all phases of the software development process. It was anticipated that the use of KBSA would result in radical changes in the process of software development. Not only would the productivity of software developers change by orders of magnitude, but the tasks that software developers perform would change, as their efforts focus more on specification prototyping rather than coding. The KBSA system would take over much of the role of program synthesis, documentation, and testing.

The annual KBSA Conference provided a forum for discussion and presentation of work related to the KBSA effort. In 1991, the conference expanded its scope to include other work in knowledge-based software engineering, and changed its name to the Knowledge-Based Software Engineering Conference. Since then, a number of steps have been taken to establish the KBSE conference as a major technical conference in this area, such as recruiting an international program committee, and developing cooperation with major professional organizations.

The core of this year's conference was a three-day block of technical presentations, including panels, paper sessions, and demonstration sessions. The main conference was preceded by a day of tutorials.

The conference proceedings may be ordered from IEEE Computer Society Press, P.O. Box 3014, Los Alamitos, CA 90720-1264, order number 2880. A more detailed summary of the conference appears in the SIGART Bulletin, vol. 4 no. 1, January 1993.

## 2. Major Themes

The papers and presentations at the conference provided a wide variety, even divergence, of perspectives on knowledge-based software engineering. This variety is to be expected, since software engineering subsumes a number of different activities and problems, and artificial intelligence offers a number of techniques to bring to bear on these problems. There were groups of papers that focused on common issues, such as transformational synthesis or reuse. On the other hand, there were papers that focused on topics virtually ignored by the rest of the conference, such as software configuration management, or training of KBSE professionals.

Nevertheless, there were some recurrent themes at this year's conference. One was the advantages of making KBSE systems domain-oriented, that is specializing them so they meet the needs of software practitioners and users in particular application areas. Another theme was how to get KBSE systems adopted and used by software engineers.

The theme of domain orientation was prominent in the two invited presentations at the conference, by Elaine Kant of Schlumberger Laboratory for Computer Science, and Gerhard Fischer of the University of Colorado.

Dr. Kant's talk was concerned with knowledge-based support for scientific programming. Kant observed that until recently scientific computing was carried out by mathematical modelers writing Fortran programs. However, the programming process has become more time-consuming as faster machines make it feasible to tackle more complex problems. Furthermore, the scientific concepts and principles underlying scientific programs are well understood and relatively easy to codify. Scientific computing therefore is an attractive domain in which to automate code synthesis. Dr. Kant has been working for several years on developing such a system, called SINAPSE, which generates finite difference programs to simulate partial differential equation models, optimized for various target languages and architectures. In SINAPSE the requirements of a given program can be described interactively, in terms of the relevant mathematical concepts rather than programming concepts. The project demonstrates the effectiveness of a domain-oriented approach, insofar as it has been used successfully to generate a wide variety of codes in this particular application domain. Kant appears to have been rather successful at getting potential users interested and involved in the project.

Gerhard Fischer's presentation gave an overview of a new paradigm for automated design support, which he calls Domain-Oriented Design Environments (DODE). In his talk he pointed out a number of problems with conventional software engineering, which in his view are not adequately addressed in current knowledge-based software engineering systems. First, the hardest part of designing complex systems is deciding what the system should do. Systems that focus on generation of code from specifications ignore the hard task of getting the specifications right in the first place. Second, an understanding of the desired properties of a system arises only through interaction and evaluation of designs. These designs must be expressed in "languages of doing" i.e., notations that designers can manipulate directly. Specification-based approaches that separate specification from implementation are bound to fail, in Fischer's view, because they distance the designer from the design activity, and because specification languages are not intended as languages of doing. Finally, the design environment should be domain-oriented, employing notations that are familiar to domain specialists in their daily practice, and incorporating knowledge about the domain. Fischer went on to describe a number of prototypes developed in his laboratory that embody the principles that he described. However, in response to a question from the audience, Dr.Fischer acknowledged that his prototypes rely on a visual, Mac-Draw-like metaphor, and it is not clear how applicable this metaphor is to complex software development.

An example of a successful effort at getting KBSE technology adopted by software practitioners, is the work of Peter Selfridge, Loren Terveen, and David Long, as described in their paper titled "Managing Design Knowledge to Provide Assistance to Large-Scale Software Development". This paper describes an experiment in which design ``folklore" was captured in an advice-giving tool. This tool, developed in cooperation with production software engineers, is in current use.

Later in the conference, Dr. Selfridge chaired a panel titled "Assessing KBSE Research: Issues in Goals, Metrics, and Transferability", whose participants were Barry Boehm of the University of Southern California, Gerhard Fischer, Douglas Smith of the Kestrel Institute, Louis Hoebel of Rome Laboratory, Glover Ferguson of Andersen Consulting, and myself. The purpose of the panel was to discuss how to evaluate KBSE research, and how to get it used. Doug Smith argued that formal mathematical analysis was critical for evaluation of research results, and there was not much disagreement about this. A number of the panelists raised questions about what is really involved in getting revolutionary techniques into practice, and what the consequences of such revolutionary steps might be. Generally, evolutionary adoption is much more likely to succeed than revolutionary adoption; in fact, Selfridge questioned whether revolutionary adoption was likely ever to occur. Lewis Johnson, using Columbus's discovery of America as a metaphor, observed revolutionary steps are likely to lead to unforeseen consequences. Glover Ferguson noted that many of the problems that software engineers face are so severe that the only recourse is revolutionary change.

**3. Observations**

It would appear from this conference that the field of knowledge-based software engineering is in a transitional phase, and its apparent promise is not fully realized. KBSE is an application-oriented discipline, yet there are few real applications in current use.

Yet the situation now is quite different from what it was even a few years ago. Automatic programming was an active area of research in the 60's and 70's, and then for quite a few years there was little apparent progress. This was in part because fully automatic code synthesis is extremely hard, and could only be applied to the construction of relatively small programs. It was several years before the KBSA report called for a departure from automatic programming, addressing the broader issues of software engineering while employing more interactive techniques. The KBSA report itself projected that a fifteen-year effort would be required before KBSA systems were fully usable. The KBSA efforts have moved progressively closer to industrial-strength systems. Systems developed as part of the KBSA program were demonstrated at the conference, including USC/Information Science Institute's ARIES system, Kestrel Institute's KIDS system, and Andersen Consulting's KBSA Concept Demonstration.

The KBSA program has so far been successful at meeting its objectives. There is good reason to believe that a full-fledged KBSA system will be available by the turn of the millennium, as the KBSA report projected. However, the field of KBSE seems to be asking itself whether it makes sense to wait that long before introducing KBSE technology into the workplace. Domain-oriented techniques seem to offer promise of making this possible. But if KBSE technology is put into practice more quickly, it is unlikely to produce the kind revolutionary change in the software development process that the KBSA program envisions. Efforts such as Selfridge's that produce marginal improvements in current software development practice are much more likely to be the norm. The question that still requires a satisfactory is answer is whether revolutionary change is feasible, and if so when it is desirable.

Gauging from the reactions of attendees, it is apparent that a great deal of cohesion and common interests within the KBSE community, even though workers on the field focus on different software engineering problems. Attendees tended not to pick and choose which sessions to attend, but instead tried to attend as many sessions as possible. Attendees were therefore unusually critical of the organization of the conference program into parallel sessions. The degree of shared interests is good news for KBSE as a field, but is something that future KBSE conferences will have to take into account when developing their programs.

Judging from my own observations and from the evaluations of the attendees, this year's KBSE Conference was quite successful. Attendees rated the conference quite highly, and the vast majority of first-time attendees indicated that they would plan to attend future KBSE conferences. When the KBSE conference format was first introduced in 1991, there was some question as to whether there was sufficient need for an annual conference in this field. At this point the consensus of the community is that these annual conferences are viable and desirable, and will be continued. Plans are already underway for upcoming conferences. Next year's conference will be held in Chicago, chaired by Bruce Johnson, the head of Andersen Consulting's research facility. The 1994 conference will be held in California, chaired by Douglas Smith of the Kestrel Institute. I encourage those interested in knowledge-based software engineering to keep these future conferences in mind

**Acknowledgements**

I would like to thank the program committee members and conference attendees who contributed their thoughts and observations to this summary. Barbara Radzisz's assistance in collecting, collating, and summarizing conference evaluations was also greatly appreciated.