# ROSE: Making Compiler Technology

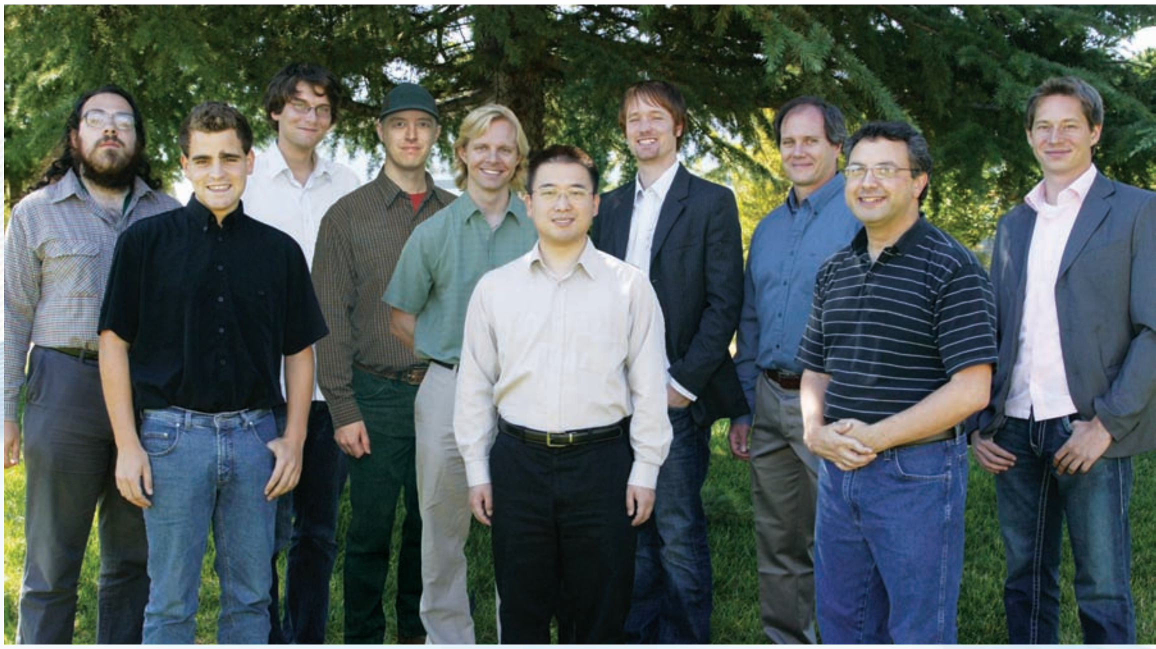
IN software development, a critical step is compiling—converting human-friendly source code into the machine-friendly binary code a computer needs to execute a program (that is, "executable" files). The conversion of source code into binaries is performed by specialized software applications called compilers. Only compiler experts, rare in the industry, know the ins and outs of this arcane task. With the increasing complexity of computing today, the conventional reliance on these experts to build sophisticated compilers has become a bottleneck.

Computer scientists at Lawrence Livermore have radically altered the programming landscape by creating ROSE, an open-source customizable compiler infrastructure that gives all programmers easy access to complex, automated compiler technology and assistance. ROSE accepts code in today's most common programming languages, including C, C++, and Fortran. Fortran is commonly used at Department of Energy (DOE) and National Nuclear Security Administration (NNSA) sites.

Because ROSE has full knowledge of these languages, it can be used to optimize code performance and find errors. For example, millions of lines can be scanned for source-code defects (bugs that are allowed by the programming language but still cause system failures when run) or malicious elements hiding in the code. Additionally, ROSE returns these improvements to the user in revised source code rather than in the form of machine-readable binaries. The ROSE development team won an R&D 100 Award for this major computing breakthrough. Initial efforts for this work were funded by NNSA's Advanced Simulation and Computing Program and DOE's Office of Advanced Scientific Computing Research. Further funding was provided by Livermore's Laboratory Directed Research and Development Program.

## Strong Tools, Easy to Use

ROSE is unique on several fronts. Only ROSE offers such a rich set of analysis, debugging, and transformation capabilities that even novice software developers can build their own expert tools. "What ROSE does is convert the user's source code into an intermediate representation to encapsulate complex compiler knowledge in a form that can easily be accessed and manipulated


Development team for ROSE (from left): Peter Colllingbourne, Martin Bauer (formerly of Livermore), Thomas Heller (formerly of Livermore), Robb Matzke, David Hamilton, Chunhua Liao, Andreas Saebjornsen, Daniel Quinlan, Jeffrey Keasler, and Thomas Panas.

## More Accessible

With the open-source ROSE compiler infrastructure, computer programmers and software developers can build their own debugging, analysis, optimization, and transformation tools to quickly, efficiently, and inexpensively improve and compile their source code.

by nonexperts," explains Dan Quinlan, who led the project. "The developer can then build customized tools using ROSE to analyze that code for bugs, security problems, inefficiency, or poor performance—basically, any glitches an expert might find during manual inspection but in a much shorter time frame."

Beyond this, more advanced ROSE users can create compilerlike tools specific to their individual needs with an easy interface. For example, when a program fails and no reason is given, a developer may not quickly find the cause in the source code. Using ROSE, however, a developer could write a tool that adds itself to the source code to perform self-diagnosis. "That is, a developer could annotate the source code with just a few lines so that when the program fails, it fails with a graceful error message, relaying exactly what went wrong and where it went wrong," says Thomas Panas, a member of the ROSE team.

The ability to directly optimize one's own source code is especially significant. "All modifications to a user's code are performed through an easy interface," says Quinlan, "so a user never needs to touch or understand the underlying sophisticated compiler technology that makes the application work. The user puts in source code, ROSE works on it in the context of the same source code, and then ROSE generates back improved, compiler-ready source code."

This new source-to-source capability is a strong draw to ROSE users. It allows them not only to be in control of their programs but also to develop and apply various cutting-edge optimizations regardless of platform. Without this tool, developers are forced to rely on compilers to optimize their code or have the experts do it for them.

### Free and Easy: Open for Business

The open-source ROSE compiler software is available to use at no cost on the project's Web site (www.rosecompiler.org). Software developers can quickly leverage compiler technologies to build and perfect their own tools, allowing them to debug, optimize, and transform their source code as needed. In addition, the Web site content is continually updated, and automated messages are distributed to users so that the most recent versions of ROSE are available. Enhancements for ROSE developed by collaborators are added to the software for others to use, and ROSE tests itself robustly each night.

The documentation on the Web site includes an installation guide, a developer's guide, a user manual, and nearly 5,000 HTML pages of programmers' references. An extensive tutorial provides users with more than 50 examples of application scenarios. "Compiler expertise is continually added into the infrastructure," says team member Chunhua Liao. "It is a truly open infrastructure, one with an excellent development, testing, release, bug-tracking, and collaborative environment."

ROSE has already shown a strong ability to leverage the power of supercomputing technology. Future applications of ROSE are limited only by the imagination of developers using the software's public interfaces.

*—Jason Carpenter*

**Key Words:** binary analysis compiler, compiler infrastructure, program analysis, R&D 100 Award, ROSE, source code optimization, source code transformation, source code translation, source-to-source.

***For further information contact Dan Quinlan (925) 423-2668 (quinlan1@llnl.gov).***