

Issues in Wireless Mobile Computing

Dan Duchamp
Computer Science Department
Columbia University
New York, NY 10027
djd@cs.columbia.edu

Position statement: wireless mobile computing might become the dominant form of computing in the next decade, and proper support for it will require re-thinking several aspects of systems software design.

Workstation-class computers have lately become small enough to carry, and portable computers have become by far the fastest growing segment of the computer industry. In addition, there is much activity in the telecommunications industry aimed at providing wireless data networks¹ which would lie at the fringes of a universal infrastructure of high speed WANs and LANs, providing the “last link” to a mobile user. These trends have converged in a number of recently announced products — portable computers with interfaces to wireless network technologies. It is possible that within the decade a vast proportion of all computers will be personal, wireless, mobile workstations.

1. Is Research Needed?

Assuming the world becomes populated with substantial numbers of wireless mobile computers, will any new problems arise in the design of systems software? If we are willing to keep our expectations sufficiently low, then the answer is no.

One can obtain “wireless mobile computing” by putting a cellular telephone interface into a portable workstation and running a protocol like SLIP over a dialup IP service. Such an approach allows an Internet site to operate anywhere within the domain of a single cellular telephone service, and is simple, cheap,² and available right now. Given the limited bandwidth of cellular telephone networks, a sensible approach would be to operate the portable workstation as an X terminal. Currently, several companies sell a poor man’s version of this — the typical “workstation” is a small alphanumeric pager-like device which limits the user to reading and composing Email. Such an approach has its uses; however, there are some fundamental problems with the idea of building the world entirely with mobile X terminals.

One problem is the very limited cellular bandwidth and its inefficient allocation by a circuit-switched, TDMA or FDMA phone network operating with a large cell size. Severe bandwidth constraints limit the number of simultaneous users, the nature of the applications that can be run, or both. (Presently in many areas callers are regularly denied service because no channel is available.) The present Email services may represent the limit of what can be done with analog phone service designed for voice. Another, hopefully temporary, problem with building on cellular telephone systems is that different systems are not always compatible. Further arguments against the X terminal approach appear below.

The situation is worse for mobile computing in which the user is not always attached to the Internet via some wireless link. In this case, a workstation disconnects and later reconnects to the network. Current software is not well suited for this type of operation. First, there is the difficult problem of how to operate while disconnected; the most advanced work to address this problem is perhaps Coda [5]. Second, there

¹The technologies most often mentioned are radio (RF) and infrared (IR).

²Except for usage charges, which are initially high in order to defray costs of development and deployment.

is the question of how to eliminate the need for reinitialization of the system or individual applications after a relocation has taken place; two proposals that address this issue are [4] and [12]. Both papers explain how a workstation can move from one IP network to another without necessarily interrupting higher level (i.e., TCP) connections.

Another concern with using existing software (either across wireless links or in disconnect/reconnect mode) is how to provide adequate response to a user who wanders a substantial distance from his/her data and cycle server(s). As communication overhead is engineered to become smaller, latency becomes more and more a direct function of distance. Current distributed systems address the latency issue by converting terminals to workstations, outfitting them with substantial local compute and caching abilities. The extent to which portable workstations can continue to depend on large caches is debatable. Rapidly increasing storage density is often assumed to render most workloads fully cachable eventually, if not already. But this trend must compete against two opposing trends as well as the possibility that a wider variety of users will generate workloads that are not so easily dealt with. The first opposing trend is that portable workstations are shrinking, perhaps down to very small sizes. The second is the movement toward multimedia user interfaces, which threatens a drastic increase in a workstation's appetite for data.

The design and use of future wireless mobile workstations may or may not permit caching to be, by itself, the solution to response time concerns. This point is discussed further in Section 2.2. In Section 3 I will outline, for both possible scenarios, what I see as the key research issues in systems software design.

2. What is the Goal?

Wireless mobile computing is a somewhat hazardous area in which to conduct software research, because it is not yet clear what hardware base we will be designing for or what workloads users will present.

Many *special-purpose* "wireless mobile computing" systems already exist. The combination keyboard/printer on the belts of car-return attendants is an example of a wireless remote-terminal system with custom hardware good for only one task. However, a goal that is more interesting and more generally useful is how to provide *general-purpose* wireless mobile computing. The remainder of Section 2 explores which goal we should pursue.

2.1. What Will the Hardware Be?

What sort of hardware base should we expect? I see two questions as especially important:

1. What do you carry and why do you carry it? Or: is the portable device a "terminal" or a "computer"?
2. What is the nature of the network connection? Or: are you disconnected often or rarely?

These two axes — which are not orthogonal — begin to define a design space.

There are two primary arguments for carrying a "terminal." The first is that it is desirable to lower the cost and size of what the user carries. The second is that the bandwidth needed to support modern user interfaces is quite limited,³ and therefore terminal-style computing is best for running over wireless links of limited bandwidth. These are good arguments.

There are several arguments against the terminal model. First, terminals are useless to the same extent

³An example: NCD's XRemote implementation of the X protocol runs over 9600 baud.

that the network is unstable or unavailable. Second, it is not clear why a portable X terminal must cost much less than a portable workstation. With current desktop models, the difference is that workstations have more memory, sometimes have non-volatile storage, usually have room for expansion, and are more likely to use a CPU chip that isn't yet a commodity item. But if terminals and workstations are both packaged as "laptops" or "handhelds" of limited size, weight, and power consumption, then the difference may boil down only to whether non-volatile storage is present. A third argument is that workstation-based systems scale better. Furthermore, even if X terminals are cheaper, the cost argument is specious: behind an X terminal must be a server, and someone must buy a fraction of that server for each user. Perhaps the overall system price/performance (or even just the price!) is optimized with mobile computers rather than with mobile terminals. Finally, as discussed above, endowing the mobile unit with substantial compute and storage capabilities better facilitates the implementation of multimedia user interfaces despite limited network bandwidth.

2.2. Who Are We Serving?

To serve persons who generate a workload akin to that for which most UNIX-like systems are implicitly targeted, I think it will work quite well to build terminal-based systems or else computer-based systems designed by tweaking existing approaches and riding technology waves. My view of this workload is that it is typified by text-oriented applications, small and fairly predictable file working sets, and use primarily within an office or campus of limited geographical extent.

This workload is an easy one to accommodate, and technology trends will make accommodating it even easier. For example, if working sets remain within a few megabytes, then soon a single DRAM chip will be big enough. Another example is that the signal propagation characteristics of optical fiber will allow optical LANs to effectively span much greater distances than is now done with technology like bridged Ethernets. The result will be an infrastructure providing stable, low-latency, high-bandwidth data movement over considerably wider geographical areas than those in which we now enjoy such service.

But wireless mobile computing systems could be the first systems we design for people other than ourselves. So far in the history of computing we have designed systems, consciously or not, mostly for use by ourselves. This strategy, seemingly so unwise, has actually been quite successful. I think the reason is that we have in the past been the vanguard of "information workers." As other professions followed ours in converting to information-based jobs, we often had the right technology ready or within reach of being ready for them.

There are many enduring examples of jobs that require movement over a wide area in order to deliver things or to visit immovable people or things. Many such jobs are now thought of as low-skill and/or labor-intensive. It is possible that in many cases such jobs could benefit from information technology, in which case mobile computing will *cause* these jobs to metamorphosize into more information-based tasks. Examples include repair personnel, nurses, and inspectors. Repair personnel, to perform their jobs, should have access to manuals, service records, drawings, animations, and so on. Doctors and nurses move among within and among clinics and hospitals to visit confined patients, and may need the same sorts of information. Inventory managers and inspectors may carry computers that they use not only for record-keeping but also as a sensing device (e.g., camera). Furthermore, many service-industry workers really belong at their clients' offices, not at their office; reference [1] describes accountants who set up networks at their clients' offices and whose database working set is too large for CD-ROM.

Note that in some of these examples the method of information use is to access a very large database, often in semi-random fashion. These people might often have working sets too large and unpredictable to be adequately addressed by cache-oriented techniques.

2.3. Conclusion

Both terminal-based and workstation-based systems will exist in the future, just as they do now. But I believe that in the future there will be many more and more different kinds of information workers than there are now, and that the flexibility of the workstation-based model will make it the more desirable platform. For the reasons foreseen above, for the reason that more flexible systems will cope better with foreseen uses, and for the reason that general-purpose, computer-based systems present the more interesting research agenda, the rest of my position statement assumes that the goal is to learn how to build general-purpose, computer-based wireless mobile computing systems.

3. Major Challenges

Taking the viewpoint that many users will be carrying computers, I see four areas of new or increased challenges in system software design:

1. Designing distributed services to support the mobile user.
2. Adjusting to the unique characteristics of new technologies that might be used in future portable workstations.
3. Building user interfaces for computers that may be arbitrarily small (i.e., walkman-size, pocket-size, ring-size).
4. Providing authentication, accounting, and management over a wide area and across organizations. This problem is not new — it is part of the problem of designing world-scale computing systems — but seems likely to be exacerbated and made more urgent by the advent of mobile computers.

A final challenge will be to do all this, at least initially, while retaining a reasonable level of compatibility. That is, to provide applications with unchanged interfaces and performance so that a faithful copy of a user's desktop computing environment is available in his/her hand without the need to re-write applications.

Sketches of possible solutions to the first two problem areas are provided below.

3.1. Services for the Mobile User

The exact nature of the challenges and the solutions depend on the model of operation. Following the discussion above, I postulate two models of operation: the *caching* model, and the *remote access* model.

In the caching model, it is presumed that working sets are smaller than practical cache sizes, and nearly everything can be loaded into a user's cache. In this model, the central challenges will include:

- Providing TCP connections that persistent even as a mobile host moves from network to network.
- Cross-domain authentication.
- Compression techniques (and, possibly, new types of protocols) to conserve bandwidth on wireless links.
- Techniques for coping with the fact that a mobile computer, as it moves, will encounter varied computing environments — meaning, for example, different sets of devices in different locales.
- More dynamic approaches to load balancing.
- Building client/server systems so that either side can disengage without harm. This type of design would help with disconnected operation and load balancing.

These range from small engineering efforts (e.g., compression) or nearly-solved problems (e.g., support for cross-realm authentication is already built into Kerberos) to major open areas (i.e., the last point).

In the remote access model, it is presumed that not all data can be cached, either because there is too much data or because the data is shared and cache consistency mechanisms regularly remove data from caches. In these cases, what to do is not so obvious. At Columbia, we are investigating the design of so-called “reconfigurable services” — services that can reconfigure themselves to continually provide a service point near to a mobile client. The basic idea is that the reconfigurable service provides a larger second-level cache that follows the user around.

We have designed and are implementing several reconfigurable services: a file replica management algorithm [9, 11]; an adaptive algorithm for paging from a mobile computer into the memory of the “closest” paging server [8];⁴ and X window display [3]. Each design took advantage of the unique aspects of the service to accomplish reconfiguration in the most “natural” way. Space limitations prevent describing these projects with enough detail to make them understandable.

3.1.1. Mobile Internetworking

In any model, the first order of business is to provide the mobile computer with the ability to move without having to reinitialize. The work described in [4] consists of two protocols that permit a computer to move from one IP network to another in a manner that is transparent above IP.

The gist of the solution is to, first, give mobile computers an address on a special “virtual subnet” (i.e., one that does not exist), and, second, add special routers that serve as gateways between the virtual subnet and the rest of the world. Of course all these computers must have locations on the physical Internet. The special routers form a distributed database that tracks the mapping of each mobile computer on the virtual subnet to its present location in the physical Internet. Each network supporting mobile computers must have one of these special routers which stores part of the database (specifically, it records which mobile computers are presently on its network) and which is the first (last) gateway for packets from (to) the mobile computer to (from) any computer on another network.

Initial location of a mobile computer is done by multicasting among these gateways, and so this solution scales only to the “campus” level. To relocate on another campus, a mobile depends on forwarding packets through its “home base.”

3.2. Adjusting to New Technologies

Two examples of new technologies that might be employed in wireless mobile computing systems are wireless networks links and solid-state non-volatile storage. The former seems a certainty, the latter a speculation.

To cope with the limited bandwidth of wireless LANs, we have developed a file prefetching algorithm that is often successful in looking tens of references ahead [10]. The motivation for prefetching is that slow links might not be able to satisfy bursty demands for data and therefore it is necessary to “smooth out” the flow of data to the mobile workstation. This will be especially true as processors increase in speed, and so the prefetching algorithm is designed to trade client-side cycles in return for greater look-ahead.

The essence of the algorithm is that `fork`, `exec`, and `open` systems calls are traced during execution; they tend to form “trees”⁵ whose nodes are files and whose edges are system calls. Each such tree can

⁴The latter is an example of the growing trend towards using remote memory [7, 2, 6].

⁵The actual pattern is a graph, but node replication allows formation of a tree.

be interpreted as defining a computation. A quick heuristic form of tree-matching is used to match the currently forming tree with trees saved from the past. A sufficiently good match initiates prefetching of the files in the remainder of the saved tree.

There are several arguments for studying algorithms for the management of *solid-state* non-volatile storage. First, non-volatile storage is useful in workstations as a file caching medium, especially if the workstation might not always be able to contact the file server. Further, rotating disks are probably not appropriate in small computers because of their size, weight, power consumption, and lack of ruggedness. Finally, battery-refreshed DRAM might be too costly or power-hungry. At Columbia we are presently undertaking a study of how to manage the flow of file data among remote servers, workstation DRAM, and workstation “flash EPROM.” Flash EPROM is a dense, non-volatile, solid-state technology with a read latency close to that of DRAM, a write latency close to that of disk, and which can withstand only a limited number of writes over its lifetime. Flash EPROM, currently quite expensive, carries the promise of being cheaper than DRAM once the technology becomes mature.

4. Summary and List of Issues for Discussion

Wireless mobile computing seems very likely to be a reality. Future computing systems based on such technology will take a number of forms. Systems consisting of very small mobile workstations capable of operating over wireless links present several new and important challenges in system software design.

Important questions include these:

- What is the nature of the portable device? Is it a terminal or a computer? What is its eventual size? Will it have a disk?
- If workstations become very small, what happens to the user interface, especially input?
- What is the nature of the network? Will wireless service be omnipresent? What will be the quality (i.e., stability, bandwidth) of wireless links?
- Who should we be designing systems for? What are characteristics of their movement and data access patterns?
- Reconfigurable services: necessity or crazy idea? Or: does wireless mobile computing really make important changes to the ground rules?
- Assuming the answer is yes, is the goal of insulating applications from the new computing environment a proper one, or should we “give up” and go straight to thinking about re-writing applications?

References

- [1] K. Coale.
Mobile Workers Need Connectivity.
InfoWorld 14(10):59-62, March 2, 1992.
- [2] D. Comer and J. Griffioen.
A New Design for Distributed Systems: The Remote Memory Model.
In *Proc. 1990 Summer USENIX Technical Conf.*, pages 127-135. USENIX, June, 1990.

- [3] D. Duchamp and A. Shamash.
X Window System Extensions Supporting Mobile Users.
August, 1991.
Submitted for publication.
- [4] J. Ioannidis, D. Duchamp, and G. Q. Maguire, Jr.
IP-based Protocols for Mobile Internetworking.
In *Proc. SIGCOMM '91*, pages 235-245. ACM, September, 1991.
- [5] J. J. Kistler and M. Satyanarayanan.
Disconnected Operation in the Coda File System.
In *Proc. Thirteenth ACM Symp. on Operating System Principles*, pages 213-225. ACM, October, 1991.
- [6] A. Leff, P. S. Yu, and J. L. Wolf.
Policies for Efficient Memory Utilization in a Remote Caching Architecture.
In *Proc. First Intl. Conf. on Parallel and Distributed Information Systems*, pages 198-207. IEEE, December, 1991.
- [7] M. N. Nelson, B. B. Welch, J. K. Ousterhout.
Caching in the Sprite Network File System.
ACM Trans. Computer Systems 6(1):134-154, February, 1988.
- [8] B. N. Schilit and D. Duchamp.
Adaptive Remote Paging for Mobile Computers.
Technical Report CUCS-004-91, Columbia Univ. Computer Science Dept., February, 1991.
Submitted for publication.
- [9] C. Tait and D. Duchamp.
Service Interface and Replica Consistency Algorithm for Mobile File System Clients.
In *Proc. First Intl. Conf. on Parallel and Distributed Information Systems*, pages 190-197. IEEE, December, 1991.
- [10] C. Tait and D. Duchamp.
Detection and Exploitation of File Working Sets.
In *Proc. Eleventh Intl. Conf. on Distributed Computing Systems*, pages 2-9. IEEE, May, 1991.
- [11] C. Tait and D. Duchamp.
An Efficient Variable Consistency Replicated File Service.
March, 1992.
Submitted for publication.
- [12] F. Teraoka, Y. Yokote, and M. Tokoro.
A Network Architecture Providing Host Migration Transparency.
In *Proc. SIGCOMM '91*, pages 209-220. ACM, September, 1991.