Tony Kennedy
Edinburgh University

Software Challenges in Computational Science

How can we write scientific programs that satisfy the computer scientists' goals of portability, reusable, reliability, correctness, and modularity together with the computational scientists' need for efficiency (especially on massively parallel multi-core architectures). There has been a lot of work on this in the lattice field theory community as part of the DOE SciDAC initiative, but it would clearly make sense for there to be a more general framework for expressing numerical algorithms. There are several significant issues, including abstraction of mathematical structures (such as linear spaces), memory management, data layout, and data exchange (XML schema and the like). Given the effort spent in designing and building supercomputers for QCD, physicists have been loath the lose even a few percent efficiency in exchange for the manifest advantages of portable software, and thus they have been highly motivated to look for software frameworks that achieve both.

In the UK we are setting up a new project with the goal of bridging the gap between new algorithms created by Numerical Analysts and efficient practical implementations of them on modern architectures. There are two principal hurdles to be overcome in order to solve this problem, a technical one of designing the right structures/interfaces/annotations that capture the information necessary for compilers to generate efficient codes on a wide range of architecture, and a "sociological" one of getting these accepted by users and implemented on a wide range of platforms so that they are used in real applications. This will require the joint efforts of Numerical Analysts, Computer Scientists, application experts, and supercomputer vendors, and support from funding agencies to overcome the extra risks of using new software approaches in leading-edge computational projects. Furthermore international collaboration in this effort is of vital importance.