Towards a tighter coupling of the compiler with applications, runtimes and the operating system.

Susan Eggers. University of Washington
Peter Lee. Carnegie Mellon University
Steve Lucco. Carnegie Mellon University
David Wood. University of Wisconsin
Larry Peterson. University of Arizona
Calton Pu. Oregon Graduate Institute
Joel Saltz. University of Maryland
(moderator: Brian Bershad. University of Washington)

The Question

- If compilers can make programs run better,
- and operating systems and runtimes can make programs run worse,
- can compilers be used to make operating systems and runtimes run better?
- "better" is measured according to
  - performance
  - portability
  - interoperability
  - reasonability
  - programmability
  - extensibility and reconfigurability

The Answer

- Yes.

What are we doing?

- Making operating systems go faster
  - Operating system specific compiler design and optimization
  - Dynamic specialization of operating systems using partial evaluation
  - Dynamic compilation for common predicate elimination and data specialization.
- Simplifying hardware requirements
  - Compiler support for software MMU
- Improving program and programmer productivity
  - Advanced language and compiler design for safety, modularity and resource management
- Parallel processing
  - Compiler, runtime and operating system support for irregular parallel problem optimizations
  - System support for portable parallel programs, runtime systems and compilers.

The Hidden Messages

- Some problems are "easy" and we just need to get good solutions out there
  - calling conventions
  - stub generators
- Some problems are "hard" and will require analysis and experimentation
  - dynamic compilation
- Not all programming languages start with F.
- Typesafety can be good for everyone.
- GCC is not a very good compiler infrastructure.