Mobile Agents and the Future of the Internet

David Kotz and Robert S. Gray

Department of Computer Science / Thayer School of Engineering
Dartmouth College
Hanover, New Hampshire 03755

dfk@cs.dartmouth.edu, robert.s.gray@dartmouth.edu

Abstract

Use of the Internet has exploded in recent years with the appearance of the World-Wide Web. In this paper, we show how current technological trends may lead to a system based substantially on mobile code, and in many cases, mobile agents. We discuss several technical and non-technical hurdles along the path to that eventuality. It seems likely that, within a few years, nearly all major Internet sites will be capable of hosting and willing to host some form of mobile code or mobile agents.

1 Introduction

Rapidly evolving network and computer technology, coupled with the exponential growth of the services and information available on the Internet, will soon bring us to the point where hundreds of millions of people will have fast, pervasive access to a phenomenal amount of information, through desktop machines at work, school and home, through televisions, phones, pagers, and car dashboards, from anywhere and everywhere. Mobile code, and in particular mobile agents, will be an essential tool for allowing such access.

Mobile agents are programs that can migrate from host to host in a network, at times and to places of their own choosing. The state of the running program is saved, transported to the new host, and restored, allowing the program to continue where it left off. Mobile-agent systems differ from process-migration systems in that the agents move when they choose, typically through a "jump" or "go" statement, whereas in a process-migration system the system decides when and where to move the running process (typically to balance CPU load). Mobile agents differ from "applets", which are programs downloaded as the result of a user action, then executed from beginning to end on one host.

Mobile agents are an effective choice for many applications, for several reasons [LO99], including improvements in latency and bandwidth of client-server applications and reducing vulnerability to network disconnection. Although not all applications will need mobile agents, many other applications will find mobile agents the most effective implementation technique for all or part of their tasks.

Although we believe that current trends in Internet technology and usage lead to the use of mobile agents, several technical and non-technical hurdles must be addressed along the way. These hurdles represent significant but not insurmountable challenges, so we predict that many major Internet sites will accept mobile agents within a few years. The goal of this paper is to spark discussion about how best to realize this optimistic, but reasonable, vision.

2 Trends

There are several trends affecting Internet technology and activity:

A substantially similar paper appeared as a position paper in the Workshop "Mobile Agents in the Context of Competition and Cooperation (MAC3)" at Autonomous Agents '99, on May 1, 1999.

In ACM Operating Systems Review, August 1999, pp. 7-13.
Available at ftp://ftp.cs.dartmouth.edu/pub/kotz/papers/kotz:future2.ps.Z.

Bandwidth. The telecommunications industry is laying down astonishing amounts of fiber. Although Internet traffic is growing exponentially, the bandwidth soon to be available on the Internet backbone, as well as to many offices and neighborhoods, is immense.

Nonetheless, bandwidth to many end users will remain limited by several technical factors. Many users will still connect via modem, or at best, ADSL over the old copper loop. Many other users will connect via low-bandwidth wireless networks. Most users can expect to see no more than 128 Kbps to 1 Mbps available at their desktop or palmtop, although some asymmetric cable modems may reach 10 Mbps (for downloads) [Gri99, DR99].

Perhaps more importantly, the *gap* between the low-bandwidth "edge" of the network, and the high-bandwidth "backbone" of the network, will increase dramatically as the backbone benefits from increased quality and availability of fiber, while the edge remains limited by the fundamentals of wireless and copper connections. We expect that this trend will continue even as local connections improve past 1 Mbps in the next few years, since backbone bandwidths are improving much faster than local bandwidths. (For example, Nortel has announced a 1.6 Tbps optical-fiber product that will be out in the year 2000 [Sch99].)

Mobile devices. One of the hottest areas of growth in the computer industry is portable computing devices. Everything from laptops to palmtops to electronic books, from cars to telephones to pagers, will access Internet services to accomplish user tasks, even if users have no idea that such access is taking place. Typically, these devices will have unreliable, low-bandwidth, high-latency telephone or wireless network connections.

Mobile users. Web-based email services[1] make it clear that users value the ability to access their email from any computer. Web terminals will become commonplace in public spaces, such as cafes, airports, and hotels. Eventually, particularly with the growth in bandwidth, users will have full access to all of their files and applications from any terminal. Despite this, mobile devices will proliferate unchecked, since just as with public phones, Web terminals will never be available *everywhere* that a user might find herself.

Intranets. Organizations are increasingly using Internet protocols, particularly HTTP, to build internal "intranets" for their own distributed-information needs. All access to an intranet is managed by a single organization. Thus, new technologies can be deployed quickly, since (1) little coordination is needed with outside organizations, and (2) security (*within* the intranet) is of less concern.

Information overload. Internet users are already overwhelmed by the sheer volume of available information, and the problem will get worse as the Internet grows. Search engines, shopbots, portals, collaborative filtering, and email filtering are existing technologies that allow the user to reduce the torrent to a manageable stream, but these technologies are still quite limited.

Customization. Unlike broadcast media, the Internet makes it possible to customize access for each user. Current technologies allow customization at both the client (browser) and the server. Many Web sites include their own site-specific customization features, but the customization is increasingly provided by third-party "proxy" sites.

Proxies. Such proxy sites, which today are most often Web sites such as the various shopbots, interpose between a user and one or more other Internet services. As a means to both reduce information overload and customize service access, proxy sites will become more and more important. In particular, as portable devices become more prevalent, highly specialized proxy sites will be provided to meet the special needs of mobile users.

3 Mobile agents are coming

The trends outlined in the previous section lead to the conclusion that mobile code, and mobile agents, will be a critical near-term part of the Internet. Why? Not because mobile code makes new applications possible, nor because it leads to dramatically better performance than (combinations of) traditional techniques, but rather because it provides a single, general framework in which distributed, information-oriented applications can be implemented efficiently and easily, with the programming burden spread evenly across information, middleware, and client providers. In other words, mobile code gives providers the time and flexibility to provide their users with *more* useful applications, each with *more* useful features. Our full argument roughly

[1] e.g., `http://www.hotmail.com/`.

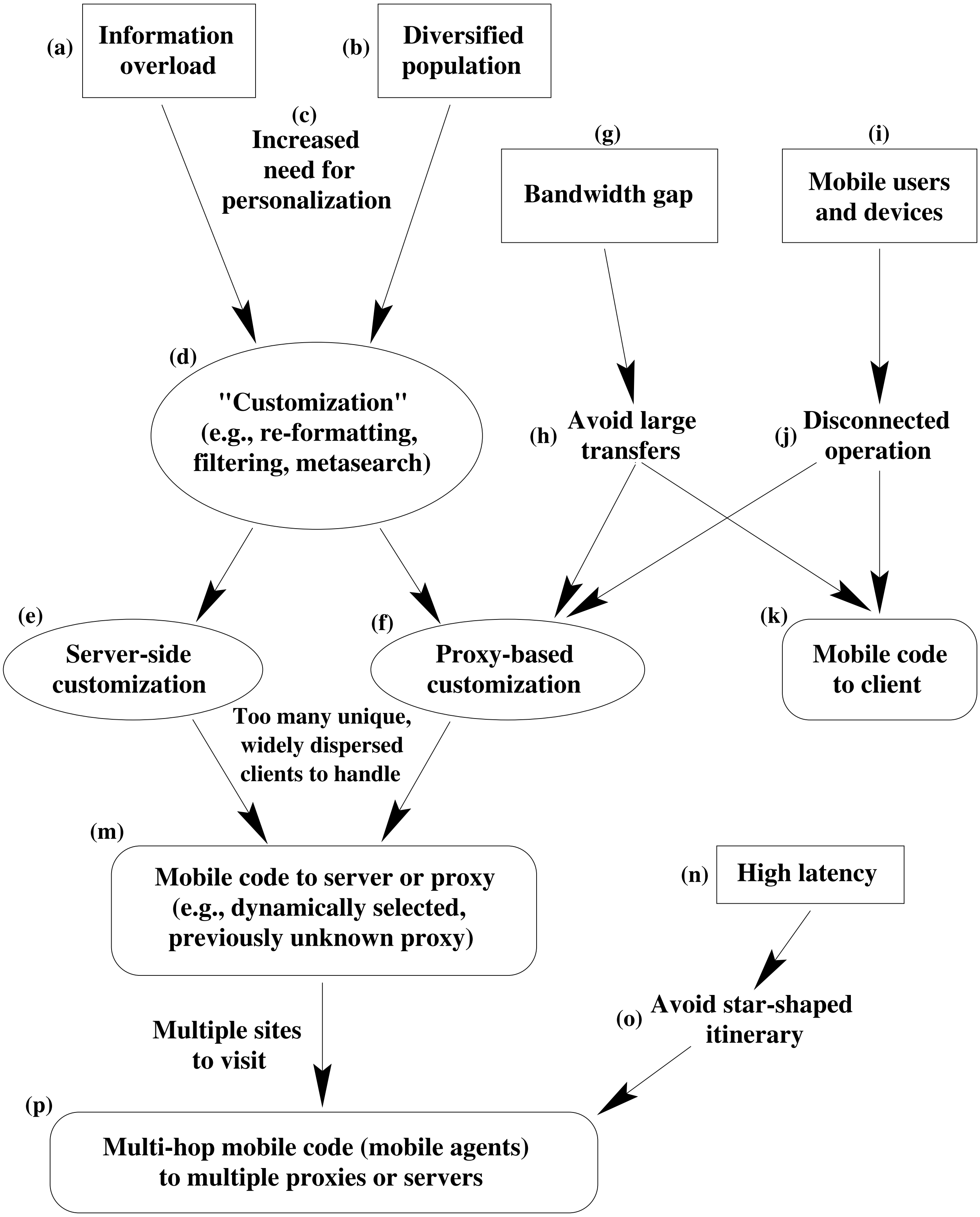

Figure 1: The trends leading to mobile agents

follows Figure 1.

Both the amount of information available on the Internet **(a)**, and the number and diversity of its users **(b)**, are growing rapidly. This diverse population of users will not settle for a uniform interface to the information, but will demand personalized presentations and access methods **(c)**. This personalization will range from different presentation formats to complex techniques for searching, filtering and organizing the vast quantities of information **(d)**. Today, such personalization facilities are provided at the information source in a site-specific manner **(e)**, at a proxy Web site **(f)**,[2] or (occasionally) as client software.[3]

Meanwhile, the network technology will lead to an increased gap in the bandwidth of the core Internet versus the fringes of the Internet **(g)**. Thus, most client hosts will shun large transfers of data **(h)**. That trend encourages the migration of application functionality from clients into proxy sites **(f)**, which are presumably better connected to the core Internet, and need send only the final results over the slower connection to the client. Furthermore, the dramatic availability of core bandwidth will allow these proxy sites to be aggressive in gathering, prefetching, and caching information on behalf of their clients.

Mobile users **(i)** will frequently disconnect from the network, and perhaps connect later at another location with poor bandwidth **(j)**. This tendency again leads to the use of proxies **(f)**. It also encourages application programmers to choose a mobile-code solution to dynamically install the necessary client code **(k)** onto the Web terminal or portable device. Moving code (applets) to the client allows a high level of interaction with the user despite a high-latency, low-bandwidth, or disconnected network.

Ultimately, Web sites and other Internet services will not be able to efficiently provide the full range of customization desired by their clients, and clients will want to use the same information-filtering and -organizing tools across many sites. Moreover, fixed-location, application-specific proxies will become bottlenecks, and as user needs change, may no longer be at the best network location for accessing the proxied services. As a result, customization tools will be specified as software, in the form of mobile code that runs either on the server, or on a dynamically selected proxy site **(m)** near the server (or client, as appropriate). Mobile code is necessary, rather than client-side code, since many customization features (such as information monitoring) do not work if the client is disconnected, has a low-bandwidth connection, or requires frequent communication with the server. Mobile code is beneficial, since servers and proxy sites need provide only a generic execution environment (along with an API that provides programmatic access to their service); the actual customization tools can be written by the services themselves, by third-party middleware developers, and even by the end users.

Finally, many clients will wish to send mobile code to multiple information sites as part of a single task. Although there will be applications for which the mobile code can be sent in parallel, many tasks require a sequence of subtasks, each at a different site. To avoid latency **(n)**, the application programmer will often want to avoid a "star-shaped graph" **(o)** where mobile code goes out to the first site and sends its results back to the client or proxy, the same or different piece of mobile code goes out to the second site, and so on, and the programmer will always want to be able to select the best migration strategy for the task and current network conditions. In other words, the mobile code must be able to hop sequentially through multiple sites; such multi-hop mobile code is a mobile agent.

4 Technical hurdles

There are several technical hurdles that must be cleared before mobile agents can be widely used.

Performance and scalability. Current mobile-agent systems save network latency and bandwidth at the expense of higher loads on the service machines, since agents are often written in a (relatively) slow interpreted language for portability and security reasons, and since the agents must be injected into an appropriate execution environment upon arrival. Thus, in the absence of network disconnections, mobile agents (especially those that need to perform only a few operations against each resource) often take longer to accomplish a task than more traditional implementations, since the time savings from avoiding intermediate network traffic is currently less than the time penalties from slower execution and the migration overhead. Fortunately, significant progress

[2] e.g., `http://www.metacrawler.com/`.

[3] e.g., Apple's "Sherlock" meta-search tool, http://www.apple.com/sherlock/.

has been made on just-in-time compilation (most notably for Java), software fault isolation, and other techniques [MMBC97], which allow mobile code to execute nearly as fast as natively compiled code. In addition, research groups are now actively exploring ways to reduce migration overhead. Together, these efforts should lead to a system in which accepting and executing a mobile agent involves only slightly more load than if the service machine had provided the agent's functionality as a built-in, natively compiled procedure.

Portability and standardization. Nearly all mobile-agent systems allow a program to move freely among heterogeneous machines, e.g., the code is compiled into some platform-independent representation such as Java bytecodes, and then either compiled into native code upon its arrival at the target machine or executed inside an interpreter. For mobile agents to be widely used, however, the code must be portable *across mobile-code systems*, since it is unreasonable to expect that the computing community will settle on a single mobile-code system. Making code portable across systems will require a significant standardization effort. The OMG MASIF standard is an initial step, but addresses only cross-system communication and administration [MBB+98], leading to a situation in which an agent can not migrate to the desired machine, but instead only to a nearby machine that is running the "right" agent system. The mobile-agent community must take the next step of standardizing on some specific execution environment(s) (such as a particular virtual machine), as well as on the format in which the code and state of a migrating agent are encoded.

Security. It is possible now to deploy a mobile-agent system that adequately protects a machine against malicious agents [Vig98]. Numerous challenges remain, however: (1) protecting the machines without artificially limiting agent access rights;[4] (2) protecting an agent from malicious machines; and (3) protecting groups of machines that are not under single administrative control. An inadequate solution to any of these three problems will severely limit the use of mobile agents in a truly open environment such as the Internet. Fortunately, groups are now exploring many new techniques, each of which addresses (or partially addresses) one of the three problems (e.g., agents paying for resource usage with electronic cash [BKR99], which allows them to live and propagate only as long as their cash supply holds out). Although many technical advances (and user-education efforts) must be made before these three problems are solved adequately for *all* Internet applications, current work is promising enough that, within a few years, mobile-agent systems will be secure enough for *many* applications.

[4]Many mobile-agent systems reduce an agent's access rights when it arrives from a machine that is not trusted, even if it was launched from a trusted user at a trusted site. The concern is that the agent may have been maliciously modified at the untrusted site.

5 Non-technical hurdles

Once the technical challenges have been met, there remain several non-technical issues that may deter the widespread adoption of mobile-agent technology. Internet sites must have a strong motivation to overcome inertia, justify the cost of upgrading their systems, and adopt the technology. While the technological arguments above are convincing, they are not sufficient for most site administrators. In the end, the technology will be installed only if it provides substantial improvements to the end-user's experience: more useful applications, each with fast access to information, support for disconnected operation, and other important features.

Lack of a killer application. The most important hurdle is that there is no "killer" application for mobile agents. The "mobile agent" paradigm is in many respects a new and powerful programming paradigm, and its use leads to faster performance in many cases. Nonetheless, most particular applications can be implemented just as cleanly and efficiently with a traditional technique, although different techniques would be used for different applications. Thus, the advantages of mobile agents are modest when any particular application is considered in isolation. Instead, researchers must present a set of applications and argue that the *entire set* can be implemented with much less effort (and with that effort spread across many different programming groups). At a minimum, making such an argument demands that the mobile-agent community actively support anyone who is writing a high-quality survey of mobile-agent applications, since no one group will be able to implement a sufficient number of applications. Once a clear quantitative argument is made, a few major Internet services can be convinced to open their sites to mobile agents, since they will recognize that agents will lead to more applications based around their services and hence more users. From there, more Internet services will follow.

Getting ahead of the evolutionary path. It is unlikely that any Internet service will be willing to jump directly from existing client-server systems to full mobile-agent systems. Researchers must provide a clear evolutionary path from current systems to mobile-agent systems. In particular, although full mobile-agent systems involve all the same research issues (and more) as more restricted mobile-code systems, researchers must be careful to demonstrate that the switch to mobile agents can be made incrementally.

For example, "applets", mobile code that migrates from server to client for better interaction with the user, are in common use, and the associated commercial technology is improving rapidly (e.g., faster Java virtual machines with just-in-time compilation). From applets, the next step is proxy sites that accept mobile code sent from a mobile client. In all likelihood, such proxies will be first provided by existing Internet service providers (ISPs). Since the sole function of the proxy sites will be to host mobile code, and since the ISPs will receive direct payment for the proxy service (in the form of user subscriptions, although not likely at a fixed rate), the ISPs will be willing to accept the perceived security risks of mobile code. Once mobile-code security is further tested on proxy sites, the services themselves will start to accept "servlets", mobile code sent from the client directly to the server (or from the proxy to the server).[5] Once servlets become widely used, and as researchers address the issue of protecting mobile code from malicious servers, services will start to accept mobile agents.

Another critical evolutionary path is the migration of agent technology from intranets to the Internet. Mobile-code technologies will appear first in the relatively safe intranet environment, particularly intranets that are built on high-latency networks such as a WAN or a wireless network for mobile computers. For example, a large company, particularly one with a mobile workforce, might find mobile agents the most convenient way to provide its employees with a wide range of access to its internal databases. Intranets tend to be early adopters of new (useful) technology, because their administrators have more control over the intranet than over the Internet; that control means that security is less of a concern, and wide deployment of agent support services can be encouraged. As the technologies mature in intranets, site administrators will become comfortable with them, and their practicality, safety and potential uses will become clear. Then they will find their way into the Internet.

Revenue and image. A final important hurdle is the problem of revenue flow and commercial image. For example, although it is not yet clear whether advertising is a viable economic foundation for Web sites, many Web sites earn money solely from advertisements. If these sites allow mobile agents to easily access the content of the site, the number of human visits to the Web pages will presumably decrease, and the advertisements will not be seen. How, then, will the site earn revenue? Similarly, when users are accessing a service with a front-end backed by mobile agents, the distinction between the service and the front-end agents starts to blur. Since the agents will likely be provided by middleware developers, the Internet service will no longer have complete control over its image. A poorly implemented agent may lead to a negative view of the *service*, even though the service is blameless. We believe, however, that mobile agents can be deployed in the near-term in many applications where the existing services do not rely on advertising; in the long-term, both the Internet and mobile-agent communities will need to explore different revenue models.

6 Conclusion

There is a strong case for the use of mobile agents in many Internet applications. Moreover, there is a clear evolutionary path that will take us from current technology to widespread use of mobile code and agents within the next few years. Once several technical challenges have been met, and a few pioneering sites install mobile-agent technology, use of mobile agents will expand rapidly.

[5] Like applets, and unlike agents, servlets are mobile code but not mobile processes. The code is moved from client to server, starts execution, and later ends execution on the same machine. It cannot migrate further once it starts executing.

7 Acknowledgments

Many thanks to the Office of Naval Research (ONR), the Air Force Office of Scientific Research (AFOSR), the Department of Defense (DoD), and the Defense Advanced Research Projects Agency (DARPA) for their financial support: ONR contract N00014-95-1-1204, AFOSR/DoD contract F49620-97-1-03821, and DARPA contract F30602-98-2-0107; to Jon Bredin, Brian Brewington, and Arne Grimstrup for invaluable feedback on early drafts of this paper; and to the anonymous reviewers (of the mAC3 workshop) for their useful and thought-provoking comments.

References

[BKR99] Jonathan Bredin, David Kotz, and Daniela Rus. Economic markets as a means of open mobile-agent systems. In *Proceedings of the Workshop "Mobile Agents in the Context of Competition and Cooperation (MAC3)" at Autonomous Agents '99*, pages 43–49, May 1999.

[DR99] Amitava Dutta-Roy. Bringing home the Internet. *IEEE Spectrum*, 36(3):32–38, March 1999.

[Gri99] Corey Grice. When will data change the wireless world? *CNET NEWS.COM*, February 10, 1999.

[LO99] Danny B. Lange and Mitsuru Oshima. Seven good reasons for mobile agents. *Communications of the ACM*, 42(3):88–89, March 1999.

[MBB+98] D. Milojicic, M. Breugst, I. Busse, J. Campbell, S. Covaci, B. Friedman, K. Kosaka, D. Lange, K. Ono, M. Oshima, C. Tham, S. Virdhagriswaran, and J. White. MASIF: The OMG Mobile Agent System Interoperability Facility. In *Proceedings of the Second International Workshop on Mobile Agents*, volume 1477 of *Lecture Notes in Computer Science*, pages 50–67, Stuttgart, Germany, September 1998. Springer-Verlag.

[MMBC97] G. Muller, B. Moura, F. Bellard, and C. Consel. Harissa: A flexible and efficient Java environment mixing bytecode and compiled code. In *Proceedings of the Third Conference on Object-Oriented Technologies and Systems*, pages 1–20, 1997.

[Sch99] Seth Schiesel. Nortel plans new product to bolster optical networks. *The New York Times*, May 9 1999.

[Vig98] Giovanni Vigna, editor. *Mobile Agents and Security*, volume 1419 of *Lecture Notes in Computer Science*. Springer-Verlag, 1998.