Université de Montréal

# MBAOS: A Mobile Bargain Agent for Online Shopping

Braz, Christina / christina.braz@hec.ca
Ncho, Ambroise / ambroise.ncho@umontreal.ca
Department of Computer Science and Operations Research
Université de Montréal, Montreal, Canada

Abstract

The Electronic commerce technology presents the opportunity to blend and optimize the consumers' needs in terms of availability, speedy response time, and efficiency. Consumers around the world are communicating via Internet to obtain the availability of goods and services, to place and confirm orders, and to negotiate delivery times. With the development of decentralized networks, it will be necessary a set of new forms of information retrieval, comparison-shopping, and a *simpler-no-complex* elaborations from agents to reach sufficient matchmaking and price setting. In this paper, we will describe the MBAOS, a Mobile Bargain Agent for Online Shopping that is triggered by a mobile device (cellular phone or PDA) to a remote server, that creates connectivity on an *anytime-anywhere-any-device-basis* to provide the specific goods required by the consumers based on transaction cost optimization and scalability which are a perfect match for a mobile agent technology, say, mobile agents. MBAOS is based on the Java mobile agent technology, called aglets. Aglets are mobile agents that are dispatched by the buyer to the several suppliers, where they negotiate orders and deliveries, returning to the buyer with their best deals for approval.

Index terms: Mobile Agents, Electronic Commerce, Wireless, Auctions, Information Retrieval.

## 1. INTRODUCTION

A software agent is a program that works on behalf of a human user. Mobile agent (MA) has the supplementary ability to travel autonomously (under its own control) from machine to machine on a network. MAs will be a fundamental part of the Internet in a short term, because mobile code makes new applications possible, it leads to considerably better performance than traditional techniques, and because it provides a general framework in which distributed, information-oriented applications can be applied efficiently and easily, with the programming concern scattered across information, middleware, and client providers. In other words, mobile code gives providers the time and flexibility to supply their users with more helpful applications, each with more worthy features, especially in the Ecommerce applications such as online comparison-shopping systems.

Many clients will wish to send mobile code to various information sites as part of a single task, for example, to ship a Mobile Bargain Agent from a cellular phone to a remote server which dispatches it to several bargaining Web sites (i.e., www.bargain.com) with the objective to get the best prices for a specific product. Although there will be programs for which the mobile code may be sent in parallel, many tasks require a battery of subtasks, each at a different site. Hence, the mobile code must be able to *jump* sequentially through multiple sites, and such *multi-jump* mobile code is a Mobile Agent (MA). MAs are programs that can migrate from host to host in a network, at

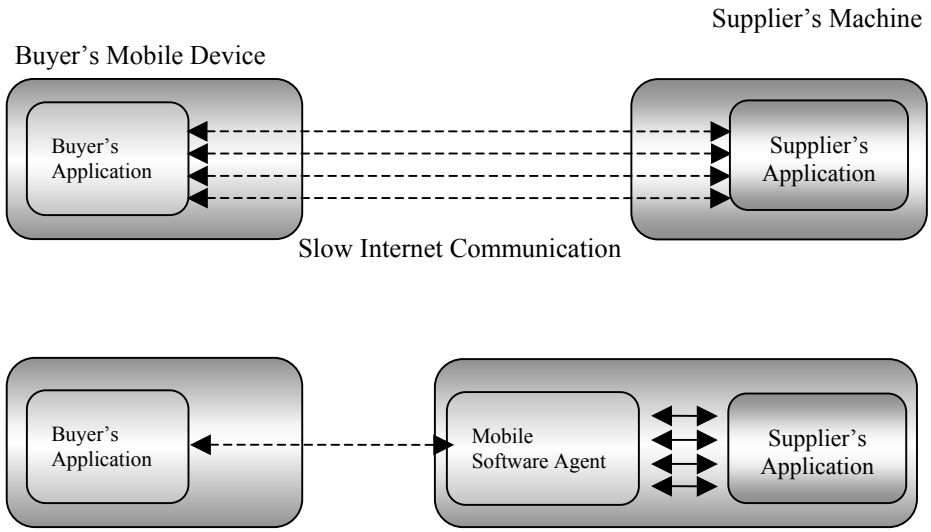
times and to places of their own choosing. The state of the running application is saved, delivered to the new host, and re-established, allowing the application to continue where it left off. MAs systems contrast from process migration systems in that the agents move when they choose, typically through a *jump & go* statement, whereas in a process-migration system, the system determines when and where to move the running process. Mobile agents contrast from applets, which are programs downloaded as the result of a user action, then executed from beginning to end on one host.

Our Mobile Bargain Agent (MBA) is a sort of "ShopBot", that is a comparison-shopping agent, but *mobile device-wise*. "Given the home pages of several online stores, ShopBot autonomously learns how to shop at those vendors. After learning, it is able to speedily visit over a dozen software and vendors, extract product information, and summarize the results for the user. ShopBot enables users to both find superior prices and substantially reduce Web shopping time. Remarkably, ShopBot achieves this performance with out sophisticated natural language processing, and requires only minimal knowledge about different product domains. Instead, ShopBot relies on a combination of heuristic search, pattern matching, and inductive learning techniques" (Doorenbos, Etzioni & Weld, 2003)[1].

Another important factor is the complex interactions required between the software systems of companies around the world that it will be extremely aided by the Internet and private Intranets. To shrink the delay that restricts closer interaction between software systems, it is suitable to use a mobile software agent technology.

As Figure 1 shows, the agent can conduct various interactions with application located on the remote computer to reach the negotiation or planning that is needed. The initial delay of sending the agent across a WAN (Wide Area Network) will be longer than the delay is run into only once. In this way, the savings in communication time may be quite significant.

Figure 1: Mobile Agents and Network Load Reduction




Source: Danny Lange & Mitsuru Oshima [3].

At present, we have Sun's Java language, which allows programs to be run on diverse platforms without modification or recompilation. IBM's Java Aglet Technology allows executing programs to move from one host to another within the network, as we have already mentioned at the beginning of our introduction.

This paper is structured as follows: We begin with a quick overview of Mobile Agents and the Mobile Bargain Agent systems. In Section 2 we rapidly describe the online bargaining. Section 3 reports the Methodology used for our system depicting the Java Aglet Technology and Jade Platform. In Section 4 we describe the implementation of the MBAOS's system with an overview of the architecture. Section 5 reports the empirical evaluation of our system, and Section 6 we report related work, and finally, Section 7 presents the conclusion and opportunities for future work.

## 2. THE ONLINE BARGAINING

Electronic exchanges are already a significant facilitator of negotiation systems and an important form of bargaining and matchmaking between potential trading partners. The majority of bargaining systems today are based on auction models which need a centralized authority, i.e. an auctioneer.

Bargaining may be defined as negotiating an agreement establishing what each party will give, receive, or perform in a transaction. For example, an e-auction. Agents may send each other messages in turn each asking, offering, threatening, or appealing to reach un agreement. However, for the purpose of this paper we are employing a single facet of bargaining, i.e. searching for the best price for a desired product. In fact, there is no simpler way to do comparison-shopping online.

The bargaining system is quite open in terms that any person with a computer and the ability to access the system is allowed to participate. There are no outlaws. However any participant is allowed to determine people he/she does not want to pick for a future deal. The bargaining system is automated, users have a minimal workload to initiate their participation and it is hoped that "in the future" they will play in the system.

In general, there are three levels of optimality in electronic bargaining systems such as:

- Price Optimality: associated to opportunistic behaviour and to the exploitation of the other party.
- Matchmaking optimality: find the best possible partner among a set of potential partners.
- Transaction cost Optimality: search costs, negotiation costs, implementation costs.

## 3. METHODOLOGY

The general framework for the MBAOS system was built with the JADE Platform and Sybase Database environment. We developed a Mobile Bargain Agent, which we think, is appropriate for an online shopping, or Shopping bot. In this model, we have the following entities: User (Home), sellerAgent, buyerAgent, Proxy site (Remote), and two bargaining Web sites (www.bargain.com and www.mysimon.com). In this way, given:

- Information about the product attributes such as year, make and model of a vehicle, or manufacturer, speed and model of a laptop.
- A set of URL's for the home pages of possible vendors.

Determine: The set of stores where the desired product is available, sorted by price.

Example 1: "Find me the cheapest price for the Chrysler Sebring Model of the year 2000".

Example 2: "Find me the cheapest price for Toshiba, speed 400-600 MHz, Model 5205-S703".

### 3.1. Jade Platform

JADE (Java Agent Development Framework) is a software development framework directed at developing multi-agent systems and applications according to Foundation for Intelligent Physical Agents (FIPA)[1] standards for intelligent agents. It includes two main products: a FIPA-compliant agent platform and a package to develop Java agents. JADE has been completely coded in Java and an agent programmer, with the objective to exploit the framework.

"It is an object-oriented (OO) platform, including an object database engine. Its different approach makes developing and deploying systems straightforward and easily. The normal industry approach is to use several tools from various vendors. With JADE we only need to use a single toolkit and it also makes simpler to introduce system changes. In this way, because this single toolkit comes from only one source, we prevent the so famous integration difficulties of using numerous tools from multiple vendors" (Tiziana Trucco, Fabio Bellifemine, Giovanni Caire, Giovanni Rimassa, 2001)[2].

JADE enables complex systems to be developed and deployed quickly. Programmers developing with JADE have considered that the speed by which applications can be developed gives it a clear advantage, by an architecture that enables systems to be altered without several of the complexities related to system modification.

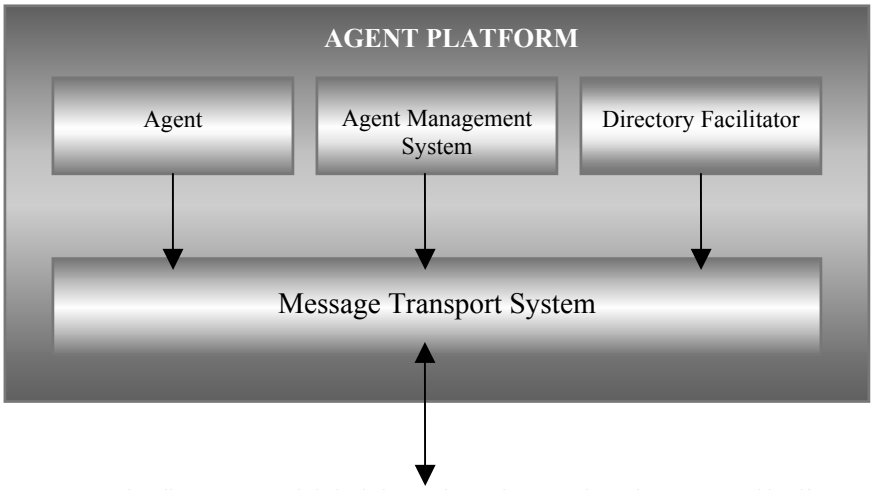
### 3.2. Database

*The Agent Platform*

The standard model of an agent platform, as defined by FIPA, is represented in the Figure 3.

[1] FIPA is a non-profit organisation aimed at producing standards for the interoperation of heterogeneous software agents.

Figure 3 - Reference architecture of a FIPA Agent Platform



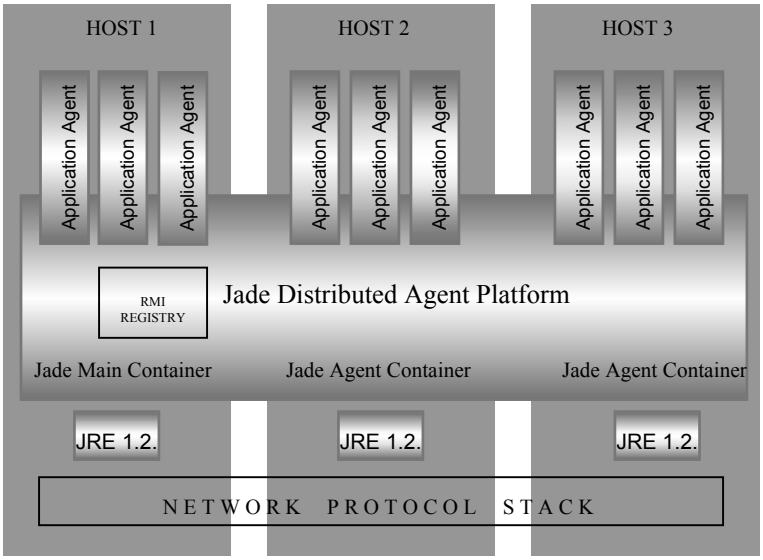

Source: http://www.ryerson.ca/~dgrimsha/courses/cps720/Resources/JADE/programmersguide.pdf

The Agent Management System (AMS) is the agent who holds supervisory control over access to and use of the Agent Platform. Only one AMS will live in a single platform. The AMS supplies white-page and life-cycle service, maintaining a directory of agent identifiers (AID) and agent state. Each agent must register with an AMS with the objective to obtain a valid AID. The Directory Facilitator (DF) is the agent who supplies the default yellow page service in the platform. The Message Transport System, also called Agent Communication Channel (ACC), is the software component controlling all the exchange of messages within the platform, including messages to/from remote platforms.

JADE conforms with this reference architecture and when a JADE platform is launched, the AMS and DF are immediately created and the ACC module is set to allow message communication.

According to Figure 4, the agent platform can be divided on various hosts. Only one Java application, and hence only one Java Virtual Machine (JVM), is executed on each host. Each JVM is a primary container of agents that supplies a complete run time environment for agent execution and enables various agents to concurrently execute on the same host. The main-container, or front-end, is the agent container where the AMS and DF lives and where the RMI registry, that is used internally by JADE, is created. The other agent containers, instead, connect to the main container and provide a complete run-time environment for the execution of any set of JADE agents.

Figure 4: Jade Agent Platform distributed over several containers




Source: http://www.ryerson.ca/~dgrimsha/courses/cps720/Resources/JADE/programmersguide.pdf

*Jade Features*

We present a list of features that JADE offers to the agent programmer:

- Distributed agent platform. The agent platform can be divided among various hosts (supplied they can be connected via RMI). Only one Java application, and hence only one Java Virtual Machine, is executed on each host. Agents are implemented as Java threads and live within *Agent Containers* that supply the runtime support to the agent execution.
- Graphical user interface to administer several agents and agent containers from a remote host.
- Debugging tools to help in building multi agents applications based on JADE.
- Intra-platform agent mobility, enclosing state and code of the agent.
- Support to the execution of several, parallel and concurrent agent activities through the behaviour model.
- FIPA-compliant Agent Platform, which encloses the AMS (Agent Management System), the DF (Directory Facilitator), and the ACC (Agent Communication Channel). All these three components are automatically activated at the agent platform launch. Several FIPA-compliant DFs might be launched at run time with the objective of implementing multi-domain applications, where a domain is a rational set of agents, whose services are passed through a common facilitator. Each DF inherits a GUI and all the standard capabilities determined by FIPA (i.e. capability of registering, deregistering, modifying and searching for agent descriptions; and capability of "federating" within a network of DF's).
- Effective dispatch of ACL messages within the same agent platform. Actually, messages are transferred encoded as Java objects, rather than strings, with the objective to prevent marshalling[2] and unmarshalling procedures. When crossing platform limits, the message is automatically transformed to/from the FIPA compliant syntax, encoding, and transport protocol. This transformation is clear to the agent implementers that only need to deal with Java objects.
- Library of FIPA interaction protocols.

[2] Marshalling: To compact the values of several variables, arrays, or structures into a single contiguous block of memory; copying values out of a block of memory is called unmarshalling. In most message passing systems, data must be marshalled to be sent in a single message.

- Automatic registration and deregistration of agents with the AMS.
- FIPA-compliant naming service: at launch agents get their GUID (Globally Unique Identifier) from the platform.
- Support for application-defined content languages and ontologies.
- InProcess Interface to enable external applications to launch autonomous agents.

## 4. THE ARCHITECTURE OF THE MBAOS SYSTEM

The MBAOS system is based on the Shopping bot (comparison-shopping) in which buyers (users) approach suppliers of products (bargaining Web sites) with their requirements. The MBAOS system involves a buyer, a Mobile Bargain Agent (MBA) for the buyer, and suppliers.

A buyer who is interested in making a comparison-shopping creates a MBA, provides it with an itinerary of supplier Web sites, through an user interface specifies criteria for the comparison-shopping, and dispatches the MBA to the potential suppliers. The MBA visits each supplier Web site, searches the product according to the buyer's criteria, and returns to the buyer site (User Home) with the best price it finds. The buyer confirms the deal and proceeds with the monetary transaction (this phase will be implemented in the future).

### 4.1. The life cycle of the Mobile Bargain Agent (MBA)

The life cycle of the Mobile Bargain Agent (MBA) is shown in the Figure 5 as follows:

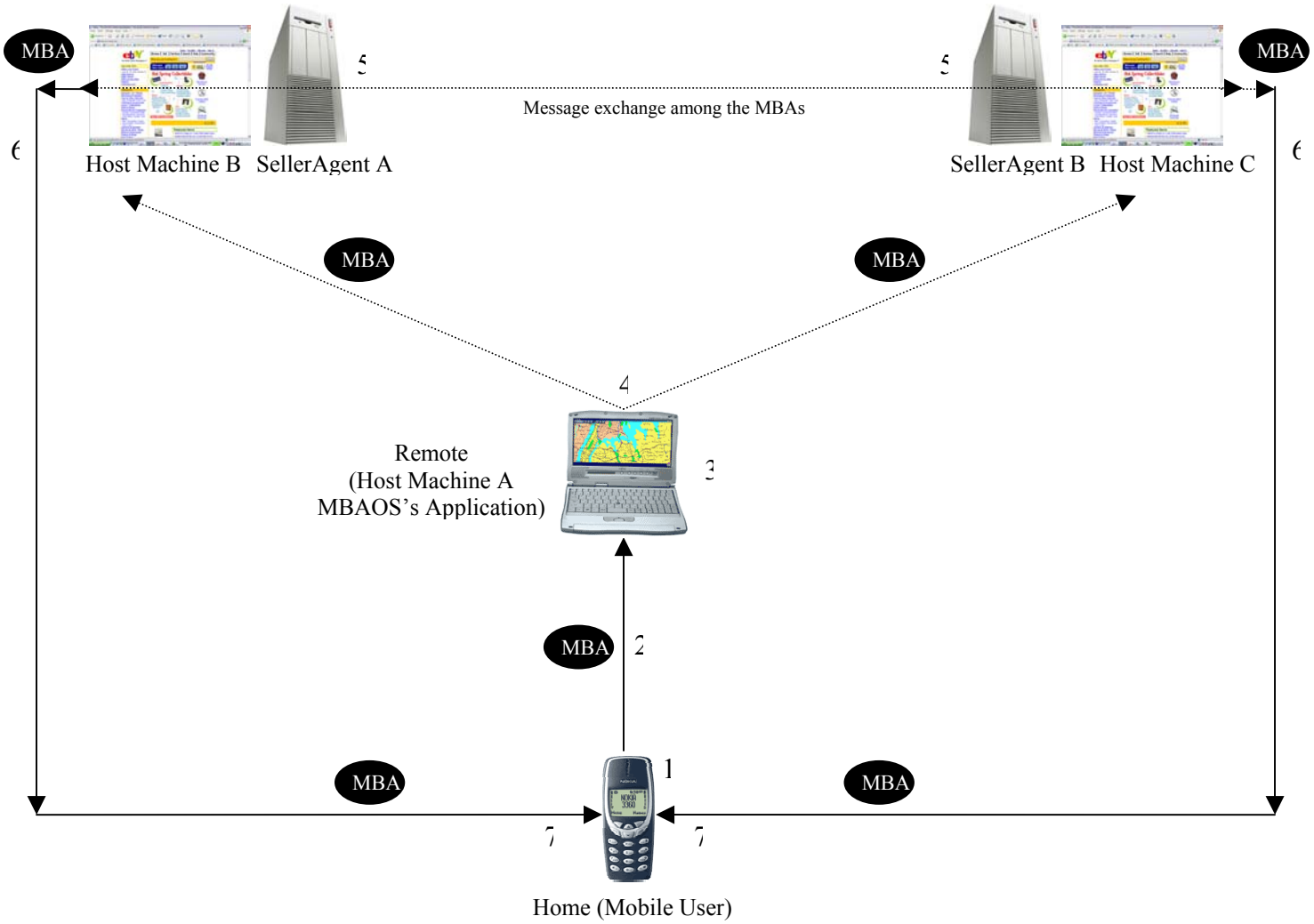
1. The mobile agent is created in the Home Mobile Device.
2. The mobile agent is dispatched to the MBAOS Host Machine A for execution.
3. The agent executes on Host MBAOS Machine A (Remote). In fact, the MBA has a set of tasks, and each task might be performed at the server (remote), which provides a local object to interact with the mobile agent.

4. After execution the agent is cloned to create two copies. One copy is dispatched to Host Machine B and the other is dispatched to Host Machine C.

5. The cloned copies execute on their respective hosts.
6. After execution, Host Machine B and C send the Mobile Bargain Agent received by them back to the User Home Machine.

7. The Home Machine retracts the agents and the data brought by the agents is analyzed. The agents are then disposed.

Figure 5: The Life Cycle & Architecture of the MBAOS' System.




MBA MOBILE BARGAIN AGENT

From this we might notice that our MBA experiences the following events in its life cycle:

*Creation:* a new agent is born and its state is initialized.
*Dispatch:* an agent moves to a new host.
*Cloning:* a twin agent is born and the current state of the original is duplicated in the clone.
*Deactivation:* an agent is put to sleep and its state is stored on a disk of the host.
*Activation:* a deactivated agent is brought back to life and its state is restored from disk.
*Retraction:* an agent is brought back from a remote host along with its state to the home machine.
*Disposal:* an agent is concluded and its state is died.

## 5. EMPIRICAL EVALUATION OF THE MBAOS SYSTEM

Nowadays we are facing a significant and consistent supply of different agent technologies which are inundating the agent-based-world. Each of them achieves a diverse spectrum of purposes. The MBA is a single and useful mobile agent which navigates effectively through a diversity of bargaining Web sites, extracts the relevant information (only the best price) making use of a mobile agent technology – a key technology in the creation of an open, active, and heterogeneous network environment.

We consider two main factors to provide an effective user experience with our MBA:

- Simplicity/Conviviality: provides very simple navigation and makes use of simple-only data retrieval (best prices).

- User Mobility: enables users to access a bargain Web site from its mobiles

devices, it means that a user might dispatch an MBA from, say, a cellular phone.

Here are some fundamental characteristics (Lange & M. Oshima, 1998)[3] that we think are good reasons to use the mobile agent technology in our MBAOS system, as follows:

- Mobile Agents reduce the network load

Distributed systems often count on communications protocols that are associated with multiple interactions to accomplish a given task. The result is a lot of network traffic. MAs enable us to package a conversation and dispatch it to a destination host, where the interactions may occur locally (see Figure 1).

- Mobile agents surmount network latency

Critical real time systems, such as robots in manufacturing processes, have to respond in real time to changes in their environments. Controlling this sort of system through a factory network of a significant size involves significant latencies. For critical real-time systems, such latencies are not admissible. MAs provide a solution, because they can be dispatched from a central controller to act locally and directly execute the controller's directions.

- Mobile agents encapsulate protocols

When data are exchanged in a distributed system, each host possesses the code that implements the protocols needed to code outgoing data and interpret incoming data. However, as protocols develop to adjust new demands for efficiency or security, it is a cumbersome task to upgrade protocol code adequately. Hence, protocols often become a legacy problem. MAs, on the other hand, can shift to remote hosts to set up "channels" based on proprietary protocols.

- Mobile agents execute asynchronously and autonomously

As we might well know, mobile devices such as cellular phones, PDAs (Personal Digital Assistant, etc.) often must count on expensive or fragile network connections. Tasks that demand a continuously open connection between a mobile device and a fixed network likely will not be economically or technically feasible. To solve this problem, tasks can be embedded into MAs, which then be dispatched into the network. After being delivered, the MAs become self-sufficient of the creating process and might operate asynchronously and autonomously. The mobile device can reconnect at a later time to gather the agent.

- Mobile agents adapt dynamically (migrate and cloning)

MAs have the ability to "feel" their execution environment and react autonomously to changes. Multiple mobile agents possess the single ability to distribute themselves among the hosts in the network so as to preserve the optimal configuration for solving a particular problem.

- Mobile agents are naturally heterogeneous

Network computing is essentially heterogeneous in a hardware and software perspective. Since MAs are generally computer-and-transport-layer-independent and are dependent only on their execution environment, they provide good conditions for seamless system integration.

- Mobile Agents are robust and fault-tolerant

The ability of mobile agents to react dynamically to adverse situations makes it easier to build robust and fault-tolerant distributed systems. If a host is being shut down, all agents running on that machine will be notified and settled time to dispatch and continue their operation on another host in the network.

## 6. RELATED WORK

Much of the related agent work demands structured information of the kind found in a relational database. *The Internet Softbot* (O. Etzioni and Weld. A SOFTBOT-BASED INTERFACE TO THE INTERNET, CACM, 37(7):72-76, July 1994) is also capable to retrieve information from the rigid formatted output of UNIX commands like `ls` and Internet services like `netfind`. There are agents that investigate unstructured Web pages, but they accomplish it only in the context of the supported browse tasks (Pattie Maes and Robyn Kozierok, LEARNING INTERFACE AGENTS. In proceedings of AAAI-93, 1993), in which the agent tries to detect hopeful links by deducting the user's interests from his/her past browsing behaviour.

Differently, "Shopping bot" systems learn about their user's interests, instead of learning about the external resources they might have access. The

exception is the Internet Learning Agent, ILA (Mike Perkowitz and Oren Etzioni. Catgeory translation: LEARNING TO UNDERSTAND INFORMATION ON THE INTERNET. In proceedings of 15th Int. Joint Conf. on AI, 1995). ILA understands external information sources by demonstrating their output regarding internal categories. ILA also learns by querying an information source with known objects and examining the relationship of output tokens to the query. For example, it queries the Université de Montréal personnel directory with Esma Aimeur and demonstrates the output token 343-6794 as her phone number.

BargainFinder (http://crowston.syr.edu/papers/icis97present/wacked/agent.htm) is an agent that searches for the lowest prices on CDs among on-line music stores. However, it does not allow users to buy and/or sell goods over the Internet. In fact, BargainFinder is not an AI program, whereas Shopping bot count on AI techniques such as heuristic search[3], pattern matching and inductive learning.

## 7. CONCLUSION AND FUTURE WORK

In this paper we have discussed how a Mobile Bargain Agent can facilitate searching activities on behalf of the online shopping user in a simpler and easier way providing at the same time a user mobility capability. The use of mobile agents introduces much more dynamic to the provision of online shopping, and therefore increases the speed of the data retrieval according to the user preferences.

While our empirical evaluation have shown that the Mobile Bargain Agent (MBA) is useful, it has also some limitations that we will address in future work such as:

- MBA need to do more detailed analysis of product descriptions.
- MBA is limited to the comparison-shopping task, and it should be extended to the other tasks such as find product specification, make comprehensible recommendations, and notice pertinent special offers and discounts.

[3] Heuristic search : The basic idea of heuristic search is that, rather than trying all possible search paths, you try and focus on paths that seem to be getting you nearer your goal state. Of course, you generally can't be sure that you are really near your goal state - it could be that you'll have to take some amazingly complicated and circuitous sequence of steps to get there. But we might be able to have a good guess. Heuristics are used to help us make that guess.

## 8. REFERENCES

[1] Robert Doorenbos, Oren Etzioni & Daniel Weld, A SCALABLE COMPARISON-SHOPPING AGENT FOR THE WORLD-WIDE-WEB, Technical Report UW-CSE-96-01-03, Detarment of Science and Engineering, University of Washington, 2003.

[2] Tiziana Trucco, Fabio Bellifemine, Giovanni Caire, Giovanni Rimassa (University of Parma), JADE PROGRAMER GUIDE, last update: 4-September-2001. JADE 2.4 [http://www.ryerson.ca/~dgrimsha/courses/cps720/Resources/JADE/programmersguide.pdf]

[3] D. B. Lange & M. Oshima, « Programming and Deploying Java Mobile Agents with Aglets », 1998, Addison-Wesley Longman, Inc.