Applying Mobile Agents to Enable Dynamic, Context-Aware Interactions for Mobile Phone Users

Mark Adler, Jeffrey M. Bradshaw, Michael Mahan, Niranjan Suri

Nokia Research Center
5 Wayside Road, Burlinton, MA USA
{mark.adler, michael.mahan}@nokia.com
Institute for Human & Machine Cognition, University of West Florida
40 S. Alcaniz St., Pensacola, FL USA
{jbradshaw, nsuri}@ai.uwf.edu

Abstract. Mobile agents can play a critical role in enabling dynamic applications on mobile phones. They can carry executable code, making possible effortless downloading of new capabilities and services to mobile phones. When combined with services that support context awareness, user customization, and sensitivity to the mobile phone environment, mobile agents can be used to provide the basis for a rich set of applications. This paper provides an overview of the problems faced in this application domain and outlines the approach we are following in our research.

1 Introduction

Mobile agents are software entities that are capable of moving themselves from one platform or host to another platform or host over the network. Unlike applets, which are pulled in a single hop from server to client, mobile agents determine their own itinerary, which may include a whole series of moves and stops in the performance of its tasks. Since they can carry code as well as data when they move, they can provide their host with new capabilities and behavior in addition to information. By optimizing the location of computing resources, mobile agents support bandwidth-efficient communication, which is particularly relevant given the widening gap between wired and wireless bandwidth. Mobile agents also support disconnected operation—the ability to continue computing on a server while the phone may be unavailable—which is extremely useful in situations with intermittent network connectivity. Finally, mobile agents can be used to move computations to backend servers, thereby reducing the processing requirements and the load on mobile phones (and consequently on batteries).

Mobile agents can be deployed to mobile phones in order to add new executable code to the phone. Once a mobile agent is deployed, the agent can remain on the phone for short or extended periods of time. Therefore, mobile agents can be used to push new services and functionality to phones for either short-term or long-term purposes. Moreover, by temporarily moving agents out of the phone to a backend

server, services can be swapped in and out of the phone on an as-needed basis– particularly important given the memory constraints typical of such devices.

In this paper, we describe our ongoing research to use mobile agents to enable such capabilities for mobile phone users. We begin by establishing the requirements through a scenario – a researcher attending a conference (such as this one). While this is only one of many possible scenarios, it does serve to illustrate the kinds of capabilities enabled by our mobile agent services. Then, we describe the technical requirements necessary to achieve the scenario. Finally, we describe the current state of our implementation.

2 Motivating Scenario

In the not so distant future, agent technology will transform the way people deal with the logistics of conference attendance. To develop a context for the following discussion of technical requirements and implementation, here is a glimpse of how mobile agents operating with mobile phone devices will enhance the experience of attending conferences of the future.

Registering for a conference, handling travel arrangements, accommodations, and rental cars are tasks that one would like to delegate to a secretary or a personal agent who knows one's travel-related preferences, calendar constraints, and can handle finding the best air fares, etc. This kind of agent scenario is not novel and does not require mobile agent technology. But mobile agents can be applied to enable dynamic, context-aware interactions that simplify and/or enhance the user experience.

A conference attendee first registers for the conference via the Internet using either conventional means (e.g., using the Web or by voice over the phone) or a hybrid mobile phone–PDA device. After the registration fee is charged to the user's credit card or authorized by the mobile phone, a mobile agent from the conference is loaded onto the user's mobile device. This agent enables access to conference hotel information, conference schedules, contact information for conference attendees, conference proceedings, and so forth. The user's own agent can interact with the conference agent to make hotel reservations, for example, since the user's preferences are situated with the user's local agent.

The Mobile Conference Agent (MCA) enables the user's mobile device in three ways:

- It encapsulates conference and local accommodation information to simplify registration and deal with accommodation details.
- It transforms the mobile phone into an enhanced conference badge, providing a security key for access to conference functions, automating identification for vendors, enabling access to dynamic conference materials on the Web, etc.
- It serves as a gateway to other registered conference attendees to facilitate informal meetings and establish birds-of-a-feather gatherings, find acquaintances among the attendees, and so forth.

This is how mobile agents can simplify daily tasks for a prototypical conference attendee, Pat. After Pat registers for the conference, the conference injects a conference agent into Pat's mobile phone. The mobile conference agent is able to

communicate with Pat's personal agent to simplify travel, hotel, and transportation selection. With both agents co-resident on the phone, communication is reduced, and battery drain is reduced. The MCA contains information about conference hotels, flight schedules, the conference timetable, and conference-related discounts. The MCA can relay the travel information back to the conference organizers to help schedule shuttle buses to meet incoming flights. Through the mobile phone, now enhanced by the MCA presence, Pat can browse through the electronic abstracts rather than being burdened by heavy printed volumes only available at the conference. If Pat wishes to print a hardcopy of a particular paper, the MCA will fetch a printable version of the paper and send it to a printer of Pat's choice.

On arrival at the conference, there is no need to register again and no additional information to obtain; all this has already been handled by the MCA. At the hotel desk, the mobile phone communicates with the hotel agent, verifies Pat's identity, coordinates Pat's preferences to find the most suitable room available, and confirms that Pat gets the conference discount rate. Upon checking in, a customized hotel agent is downloaded to Pat's phone. This hotel agent acts as the room key, allowing Pat to access her room as well as other facilities in the hotel. The MCA confirms that Pat gets the conference discount rate. After Pat is in her room, her personal agent interacts with the hotel agent to reconfigure the room to Pat's preferences for lighting, TV networks, and radio station for the radio alarm clock. Based on the conference schedule, and Pat's usual 90-minute morning routine, the alarm can be set to ensure that Pat will make it to the conference on time.

That evening, Pat's mobile phone filters information from the MCA to find some old friends attending the conference. Using the mobile phone, Pat arranges an informal meeting in the lounge. Agent technology using onboard calendar information and contact information from the MCA is used to get all the friends together. Once together, they decide to continue their discussions over dinner. The MCA provides local restaurant information, each participant's agent represents their dinner preferences, and the multi-agent planning system finds a suitable restaurant and reserves a table. After dinner, the MCA helps the dinner party discover nearby movie theaters, including what films are showing at what times. A multi-agent planning application can again take the users' preferences, schedules, and transportation requirements to plan for the after-dinner movie.

At the conference the next day, the MCA enhanced mobile phone serves as an ID badge, and only registered conference attendees are allowed to enter the conference exhibits and lectures. Having the schedule and all the abstracts at hand helps conference attendees choose between parallel tracks. By tracking (in real time) which talks are attended by attendees with common interests, the MCA can influence attendance at talks. Using agent technology, any late registrants can instantly arrange to register electronically, and immediately gain access.

We have outlined just one of many possible scenarios that illustrate how mobile agents could be used to significantly enhance the capabilities of a mobile phone. The capabilities described in this paper can enable applications such as the MCA and several others.

3 Technical Requirements and Implementation

In this section, we outline the envisioned technical requirements and implementation that are necessary for applications such as the MCA.

3.1 Discovery of services and other "relevant" users

In order for agents to discover available services, or for that matter, other agents in the vicinity, there must be a system in place to discover what is available in the vicinity. There are several systems, such as Jini, that provide this facility. Essentially, service providers must advertise their services on the network so that consumers can find them. To avoid being overwhelmed by such advertisements, physical proximity to the services is a good filter.

Service discovery either relies on a strict (and limited) interface, or an open-ended system that relies on a common, but expandable ontology to describe the goods and services. An ontology provides a basis for describing and understanding the services being offered.

Various AI technologies, particularly from the area of Knowledge Representation (KR), will provide a basis for building more powerful discovery mechanisms. In the context of distributed agent systems, the notion of the *semantic web* [1] is very appropriate. To achieve such a system, the still-emerging knowledge representation formalism DAML (DARPA Agent Markup Language [2]) and its foundation, W3C's RDF (Resource Description Framework [3][4][5][6], will provide the basis for efforts to populate the Web with content that has formal semantics; thus the semantic web will enable automated agents to reason about Web content and produce an intelligent response to unforeseen situations through matchmaking, management, and control mechanisms based on DAML representations of service descriptions and policies [7].

Sharing vocabularies and models allows automated interoperability; given a base ontology shared by two agents, each agent can extend the base ontology while achieving *partial understanding*. A base ontology is analogous to OOP systems, where a base class defines "common" functionality.

3.2 Context-awareness

Context-awareness plays an important role in the mobile computing environment [8]. Context acquisition in a mobile computing is provided explicitly by the user or implicitly by monitors [9]. We address two components of context information relevant to our described use case. The first is physical or geographical location. Such data can be provided by GPS when one is outside, or by triangulation based on signal strength in a cellular environment. Indoors, one needs to construct an analogous means of determining location by using beacons and receivers of some variety, for example, the Cricket system under development at MIT [10]. This data can be used to recognize one's location and orientation, determine the distance between one's

current location and an advertised service, and even generate a map with directions of how to get from here to there.

The second component, though equally relevant to agent behavior, is the notion of personal context. By personal context, we mean an understanding of the role of the user at any given time. This contextual information should be used to alter the behavior of an agent with respect to the user [11]. For example, during a lecture or a meeting, a phone should switch to silent ringing mode; at a coffee break between sessions, it should switch its volume to maximum to be heard above the commotion. At the agent level, this context is also relevant. Knowing whether the user can be interrupted or not may cause the agent to defer confirmation, and take on a more autonomous role. The agent also needs to be able to coordinate its action with the actions of other people and devices working with the user [12].

3.3 Customization through user preferences

The primary location of the user's personal assistant is the mobile phone. The personal assistant encapsulates two types of personal data for a user: raw Personal Information Management (PIM) data and preferential information. PIM data is stored and accessed from local and remote PIM servers. Preferential information is directly managed and maintained locally by the personal assistant. The user's messaging and alert preferences, contact preferences, scheduling preferences, and environmental preferences influence conferencing applications presented by the MCA. Using PIM data, preferences, and the presence of the MCA; the personal assistant will both customize conference services and adapt the physical environment to the user's liking.

Preferences can be applied bidirectionally. Either the personal assistant or the MCA can initiate the transaction where user preferences are applied. Relative to the MCA, the personal assistant acts as a user interface proxy (conference messages and alerts, etc.) and an interface for service interactions. Relative to the personal assistant, the MCA acts as the interface for all conference services. Three basic patterns for applying user preferences are as follows:

- **Interruption receptivity**. The personal assistant has the means and capacity to evaluate user receptivity to interruption. Knowing the context of the interruption and the user's current situation, the personal assistant will rate relevance and choose to handle the MCA's UI-related request in a manner consistent with explicit preferences [11].
- **Service customization**. The personal assistant will customize the services offered by the MCA to conform to PIM data and user preferences. One illustrating scenario details how a user receives conference materials. The user can prefer the receipt of only a subset of the conference papers that pass a keyword filter. Furthermore, the user prefers that these papers are transferred electronically to a user accessible document repository and to then insert "read this paper" tasks for especially relevant papers identified by a context filtering application. Some other scenarios for the personal assistant to customize the

services supplied by the MCA are to tailor lecture and workshop registrations, to choose conference meals, and to schedule meetings with peer researchers.

- **Environmental adaptation.** The personal assistant will also use the MCA agent to adapt to her surroundings. The conference hotel room can be selected upon preferential data: no smoking, near ground level, and outfitted with specific appliances. Dynamic environmental adaptation is also desirable, for example, setting the hotel room morning alarm system (clock radio or wake-up call) based on workout and conference schedule, setting the music, etc [9].

We intend to explore the representation of preferences in DAML, based on extensions to our DAML-based policy representations and mechanisms [7].

3.4 Sensitivity to mobile phone environment

As mobile phones and PDAs converge into a single device with greater communication bandwidth, there are still attributes that separate a mobile phone from a desktop or laptop with a broadband connection. Beyond the physical limitation of the small screen size, mobile handsets have three distinguishing characteristics:

- Limited battery life and, therefore, extreme sensitivity to functions that are power hungry.
- Extremely variable bandwidth depending on location, proximity to cell towers, and type of carrier. Signals may be lost entirely for periods of time, and the greater the distance from the tower, the greater the power requirements.
- Limited on-board computation. These limits include reduced memory sizes, no disk storage, and relatively slow CPU speeds.

So, in a mobile agent world, one can see the utility of off-loading a mobile agent to some external host on the Internet to perform some task in the relative luxury of a richer computational space, and returning later having achieved some computational goal. However, there are reasons why it may also make sense for an external mobile agent to inhabit the mobile phone:

- Privacy and security concerns may more easily be met by performing the computation on the phone to ensure that sensitive personal data is not compromised.
- The mobile agent may transform the behavior of the phone by providing additional functionality.
- The data needed for some computation is local to the phone, and the resulting cost is reduced by running the agent on the phone, rather than transferring the data to and from the network.

This last point suggests that there is some evaluation function that could be computed to determine whether the agent (and data) should be based on the phone or on some network host. We are beginning to define such an equation, and have identified key components. The equation is based on the following parameters:

- **Power consumption.** The amount of power to compute the result. This includes the power to perform the computation locally including the cost of obtaining the data, (the power required to transfer the data to the mobile phone), and the cost of

running the computation locally. Unfortunately, transmitting and receiving data are the most severe consumers of battery power.

- **Capacity.** The ability for the mobile phone to host the application, including sufficient memory and computational cycles. Currently, we are assuming that there is no monetary fee for hosting an agent either on the mobile phone or on some host computer on the Internet.
- **Time.** The estimated elapsed time to reach a result.
- **Risk.** Some measurement of risk of transferring sensitive data into the network.

3.5 Security and trust with respect to access to information

The personal assistant is the only exposed application interface to the MCA on the device. The mobile phone is a highly personal, secured device, so the MCA is disallowed direct access to resident applications. The hosting environment on the mobile phone must guarantee this security.

The user has established a trust perimeter about her personal assistant by describing what tasks it undertakes and how it will consider preferential and personal data while executing those tasks. Hence the personal assistant is semi-autonomous — it will need to interact with its sponsor, from simple informative messages to complex queries. For some tasks, the assistant maintains complete autonomy, since the action is within the trust perimeter the user grants the assistant. However, other tasks, say monetary or privacy-related, require user intervention. Granted privileges may be dynamic — the trust a user has in its agent shadows an assistant's satisfactory or poor performance.

Security and trust are significant issues for mobile agents [13], [14]. With respect to information security and trust, two separate kinds of policies are considered for hosting and interacting with a MCA on the mobile device. The first is to assume that all information the personal assistant gives to the MCA will be public so it is the personal assistant's direct responsibility to limit the exposure of sensitive personal data or deducible preferences. The second policy is put into force with the MCA upon MCA migration to the mobile device or prior to migration. This policy will bind the MCA or the conference agent system to not expose personal information or preferential constraints. With an enforceable policy in place, the MCA can become more than a proxy for the conference system. It can become a smart delegate for the personal assistant to use and trust. The trust perimeter can be extended to the MCA.

3.6 Code mobility

Code mobility is a core requirement for scenarios such as the one outlined earlier. Code mobility allows new capabilities to be downloaded dynamically to mobile phones. In the conference setting, the MCA carries with it new code in order to provide the functionality specific to the conference that the user is attending. We expect that a mobile phone user will experience a variety of situations that will benefit from specialized code being dynamically downloaded to the mobile phone. For

example, each airline might provide a customized flight agent that is sent to a mobile phone when a user makes a reservation. This flight agent could help users make seat selections, meal selections, get information about flights and gates, show maps of airline terminals, and provide access to airline clubs. Similarly, hotels could have customized agents that are downloaded to a user's mobile phone upon checking in.

Another significant advantage of code mobility is support for small memory capacities. Since code mobility allows code to be downloaded to a mobile phone on demand, code mobility also allows the luxury of removing code that is not currently required. For example, in the previous scenario, before the MCA is downloaded to the mobile phone, other agents left over from previous situations (such as a flight agent from the last flight taken by the user) can be removed. Similarly, once the conference is completed, the MCA could be removed to make room for another agent (such as the flight agent for the user's return flight back home).

3.7 Safe and controlled execution

The mobile phone may become the most personal of devices. It will host or access our private messages, contacts, access keys, and credit and debit electronic payment systems. For this device to host a mobile agent, it must be secured from malicious attack or inept agent behavior. At an agent application level, policy-based mechanisms are a partial solution. At lower abstraction levels, enforcement of these policies through the mobile phone's base applications, operating system, and drivers must be assured.

Safe execution is particularly important with mobile code. Currently, most mobile code systems rely on code signing to protect an execution environment. However, while code signing provides a means of determining the originator of the code, it is by no means a guarantee regarding the performance of the code. Therefore, even code that has been signed could still be malicious or buggy.

We also feel that as situations become more dynamic and the capabilities of mobile phones improve, mobile phones will see a significant increase in mobile code usage. For example, end users could use mobile agents as active mail by sending executable content to other users. In addition, if multiple hop scenarios arise, agents could also be tampered by malicious execution environments. An example of a multiple hop scenario is a meeting scheduler agent that visits a number of mobile phones to consult user calendars in order to schedule an appointment.

Another requirement is being able to control the resource usage of agents executing on a mobile phone. If phones are to host more than one mobile agent simultaneously, the execution environment must be able to distribute the resources appropriately to the agents. Also, user operations and the environment of the mobile phone must be taken into account. For example, future mobile phone systems are likely to be packet-based, allowing more than one communication channel to be active simultaneously. While a user is communicating over a voice channel, agents might still be allowed to communicate with other agents or services. Moreover, the total bandwidth available to the mobile phone may vary over time, as the number of customers in a cell change or as the user moves across communication cells. Under such circumstances, the

execution environment must be able to limit and distribute the bandwidth used by agents so as to not interfere with the user's voice communication.

3.8 Implementation architecture

The following mobile reference architecture can be used to realize the conference attendance scenario or another similar application that requires mobile agents and mobile devices [15]. The reference architecture identifies the required, high-level components for agents to be discovered, moved onto the mobile device, interact with a user's agent on this device, and interact with remote services. These mobile agents can be self-contained, or can act as a networked federation to provide a more dynamic service to the mobile phone user vis-à-vis the user's personal assistant agent.

The personal assistant (PA) lives and runs on the mobile device. The 2.5/3G network and the low-power wireless network will provide coarse and fine-grained information to ascertain device location. In addition to interacting with mobile application services, the personal assistant will communicate with PIM services, location independent services, and location dependent services. One special type of location dependent service, a local ad hoc service, can be made available across the low-power wireless network. Mobile application services are unique in that they can dispatch a mobile application agent (MAA) to a host phone. If a desired mobile application service is detected from across the low power network, a MAA can migrate to phone either across that network or the 2.5/3G network, dependent on agent size and bandwidth considerations.

Once on the mobile phone, the MAA will be hosted inside the safe execution environment. All user interactions and phone application services requested by the MAA are routed through the PA. All other resource requests are supplied or denied by the safe execution environment.

The safe execution environment will be based on the NOMADS mobile agent system [16] and the capabilities of the KAoS agent framework [7]. NOMADS provides unique capabilities for strong and forced mobility and safe execution of mobile agents. Strong mobility allows the execution state of an agent to be captured and moved with the agent from one host to another. In addition, NOMADS relies on its state-capture mechanism to support forced mobility, which allows the system to move agents from one host to another at the discretion of a user or agent management facility, (potentially in a completely transparent manner to the agent. Such forced mobility is essential for applications involving load balancing, process migration, devices shutting down, and so forth. Safe execution of agents is based on the ability of NOMADS to control the resources accessed and consumed by agents. The resource control mechanism allows control over the rate and quantity of resources used by agents. Dynamically adjustable limits can be placed on several parameters including the disk, network, and CPU. These resource control mechanisms complement Java's access control mechanisms and help in making the NOMADS system secure against malicious and buggy agents. NOMADS derives its unique capabilities from a custom Java Virtual Machine called Aroma.

Complementing the NOMADS features, the KAoS agent framework provides mechanisms for overall management of agents grouped into domains. The KAoS domain manager serves as a policy decision point to determine whether agents can join a domain and for policy conflict resolution. Guards interpret policies that the domain manager has approved and enforce them with appropriate native mechanisms. The domain manager ensures policy consistency at all levels of a domain hierarchy, notifies guards in the event of a policy change, and stores policies in a secure repository. These policies are stored in an implementation-neutral format, currently very simple but soon to be based on our DAML policy representation. Because the library expresses the policies declaratively, authorized entities can analyze and verify them in advance and offline, maximizing the efficiency of execution mechanisms.

KAoS policy-based agent management includes features supporting authorization, encryption, and access control while adding the ability to represent policy for NOMADS resource control mechanisms. But because of our focus on agent systems, KAoS goes beyond these typical security concerns in significant ways. For example,

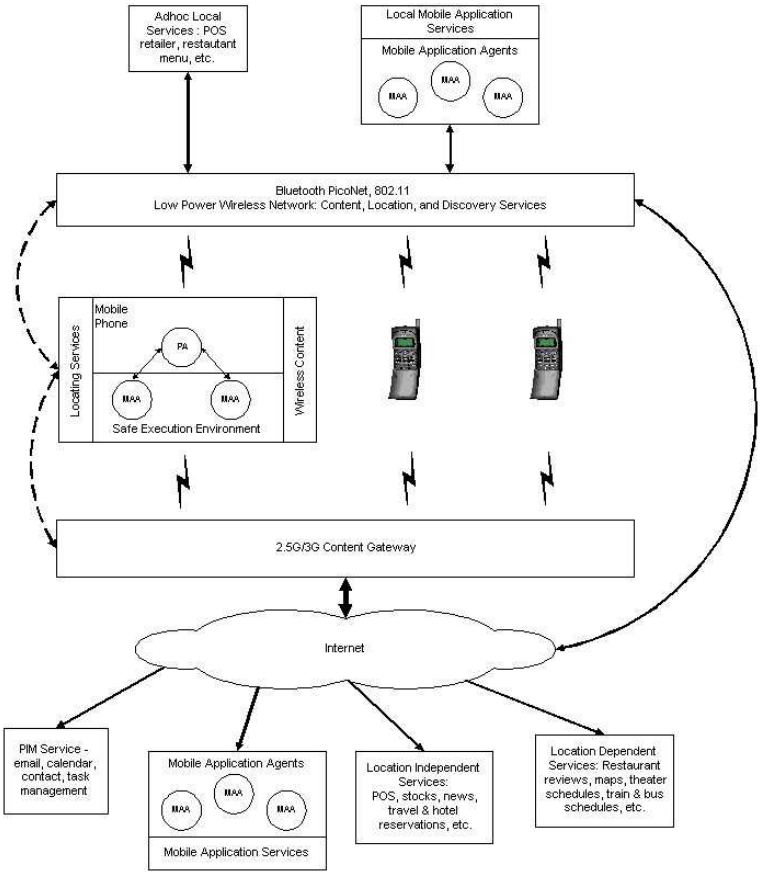

Fig. 1. Implementation Architecture

the KAoS architecture introduced the concept of agent conversation policies. The agent-to-agent communication process uses appropriate semantics to form, maintain, and disband teams of human and software agents assisting the user with a given task [12]. In addition to conversation policies, we are developing representations and enforcement mechanisms for mobility policies, privacy policies, domain registration policies, and various forms of obligation policies.

References

1. Dieter Fensel (ed.): "The Semantic Web and its Languages", IEEE Intelligent Systems 15(6): 67-73 (November/December 2000).
2. James Hendler & Deborah L. McGuinness: "The DARPA Agent Markup Language", in [1].
3. Ora Lassila: "Web Metadata: A Matter of Semantics", IEEE Internet Computing 2(4): 30-37 (1998).
4. Ora Lassila & Ralph Swick: "Resource Description Framework (RDF) Model and Syntax Specification", W3C Recommendation 1999-02-22.
5. Dan Brickley and R.V.Guha: "Resource Description Framework (RDF) Schema Specification 1.0", W3C Candidate Recommendation 2000-03-27.
6. Ora Lassila & Deborah L. McGuinness: "The Role of Frame-Based Representation on the Semantic Web", Electronic Transactions on AI (to appear); available as a Stanford KSL technical report KSL-01-02.
7. Jeffrey M. Bradshaw, Niranjan Suri, Alberto Cañas, Robert Davis, Kenneth Ford, Robert Hoffman, Renia Jeffers, and Thomas Reichherzer: "Terraforming Cyberspace," IEEE Computer, 2-10 (July 2001).
8. Bill Schilit, Norman Adams, & Roy Want: "Context-Aware Computing Applications", IEEE Workshop on Mobile Computing Systems and Applications, December 8-9, 1994.
9. Albrecht Schmidt, Michael Beigl, & Hans-W. Gellersen: "There is more to Context than Location", *Computers & Graphics Journal* 23(6): 893-902.
10. The Cricket Location-Support System. *In Proc. 6th ACM MOBICOM Conf.* (Boston, MA, Aug. 2000).
11. Anthony Jameson: "Modeling both the Context and the User", Personal and Ubiquitous Computing 5(1): 29-33.
12. Jeffrey M. Bradshaw, Maarten Sierhuis, Yuri Gawdiak, Renia Jeffers, Niranjan Suri, and Mark Greaves: "Adjustable Autonomy and Teamwork for the Personal Satellite Assistant." In Proceedings of the IJCAI-01 Workshop on Autonomy, Delegation, and Control: Interacting with Autonomous Agents, Seattle Washington, in press (6 August 2001).
13. W. M. Farmer, J. D. Guttman, and V. Swarup. Security for mobile agents: Authentication and state appraisal. In Proceedings of the European Symposium on Research in Computer Security (ESORICS), pages 118--130, September 1996.
14. U. G. Wilhelm, L. Buttyàn, and S. Staamann. On the problem of trust in mobile agent systems. In Symposium on Network and Distributed System Security. Internet Society, Mar. 1998.
15. Michael Mahan: "Agents and Mobile Handsets", IEEE Workshop on Wireless Networks and Mobile Computing, April 16-19, 2001: 448-451.
16. Suri, N., Bradshaw, J.M., Breedy, M.R., Groth, P.T., Hill, G.A., and Jeffers, R. Strong Mobility and Fine-Grained Resource Control in NOMADS. Proceedings of the 2nd International Symposium on Agents Systems and Applications and the 4th International Symposium on Mobile Agents (ASA/MA 2000). Springer-Verlag.