Grand Challenges Conference Application

Two "Grand Challenges" in Computer Systems Research

Richard C. Waters
President and CEO of MERL

The call for participation could mean many different things by its use of the term "systems." I favor "Complete Application Systems" that may combine software with special hardware as the pragmatically most important thing the call for participation could be meaning. This is to be contrasted with research on file systems, operating systems, networking and the like that do not impinge so directly on users.

Since the beginning of the field, the dominant engine of progress in application systems has undoubtedly been improvements in hardware not software. The capability of the software in applications has been propelled forward like flotsam on a tidal wave of hardware improvement.

By highlighting "miniaturization, micro-electro-mechanical devices, nano-technology, and wireless communications" the call for participation plays into the frame of mind that hardware changes will remain the dominant force for change in our field.

Hardware improvement is a delightful force for change to be sure, and it may well remain the dominant force for many years to come; however, responding to these improvements is not the only source of grand challenges. In my mind, perhaps the greatest challenges are those that would remain even if hardware changes were to stop tomorrow.

Two huge challenges stand out particularly strongly in my mind. They are not trendy new challenges, but they are essential and far from solved. I strongly doubt they can be solved without revolutionary progress.

Challenge 1: True Man-Machine Co-Operation

There are two things meant by this challenge. First that a system would truly cooperate with the user and second that the user could truly cooperate with it.

In the first meaning, the system would be supportive, helping the user toward a common goal. It would be tolerant of errors and support negotiation and successive approximation to lead toward a successful result. The interaction would feel like walking down the middle of a valley where the terrain naturally guides you back on the path after any missteps.

In marked contrast, interacting with the typical system today is like playing an adversarial game where you are trying to seek out hidden clues to navigate a winding and complex path to success. At best, it feels like you are trying to stay on the knife edge of a ridge in the fog.

In the second meaning, the main outlines of the internal operation of the system would be clear and there would be ways for the user to advise and direct this internal operation--for example, to inject human insight into an optimization problem.

In contrast, almost without exception, current systems do what they do without any explanation or chance for intervention. These systems have no introspective ability with regard to what they are doing, let alone any ability to communicate with the user about their internal operations.

Making significant steps toward this challenge could be the dawning of a new age of computing, in which computers are much more supportive of people than they currently are.

Meeting this challenge will require major steps in the ability of programs to represent what they do and why they are doing it. Even more, it will require an ability to represent the larger task that the program in involved in and the "common sense" knowledge associated with this task. It will require the ability to do basic reasoning about what the user is doing in the context of this knowledge.

It will also require new and innovative perceptual & multi-modal interfaces, e.g., featuring spoken interaction, computer vision, and novel input devices. It could include multiple people interacting with a system via shared display group-ware or simple but extremely convenient and robust interfaces to embedded devices.

Naively, this could of course encompass all of Artificial Intelligence. However, given sufficiently narrow domains, great progress could be made toward this challenge without having to make a system that can, in general, "think like a human."

The fact that this is a long-standing challenge is amply illustrated by the fact that the main features of the challenge were ably put forward by J.C.R. Licklider, in his 1960 paper "Man-Computer Symbiosis," [IRE (now IEEE) Transactions on Human Factors in Electronics, Volume HFE-1, pp. 4-11].

But this challenge is neither stale nor hopeless. It is in my mind THE principle challenge of computer science; and progress (albeit slow) is being made. In particular, it is a major focus of our work at MERL (e.g., Collagen, Spoken Interfaces, Human-Guided Tabu Search, ...) see "www.merl.com".

Challenge 2: Much Higher Quality Systems

There are a lot of components to quality. The first challenge above addresses a central issue of the quality of interaction. This challenge address a more mundane aspect.

So perhaps it is a bit hard to figure out what a system does and perhaps it is a bit hard to interact with the system to indicate what you want, but in the final analysis once you do your part, does the system do its part, or is it brittle and buggy?

Does the software have a "warranty" something like "Manufacturer disclaims all warranties with regard to this software, including all implied warranties of merchantability and fitness for any purpose ...". Paper clips have implied warranties that are better than that.

It is an old saw that nobody is prepared to pay for quality in Software. Perhaps not. But certainly it is an essential challenge to our field to reduce the cost penalty for improved quality.

Fundamental systems research is needed to yield large, more generalized, more flexible, components from which robust systems can flexibly be created. The kind of introspective and task-knowledge capabilities needed for the first challenge above can also help make systems less brittle.

It will be delightful to discuss challenges never before discussed, but any discussion of the key challenges facing computer systems research would be remiss to omit the challenges above.

Short Biography

As president and CEO of MERL, Dr. Richard C. (Dick) Waters oversees Mitsubishi Electric's three North American research laboratories. As in any research organization, his central challenge is creating a culture that leads to results that are both scientifically significant and of significant economic benefit to MERL's parent company Mitsubishi Electric.

Dr. Waters joined MERL's Cambridge (MA) research lab at its inception in 1991. From then until 1998, Dr. Waters worked with an interdisciplinary team of researchers on multi-user interactive environments for work, learning and play. This led to the creation of two principal demonstrations: The Electronic Meeting Place (1994) and Diamond Park (1995). A key result of the research was the development of the Scalable Platform for Large Interactive Networked Environments (SPLINE), which can be used to support environments like Diamond Park. In recognition of this work, Dr. Waters became MERL's first research fellow in 1996. In 1998 he become director of the Cambridge research lab and was promoted to president of MERL as a whole in 1999.

From 1978 to 1991, Dr. Waters was a research scientist at the MIT AI laboratory. During that time, he was co-principal investigator of the Programmer's Apprentice project. The goal of this project was the creation of knowledge-based tools for the formalization of software requirements and for program creation, editing, and analysis. He was also the developer of Series Expressions, which allow looping computations to be expressed as functional compositions without loss of efficiency.

A life-long resident of New England, Dr. Waters received his B.S. degree from Brown University in 1972, an M.S. from Harvard in 1973, and his Ph.D. from the MIT AI Lab with a minor concentration in Linguistics in 1978. He, his wife, and their four children enjoy the outdoors and learning new things.