Grand Challenges

Eric Brewer, May 2002

1) **Simplifying the development of complex systems**. Today it remains extremely difficult to build robust systems that interact with the environment in any significant way. This includes embedded systems, networked systems and large-scale Internet services. I believe the way out if through the combination of compiler technology, *less* expressive languages (perhaps domain specific), and some forms of formal methods such as model checking. Early results show that we can prevent buffer overruns through compile-time detection; this single class of errors is responsible for over 50% of security problems on the Internet, so prevention has huge impact. We also believe a similar approach can prevent race conditions in embedded systems (and other synchronization bugs), which would have prevent such classic failures as the Therac-25 and the Mars rover. Today systems are not really designed to make use of the full range of compile-time analysis (and thus prevention) that is possible.

2) **A new architecture for data**. The traditional architecture, the relational database, has served us well, but is far from perfect and is frequently misused due to the lack of alternatives. Example applications for which the traditional approach is a poor fit include: bioinformatics (like the human genome), search engines, scientific data with error distributions (which is pretty much all of it), and semi-structured data in the style of XML. The new architecture must deal directly with statistics and error propagation, data transformation and workflow, combining structured and unstructured data, schema evolution, and federation (combining partially distrusting databases).

3) **A movement toward "flow" instead of computation**. Currently computer science is based on computation not on flow, but I believe that in most cases the flow of data is the harder problem. Some examples: chips are limited by wiring (flow) not by transistors (computation); the Internet is limited by flow properties such as QoS and bandwidth (not by the PC at the endpoint); information overload is flow problem; databases are limited by I/O not by computation but the programming model (threads) is geared toward computation; event-driven systems (flow oriented) are widely used for giant servers and embedded systems, but are poorly supported and not well understood.

Dr. Eric A. Brewer

Dr. Eric A. Brewer is an associate professor of Computer Science at UC Berkeley, where he has led projects on scalable servers, wireless networking, giant-scale Internet-based services, large collections of tiny networked devices, and security. He received his Ph.D. in EECS from MIT in 1994.

Dr. Brewer is also the Co-Founder and Chief Scientist of Inktomi Corporation, now listed on the Nasdaq stock exchange.

In 2000, Brewer founded the Federal Search Foundation to improve consumer access to government information. Working with President Clinton, Dr. Brewer helped to create FirstGov.gov, the official portal of the Federal government, which now contains over 50 million searchable documents, and represents the cornerstone of US online government. It also now includes information for state governments.

Major awards: He was named a "Global Leader for Tomorrow" by the World Economic Forum, by the *Industry Standard* as the "most influential person on the architecture of the Internet", and by *Forbes* as one of 12 "e-mavericks", appearing on the cover in 1999. *InfoWorld* named him one of their top ten innovators in the information revolution. *Technology Review* named him as one of the top 100 most influential people for the 21st century (the "TR100"). Fellowships include the Sloan Foundation, the Okawa Foundation, DARPA and the Office of Naval Research.