Mobile Computing

M. Satyanarayanan
School of Computer Science
Carnegie Mellon University

In his book ‘‘Mind Children’’, my colleague Hans Moravec draws an analogy between the seminal role of mobility in the evolution of biological species, and its influence on the capabilities of computing systems [3]. Although Hans’ comments are directed at robotic systems, I believe that his observation applies equally well to a much broader class of distributed computing systems involving mobile elements. Mobility will influence the evolution of distributed systems in ways that we can only dimly perceive at the present time.

The recent proliferation of portable computers, in conjunction with nascent high- and low-bandwidth cordless networking technology, will soon provide a pervasive hardware base for mobile computing. The stage is set for hand-held and wearable computing devices of every variety imaginable. Computing will truly be ubiquitous, and no longer an activity restricted to a person’s desk. What new challenges will this pose?

Mobile computing systems are constrained in important ways, relative to static systems. These constraints are intrinsic to mobility, and are not just artifacts of current technology:

- *Mobile elements are resource-poor relative to static elements.* At any given cost and level of technology, considerations of weight, power, size and ergonomics will render mobile elements less computationally capable than their static counterparts. While mobile elements will undoubtedly improve in absolute ability, they will always be at a disadvantage relative to static elements.
- *Mobile elements are more prone to loss, destruction, and subversion than static elements.* A Wall Street stockbroker is more likely to be mugged on the streets of Manhattan and have his or her laptop stolen than to have the workstation in a locked office be physically subverted. Even if security isn’t a problem, portable computers are more vulnerable to loss or damage.
- *Mobile elements must operate under a much broader range of networking conditions.* A desktop workstation can typically rely on LAN or WAN connectivity. A laptop in a hotel room may only have modem or ISDN connectivity. Outdoors, a laptop with a cellular modem may find itself in intermittent contact with its nearest cell.

These constraints violate many of the assumptions upon which today’s distributed systems are based. Further, the ubiquity of portable computers will result in mobile computing systems that are much larger than the distributed systems of today. *Scalability* will thus be a continuing concern.

A key requirement of mobile computing systems will be the ability to access critical data regardless of location. Data from shared file systems and databases must be made available to programs running on mobile computers. For example, a technician servicing a jet engine on a parked aircraft needs access to engineering details of that model of engine as well as past repair records of that specific engine. Similarly, a businessman who is continuing his work on the train home from Manhattan needs access to his business records. Yet another example involves emergency medical response to a case of poisoning: the responding personnel will need rapid access to medical databases describing poison symptoms and antidotes, as well as access to the specific patient’s medical records to determine drug sensitivity.

The requirement of access to shared data implies interdependence between the elements of a mobile computing system. At the same time, the need to be robust against network and remote site failures requires clients to be as autonomous as possible. Thus mobility exacerbates the the tension between *autonomy* and *interdependence* that is characteristic of distributed computing.

Ideally, mobility should be completely *transparent* to users. Transparency relieves users of the need to be constantly aware of the details of their computing environment, thus allowing them to focus on the real tasks at hand. The adaptation necessary to cope with the changing environment should be initiated by the system rather than by users. Of course, perfect transparency is an unattainable ideal. But that should not deter us from exploring techniques that enable us to come as close as possible to the ideal.

The Coda File System, built by my research group at Carnegie Mellon University, represents an initial step in providing such transparency [1, 4]. Coda facilitates the use of shared data in mobile computers by simplifying pre-caching of files, allowing autonomous operation while *disconnected*, and transparently *reintegrating* changes upon reconnection. Work is under way to extend the system to exploit *weak* (i.e., low-bandwidth or intermittent) connectivity when available.

The design of Coda extends the file access paradigm of the *Andrew File System* (AFS) [5] to mobile environments. Coda completely hides mobility from applications and users. An application cannot tell whether it is operating connected or disconnected. This has the great advantage that Coda is binary compatible with applications written for the BSD Unix interface. For an important class of common applications, full transparency has indeed worked out well.

But the strategy of insulating applications from mobility can only be pushed so far. Supporting more ambitious applications will require controlled exposure of mobility to allow these applications to modify their behavior in application-specific ways to cope with the vagaries of a mobile communication environment.

Consider, for example, an application capable of displaying stored full-motion video images. Below a certain bandwidth and network quality, it will not be possible for the application to display the images in full-motion color. Extensive compression will help, but cannot solve the problem completely. But if the application is also capable of displaying the image in slow-scan black and white, it could automatically do so when the bandwidth fell below a critical threshold. Thus sophisticated applications will be able to sense and react to mobility in an application-specific manner, while minimizing user involvement. This in turn will allow the best level of service achievable at the current physical location of a mobile client to be offered to users.

As another example, consider the resolution of conflicting updates made to shared data. Such conflicts may arise if an optimistic replication strategy is used to improve data availability in failure-prone mobile networks. Since resolution is an application-specific concept, masking such conflicts from users will require the system to invoke application-specific code to perform the resolution [2].

Since different applications need to adapt differently to mobility, *extensibility* will be an important requirement for effective support of applications in mobile computing environments. Techniques that allow operating system functions such as caching and network transmission to be customized on an application-specific basis thus assume importance. The challenge is to design and implement such extensibility without compromising efficiency.

[1] Kistler, J.J., Satyanarayanan, M.
Disconnected Operation in the Coda File System.
ACM Transactions on Computer Systems 10(1), February, 1992.

[2] Kumar, P., Satyanarayanan, M.
Supporting Application-Specific Resolution in an Optimistically Replicated File System.
In *Proceedings of the 4th IEEE Workshop on Workstation Operating Systems*. Napa, CA, October, 1993.

[3] Moravec, H.
Mind Children.
Harvard University Press, Cambridge, MA, 1988.

[4] Satyanarayanan, M., Kistler, J.J., Kumar, P., Okasaki, M.E., Siegel, E.H., Steere, D.C.
Coda: A Highly Available File System for a Distributed Workstation Environment.
IEEE Transactions on Computers 39(4), April, 1990.

[5] Satyanarayanan, M.
Scalable, Secure, and Highly Available Distributed File Access.
IEEE Computer 23(5), May, 1990.