Software Demo: Delegating Offline Shopping Tasks to Mobile Agents

In the course of the European ESPRIT Project AIMedia (Targeted Advertising on Interactive Media) we developed a proof of concept application based on mobile agent technology by which we intend to demonstrate some of the benefits that can be expected from using mobile agent technology. The basic idea is to delegate a high-level shopping task to a mobile agent (the personal commerce assistant, PCA). In our case, the agent’s goal is to organise the catering for a birthday party. The graphical front-end to the application is managed by a dedicated agent that may be accessed through the World Wide Web using commonplace browsers. The demonstrator is built on top of a security-centric mobile agent system that we developed. The system is under active development and is available as open source from our Web site. Among the security features of the system are services for transparent authentication of agents, and selective encryption of agent contents for configurable access groups.

The PCA scenario consists of the user interface agent, the search agent, a recipes server, a Yellow Pages server, and the servers of retailers. The user either runs a lightweight agent server himself or attaches his browser to the PCA run by some service provider (the browser is used in both cases as the user interface).

The user interface agent allows the user to configure the PCA with information on the person giving the party and the expected number of guests. So far, this information includes the sex, age group, category of hobbies, preferred beer brand, and “theme” of the food (either Chinese, American, European, or none in particular). On submitting the information the user interface agent generates and returns an URL that can be used to retrieve the results once they become available. At the same time it selects two meals (by name) and beverages based on the entered information. If the age group is above 18, five cocktails are also selected (by name). The user interface agent then creates and configures a search agent.

The search agent then migrates to a recipes server on which it resolves the names of the selected recipes against a small database of recipes. The recipes are normalised to one serving. The quantities are multiplied by the expected number of guests by the search agent. The search agent also looks for a picture database service and retrieves pictures of the selected meals and cocktails if available. The search agent then proceeds to the Yellow Pages server, locates the retailer information service and queries for the sites of retailers that may offer the required products. The search agent as-
sembles its subsequent itinerary and hops from one retailer to the next until all products are found or the itinerary is finished. The product databases were kindly provided for testing purposes by our project partners J. Sainsbury’s, United Kingdom, and OTTO Versand Hamburg.

The search agent then returns to the originating server, passes the results to the user interface agent, which formats them into a nicely looking HTML page with pictures of the food and beverages. This page is then made available through the URL initially passed as the response to the submission of the user information. It contains a complete shopping list with product details, numbers of required units and capacities per unit.

For demonstration, we usually use three agent servers. The first agent server runs the Web agent and user interface agent. The second server runs the recipes and yellow pages service. The third and final server runs the product information services (see Figure 1). The PCA application as well as the mobile agent system are developed purely in Java 2, leveraging of the improved security architecture of this Java version.
Hardware Requirements

The common demo setup consists of three computers: one front end and two backend servers. We bring the frontend (a Laptop running Windows NT). The backend machines should be either

- state-of-the-art PCs with Windows NT installed,
- or Sun Workstations (preferably UltraSPARC $\geq 200$Mhz) running SunOS 5.7.

All machines need to run the JDK version 1.2.2; on Suns the appropriate patches for this Java version need to be installed.

In case PCs are provided we use Microsoft Access as the database back-end; in the case of Sun workstations we will have to install PostgreSQL. The databases are accessed by means of JDBC. All machines need to be interconnected by an Intranet LAN.

Preferably, the machines should have an Internet connection though this is not strictly required for the demo. However, it might provide us a fallback to send agents to servers of our own for database access.