Distributed Mobile Computing: Development of Distributed Applications Using Mobile Agents

Carlos Bobed, Sergio Ilarri, and Eduardo Mena
IIS Department, University of Zaragoza, Zaragoza, Spain
{cbobed, silarri, emena}@unizar.es

Abstract— *There exist distributed scenarios in which the need for dynamism, mobility, and adaptivity, has to be addressed with highly dynamical approaches. These scenarios present different challenges and difficulties: efficient access to heterogeneous and distributed data sources, dynamic load balancing, unstable connections and communication failures, etc. So, different approaches and middleware have appeared to tackle these challenges and help the developer of distributed applications. In particular, mobile agent technology can provide significant advantages for the development of applications in these contexts.*

In this paper, we emphasize the benefits that mobile agents can provide to distributed systems by illustrating them with real distributed systems that we have developed in our research group. Mobile agents play a key role in all the scenarios presented: 1) They are able to track the relevant data and computers involved in a certain computation, 2) they move the processing tasks wherever they are needed to ensure a good performance, and 3) they adapt themselves to changes in their execution environment.

Keywords: Mobile Computation, Distributed Applications, Mobile Agents

1. Introduction

Distributed computing can provide several advantages, such as scalability, fault-tolerance, and load balancing. Besides these benefits, in many cases such a distribution is simply the only possible approach (e.g., when we have to access to several independent and distributed data sources). Indeed, many scenarios would be unmanageable from a centralized point of view due to performance and/or processing requirements. Finally, in many contexts the data needed to perform the processing tasks are distributed, and so a distributed solution for processing those data seems natural and could be more efficient.

However, distributed computing also brings a number of challenges. For example, some mechanism is needed to keep track of the computers which are relevant for a certain computation, mostly if this set of computers changes along time; e.g., different computers can store data that could be needed at different moments during the computation, or some computers may become more relevant/irrelevant due to load balancing issues. Therefore we need efficient mechanisms to select those relevant computers, involve them in the computation, and retrieve the data needed. Those computers should provide suitable services for the distributed computation needed, or else we will need to implement the computation required by means of multiple remote interactions which may retrieve large amounts of data. Moreover, communications could fail at any time, especially if our distributed environment includes wireless networks, and so our application should be reliable enough in order to take appropriate action if such problems arise. And these are only some examples of the difficulties that a developer of applications for distributed environment faces.

To try to overcome these difficulties, different approaches have appeared, which basically propose some middleware to ease the tasks needed to develop applications for distributed environments. In particular, mobile agent technology provides important advantages in these contexts. Mobile agents [1], [2], [3] are programs that execute in contexts called *places*, hosted on computers or other devices, and can autonomously travel from *place* to *place* resuming their execution there. Thus, they are not bound to the computer where they were created; instead, in run-time they can move freely across different computers and devices. Thus, they provide interesting features for distributed environments, thanks to their autonomy, adaptability, and capability to move to remote computers.

In this paper, we analyze some common challenges and difficulties that the development of distributed systems implies and how they can be addressed by using mobile agents. We point out the benefits of using them and illustrate these benefits with real distributed systems that we have developed in our research group. Mobile agents play a key role in all the scenarios presented. Whereas the advantages of mobile agents have been highlighted in other previous works, our contribution resides in sharing our experiences using this technology to build different types of applications for distributed environments.

The structure of the rest of the paper is as follows. In Section 2 we analyze some difficulties that appear in most distributed environments. In Section 3 we overview the main features of mobile agent technology. In Section 4 we present as use cases the basic aspects of three systems that we have developed using mobile agent technology for distributed environments. In Section 5 we summarize the benefits obtained out of using mobile agents in the previous use cases. Finally, in Section 6 we draw some conclusions.

2. Some Common Challenges in Distributed Computing

The adoption of a distributed paradigm allows us to develop flexible and scalable systems. However, this does not come at no cost. Dealing with distributed systems and data introduces several challenges and difficulties:

- *Need for appropriate development and programming abstractions*. From a conceptual point of view, the distribution of tasks and data introduces new issues to be taken into account: communication management, synchronization, concurrency, the need to process tasks in different nodes, etc. Developers have to make an extra effort to bear in mind all these issues and use the new mechanisms that are available.
- *Need for mechanisms to use the network efficiently*. Distributed computing makes exhaustive use of networks, as they are the base on which it is built. The different characteristics of network connections and their reliability in terms of stability and bandwidth assurance might compromise our system. Moreover, the efficient usage of the network is crucial to make our system scalable.
- *Need for flexible and automatic reconfiguration of the distributed system*. When distributing tasks, they will be executed in computers or devices that are nodes of a network. It should be easy to add new nodes to the system or to remove existing nodes when they are not available anymore.
- *Need for approaches that adapt to the current environment conditions*. In order to build a robust distributed system, we need to consider possible changes that may affect the performance or the success of the tasks that are performed by the system. For example, the availability of the nodes might change along time due to many reasons (e.g.: overloaded computers, physical disconnections, etc); fault tolerance in distributed systems implies to be able to react to these changes allowing dynamic adaptation to the new environment. Similarly, the communication costs may change along time, and in that case the system could try to adapt its behavior to keep its performance and accuracy.
- *Need for approaches that facilitate the distribution of tasks among the available nodes*. Even when we have a stable network and all the computing nodes are available, these nodes might differ regarding their resources (computation power, memory, etc), being some of them more suitable for some tasks than others. Optimizing the overall resource consumption is another important challenge.
- *Need for suitable techniques to perform an efficient data processing*. Processing distributed data also raises several concerns: It might not be feasible to deal efficiently with the heterogeneity of the data sources and the size of the data to be processed using traditional approaches.

Note that the above list is not meant to be an exhaustive enumeration, but just a list of the main challenges that we all have found in our experience.

3. Mobile Agents

Mobile agents [1], [2], [4], [3], [5] are programs that execute in contexts called *places* and can autonomously travel from place to place resuming their execution there. Mobile agents are not bound to the computer where they were created; instead, they can move freely across places in different computers or devices. The only requirement is to have a certain *mobile agent platform* [6] that allows their execution in those computers.

Thanks to their mobility, mobile agents offer many interesting benefits [2], such as:

1) *They encapsulate tasks*. As they can move to remote computers to achieve their goals, they avoid the need for installing one specialized server process, on every machine, to provide access to *just one required service*. Instead, *only one server process* (the mobile agent platform) *allows many different agents to travel to that computer carrying the required functionalities to provide such services*. This feature results in a great flexibility to develop applications for distributed environments.
2) *They reduce the network load*. Thus, a mobile agent can travel where the data are and access to them locally, filtering out the data that do not need to be sent over the network. Moving the computation to the data, instead of the other way around, can save many resources when huge volumes of data must be analyzed. Moreover, network connections are not needed while agents process data directly on remote computers, the network is used only while the mobile agent moves to another computer.
3) *They overcome network latency*. They can move to another computer in order to optimize the response time. The ability of mobile agents to reduce the latency is critical for real-time systems, such as robots in manufacturing processes [2]. Thus, agents can be dispatched from a central controller to a certain computer to control the robot locally, avoiding multiple remote interactions and, therefore, saving network communications.
4) *They are asynchronous and autonomous*. In traditional synchronous client/server architectures, the client must keep the connection active while its request is being

processed by the server. If the connection fails, the client has to send the request to the server again, which will process it from the beginning. Alternatively, a mobile agent does not need to keep contact with its source computer while performing its tasks. This is particularly important with mobile devices, which usually communicate via an expensive and unreliable wireless connection. Thus, a mobile device can send a mobile agent to a computer in the fixed network and then go off-line. The agent becomes independent of its originating device, and thus it is a flexible way of dispatching a task into the network. When the device re-establishes the connection, it can collect the mobile agent or its results.

5) *They adapt dynamically to their environment* [2]. They are able to sense the environment and can be programmed in order to react autonomously to adapt themselves to changes. For example, they could travel to another computer when the current computer is overloaded, for load balancing.

Furthermore, they also exhibit a *good performance* compared with the traditional client/server approach. For example, [7] evaluates the savings introduced by mobile agents when interacting with a remote database in a wireless environment. In [8] the benefits of mobile agents in information retrieval are analyzed (documents are filtered at the remote site). Finally, in [9] several strategies to download files from a wired network are evaluated, and it is shown how mobile agents exhibit the same performance as CORBA and FTP approaches.

Before concluding this section, it should be emphasized that mobile agents are a nice programming abstraction for many application scenarios where the processing should move from one computer to another. Besides, mobile agent platforms provide different services (such as a transportation/mobility service, a communication service, and a security service) that facilitate the development of distributed applications. For example, mobile agent platforms offer *location transparency*, which offers mobile agents an easy to use method to communicate with other agents independently of the computers or devices where they are executing at that moment [10].

4. Use Cases

In this section, we overview three different systems where we have applied mobile agents. They are three different scenarios that include mobile computing, distributed data processing, and pervasive computing.

4.1 LOQOMOTION

In the mobile computing field, *location-based services* are a subject of intensive research [11]. They provide value-added by considering the locations of the mobile users to offer customized information. In particular, many efforts have been focused on the efficient processing of location-dependent queries [11], which are queries whose answer depends on the locations of objects (e.g., retrieve the taxi cabs within a certain area around a mobile user).

LOQOMOTION (*LOcation-dependent Queries On Moving ObjecTs In mObile Networks*) [12] is a distributed location-dependent query processing system whose architecture is based on mobile agents. The system deploys a network of agents to perform the query processing over a distributed set of computers, called *proxies*, which manage information about objects moving within different geographic areas: each proxy is in charge of one *proxy area* and so the data management tasks are distributed.

We use the scenario shown in Fig. 1 to illustrate the different steps in the agent network deployment. Moving objects (e.g., cars) are represented as circles in the figure, and the dashed ellipses represent the different proxy areas. Each proxy has a *Data Management System* or *DMS* (e.g., a Database Management System) that stores information about the moving objects within its proxy area and can answer queries about them. In this scenario, we assume for illustration purposes that the user submits a query that must retrieve the white objects located within a certain distance of each black object, that is, a query that retrieves the white objects within the circular *relevant areas* shown around the black objects. As the objects are continuously moving, the answer to the query must be updated continuously, with a certain *refreshment frequency*.

As shown in Figure 1, there are three main types of agents deployed over the fixed network to process the query and keep the answer up-to-date: the *MonitorTracker* mobile agent, the *Tracker* mobile agents, and the *Updater* agents. We briefly explain in the following how the agents deploy themselves on the fixed network and their roles:

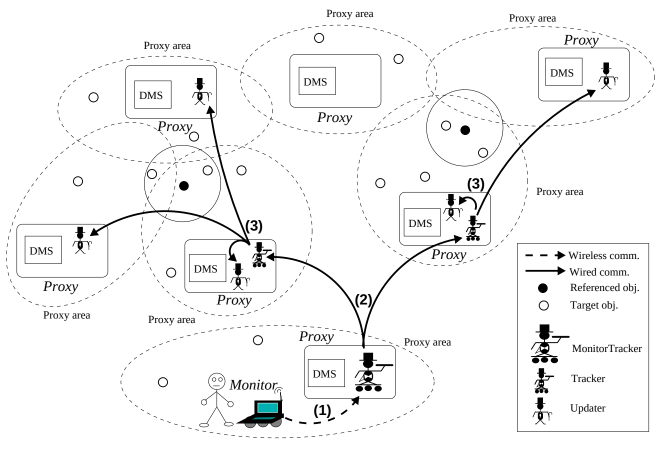

Fig. 1: Tracking moving objects using mobile agents: agent network deployment

1) A MonitorTracker mobile agent is created on the proxy that provides the mobile device of the user (called

monitor in the following) with coverage (i.e., the monitor is within its proxy area). The MonitorTracker performs three main tasks: a) to follow the monitor wherever it goes (moving from proxy to proxy with the goal of staying *close* to the user), b) to store the data requested by the user in case of disconnection of the monitor, and c) to efficiently refresh the data that have to be presented to the user, minimizing the wireless communications with the monitor. For the third task, it creates a Tracker mobile agent to track the location of each reference object and to process the query constraints related to such a reference object.

2) Each Tracker moves to the proxy which handles the location data of its reference object, and performs three main tasks: a) to stay on the proxy that manages the location of its reference object, traveling from proxy to proxy when needed in order to always execute on the proxy whose proxy area contains its reference object; b) to detect and process new locations of its reference object by querying *locally* the corresponding DMS at its proxy; and c) to detect and process, with the help of Updater agents, the locations of target objects (the white objects in the figure) that satisfy the query constraints where such a reference object appears.
3) The Tracker agent creates one Updater agent on each proxy whose area intersects the area of interest around the reference object. The idea is to keep an eye on any possible way to enter that area. Thus, Updater agents are initialized with a certain local query that they have to execute to retrieve the objects within the interesting area. This query has to be updated periodically by the Tracker with the current location of its reference object.

The data retrieved by the Updaters must be communicated through the network of agents up to the mobile device and are processed and correlated along the way. It should be stressed that the mobility of the moving objects require the mobile agents to perform two important tasks: 1) to keep themselves ready to obtain an answer according to the current locations of the interesting objects, and 2) to update the answer to the query automatically.

In this system, mobile agent technology provides us with the right mechanisms to process location-dependent queries in a distributed and efficient manner. The different query processing tasks must *move* from proxy to proxy following their relevant data, which is naturally achieved by packaging those processing tasks as mobile agents. Summing up, the use of mobile agents in LOQOMOTION allows: 1) to track the positions of relevant objects efficiently, 2) to optimize the wireless communications, and 3) to support the distributed query processing efficiently.

4.2 Software Retrieval System

The *Software Retrieval System* (*SRS*) [9] is a service that allows mobile users to find, retrieve and install software. This service presents two main features: 1) *Semantic Search*: the service allows users to express their software requirements at a semantic level and helps them to browse customized software catalogs in order to select the adequate software. For that, the service makes use of an ontology (specifically created for the service) that contains a semantic description of software available at different repositories, and so makes transparent for users most of the technical details related to the software retrieval task. 2) *Analysis of user behavior*: the system “observes" users information requests in order to anticipate their needs and even learns from its mistakes to improve its future behavior with those users. So, it offers a customized and adaptable service to mobile users, putting a special emphasis on optimizing communication time.

The SRS takes part of ANTARCTICA [13], a system that provides users of wireless devices with a new environment that fulfills some of their data management needs. ANTARCTICA follows the widely accepted architecture for mobile computing [14] and so it deals with users of wireless devices and the GSN (Gateway Support Node) situated at the fixed network (i.e. the proxy that provides services to wireless users). Each GSN provides its users with different services being the SRS just one of them.

In Figure 2, we see the four main agents that participate in the service:

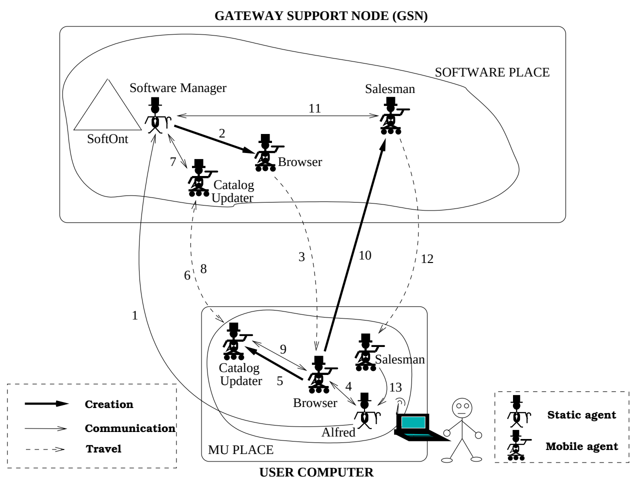

Fig. 2: Main architecture for the Software Retrieval Service

- *Alfred*, the user agent situated at the user device, which is an efficient majordomo that serves the user and is on charge of storing as much information about the user device and the user herself/himself as possible.
- The *Software Manager* agent (situated at the closest

GSN[1]), that prunes the software ontology taking into account the user requirements.

- The *Browser* agent, which is created at the GSN and then it moves to the user computer to help her/him to navigate the pruned ontology and select the wanted software.
- The *Catalog Updater* agent, that retrieves new information requested by the user. It is created by the Browser to update the user software catalog.
- The *Salesman* agent, which carries the selected program to the user computer and installs it whenever possible.

Their interactions are summarized next. The user agent Alfred requests a piece of software (step 1 in Figure 2) after a user request. For the creation of the user software catalog, the Software Manager consults the SoftOnt ontology to obtain a catalog of the available software that satisfies the needs expressed by Alfred (on behalf of the user), that is, the Software Manager is capable of obtaining customized metadata about the available software. Then, the Software Manager agent creates the Browser agent (step 2) and provides it with the user software catalog. The Browser agent travels to the user device (step 3) and interacts with the user in order to help him/her to browse the catalog of software (step 4); new information requested by the user will be retrieved by the Catalog Updater agent which is created by the Browser (step 5) to update the user catalog with software (step 9). The process of updating the user catalog could be done by traveling to the software place and interacting with the Software Manager (steps 6, 7, and 8). When the browsing process ends because the user finally chooses a concrete piece of software, the Salesman agent is created by the Browser (step 10) which first consults the Software Manager concerning the selected software (step 11), and then carries the program selected by the user to his device (step 12) in order to install it (step 13).

Summing up, two main tasks are performed to provide the service: 1) to prune the ontology in order to present to the user a software catalog containing only that part related to her/his requirements (to avoid confusing the user and overloading her/his computer with irrelevant information), and 2) to attend user refinements on such a catalog trying to predict the future behavior of the user (to minimize network communications). The agents responsible for those tasks are the Software Manager and the Browser agents, respectively.

The use of mobile agents in the SRS allows to:1) process locally the information in the data sources, tailoring the information that is presented to the user; 2) optimize the volume of data that is to be sent over the net, and 3) avoid burdening the user device, as most of the computations are performed in the fixed network.

[1]There exists one Software Manager agent on each GSN.

4.3 Nescrem

Nescrem[2] [15] is an infrastructure to achieve an "ubiquitous computer". The objective is that the user should not be attached to a physical computer in a physical location, but s/he would be accompanied by it. Nescrem detects the location of the user and presents his/her environment on the closest computer. All the data (and the state of the different applications) "move" wherever the user goes, giving him the impression of gaining control over a computer just by getting close to it. The system finds the optimal balance between remote and local access to user files and applications, to reduce the use of remote resources whenever possible.

The mobile infrastructure that manages the movements and the data of the user is based on agents. We distinguish three different kinds of agents: *Guardian*, *MovementManager*, and *UserAgent* (see Figure 3). The tasks of each one are the following:

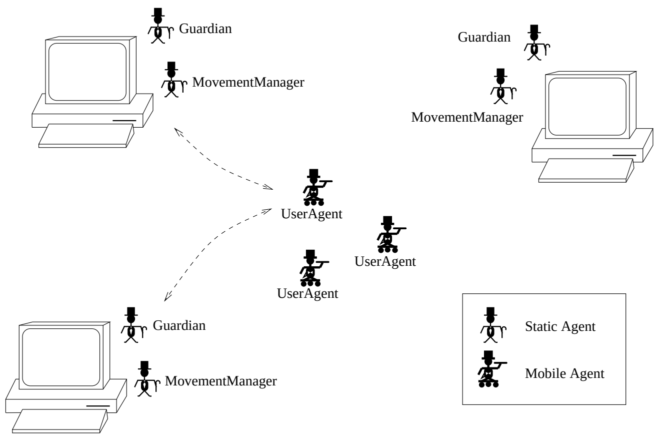

Fig. 3: Mobile infrastructure design

- The Guardian agent manages the resources of the host on which it resides (there is one Guardian agent on each host of our system). It physically logs the users in and out, maintains the user resource information up to date, and manages the host from our mobile system point of view. Guardian agents share the information about which Guardian is in charge of each user's files.
- The MovementManager agent coordinates the movements of the agents (UserAgents) and the communication of information across Guardian agents. They detect when a user is leaving or approaching a computer and decide whether to log that user in or out the nearest computer, using the services provided by the Guardian agents. There is one MovementManager agent on each host of our system.
- The UserAgent is a mobile agent that follows the user from one host to another, carrying with it the user

[2]Nescrem means "Not Remote Desktop", in Spanish "No EScritorio REMoto".

credentials, configurations, and resource information. It also carries information about the last accessed files to serve as a cache and make it possible to speed up the following accesses. In our system, there is one UserAgent for each mobile user.

On one hand, the use of mobile agents in Nescrem enhances the adaptability and the resilience against node failures. The mobile agents impersonate users in the system and follow them wherever they go, along with their credentials, files, etc. This makes the system independent of particular node. On the other hand, it also enhances the flexibility of the system as it enables the users to carry with them their configurations and modify them according their needs.

Finally, from the programming point of view, the agent abstraction is specially useful: 1) in the design phase, as it provides us with high level software components (agents) which are almost directly mapped to the real implementation and this also eases the deployment; and 2) in the development stage, the different communication and agent trace mechanisms provided by the mobile agent platforms eases the tasks of implementing the system.

5. Summary of Benefits of Using Mobile Agents

In this section, we summarize the benefits obtained out of using mobile agents in the distributed systems described in Section 4 (see Table 1):

Feature	**LOQOMOTION**	**SRS**	**Nescrem**
Agents Abstraction	√	√	√
Efficient Network Usage	√	√	
Flexibility	√		√
Adaptability	√	√	√
Efficient Distributed Processing	√	√	
Efficient Data Management	√	√	√

Table 1: Benefits obtained out of using mobile agents

- *Agents Abstraction* refers to the benefits obtained by considering the design of the system at the abstraction level that mobile agents provide.
- *Efficient Network Usage* refers to whether the use of mobile agents has helped to optimize the usage of the network in some way.
- *Flexibility* refers to the possibility to be able to change easily the configuration of the system, for example in such a way that it can react automatically to changes in the topology of the network of nodes (addition/loss of nodes, for example).
- *Adaptability* refers to the ability of the system to react to changes in the environment, adapting itself for example to changes in the delays of the network communications.
- *Efficient Distributed Processing* refers to whether the use of mobile agents has allowed to distribute the computations in an efficient way.
- *Efficient Data Management* refers to whether mobile agents have helped to achieve efficient data processing.

The mobile agent technology has proved to be useful in the design phases, as it encapsulates many low-level details (communication, synchronization, etc). Thinking in terms of mobile agents eases the complexity of establishing the different tasks and distributing them.

The ability of mobile agents to travel to another agent-enabled computers makes systems that use them more flexible, dynamic, and adaptive. For example, LOQOMOTION and Nescrem take advantage of this feature to adapt themselves to the addition/loss of nodes in their computing networks. Adding a new proxy in LOQOMOTION or a new computer in Nescrem does not involves any extra configuration step and can be done in runtime.

LOQOMOTION and the SRS also obtain benefits in terms of global efficiency: They optimizes the network usage by enabling local access as much as possible; in LOQOMOTION the query processing is distributed and performed close to the relevant objects, and only the relevant data are processed, while in the SRS user software catalog are customized and managed locally on the user device, most of the times without accessing the network.

6. Conclusions

In this paper, we have analyzed some common challenges and difficulties that the development of distributed systems implies. Our experience on distributed information systems, focused on mobile and pervasive environments, tells us that mobile agent technology has to be taken into account when developing this kind of applications.

To sum up, the main advantages that we have experienced when using mobile agents in mobile and pervasive environments are:

- From the designing point of view, the mobile agent abstraction eases the communications and error recovery management and provides a natural way to develop different types of applications where processing tasks should move from one node to another.
- The use of mobile agents can lead to huge communications savings. This is very important particularly in mobile environments, where wireless communications can be slow, unreliable, and expensive, according to different parameters (scarce battery, low bandwidth, etc).
- The ability of encapsulating dynamic behavior in the agents eases the development of applications that have to react dynamically to changes in the environment.
- Their mobility, that is, their ability to travel to the place where they have to be executed makes them perfect for mobile and distributed scenarios.

Besides the use cases presented in this paper, we have also developed other systems using mobile agent technology, such as:

- A framework to test mobile computing applications [16]: The usage of mobile agents, among other advantages, allows distributed testing which increases the scalability and accuracy of the results.
- A system for adapting graphical interfaces to different user mobile devices [17]: The mobile agents adapt the GUI dynamically to the presentation capabilities of the device in which they are being executed.
- A Locker Rental Service [18]: This system allows a mobile user to rent physical disk space in a fixed network. A mobile agent is devoted to serve a particular user and stay close to him/her, keeping the locker in the closest location within the fixed network. This provides efficient access to the ubiquitous persistent storage space provided by the locker.
- A framework to monitor environments using vehicular networks [19]: The mobile agents jump from car to car as necessary to reach the area of interest and keep themselves in that area. This makes it possible to avoid relying on an expensive fixed infrastructure of sensors.

As future work, we are planning to use mobile agents to answer semantic queries in mobile scenarios, harvesting knowledge from different sources and using it to answer the user queries without burdening their mobile devices.

Acknowledgments

This work has been supported by the CICYT project TIN2007-68091-C02-02.

References

[1] D. Milojicic, F. Douglis, and R. Wheeler, *Mobility: processes, computers, and agents*. ACM Press/Addison-Wesley Publishing Co., 1999.

[2] D. B. Lange and M. Oshima, "Seven good reasons for mobile agents," *Commun. ACM*, vol. 42, pp. 88–89, March 1999. [Online]. Available: http://doi.acm.org/10.1145/295685.298136

[3] F. L. Bellifemine, G. Caire, and D. Greenwood, *Developing Multi-Agent Systems with JADE*. Wiley, 2007.

[4] C. Harrison, D. Chess, and A. Kershenbaum, "Mobile agents: are they a good idea?" in *Second International Workshop on Mobile Object Systems (MOS'97): Towards the Programmable Internet, Linz, Austria*. Springer, 1997, pp. 46–48.

[5] P. Braun and W. R. Rossak, *Mobile Agents: Basic Concept, Mobility Models, and the Tracy Toolkit*. Morgan Kaufmann, 2005.

[6] R. Trillo, S. Ilarri, and E. Mena, "Comparison and performance evaluation of mobile agent platforms," in *Third International Conference on Autonomic and Autonomous Systems (ICAS'07), Athens, Greece*. IEEE, 2007, p. 41.

[7] C. Spyrou, G. Samaras, E. Pitoura, and P. Evripidou, "Mobile agents for wireless computing: the convergence of wireless computational models with mobile-agent technologies," *Mobile Networks and Applications*, vol. 9, no. 5, pp. 517–528, 2004.

[8] R. S. Gray, D. Kotz, R. A. Peterson, J. Barton, D. Chacón, P. Gerken, M. Hofmann, J. Bradshaw, M. Breedy, R. Jeffers, and N. Suri, "Mobile-agent versus client/server performance: Scalability in an information-retrieval task," in *Fifth IEEE International Conference on Mobile Agents (MA'01), Atlanta, Georgia, USA*, vol. 2240. Springer, 2001, pp. 229–243.

[9] E. Mena, J. A. Royo, A. Illarramendi, and A. Goñi, "Adaptable software retrieval service for wireless environments based on mobile agents," in *International Conference on Wireless Networks (ICWN'02), Las Vegas, Nevada, USA*. CSREA Press, 2002, pp. 116–124.

[10] S. Ilarri, R. Trillo, and E. Mena, "SPRINGS: A scalable platform for highly mobile agents in distributed computing environments," in *Fourth International WoWMoM 2006 Workshop on Mobile Distributed Computing (MDC'06), Buffalo, New York, USA*. IEEE Computer Society, ISBN 0-7695-2593-8, June 2006, pp. 633–637.

[11] S. Ilarri, E. Mena, and A. Illarramendi, "Location-dependent query processing: Where we are and where we are heading," *ACM Computing Surveys, ISSN 0360-0300*, vol. 42, no. 3, pp. 1–73, March 2010.

[12] S. Ilarri, E. Mena, and A. Illarramendi, "Location-dependent queries in mobile contexts: Distributed processing using mobile agents," *IEEE Transactions on Mobile Computing (TMC), ISSN 1536-1233*, vol. 5, no. 8, pp. 1029–1043, August 2006.

[13] A. Goñi, A. Illarramendi, E. Mena, Y. Villate, and J. Rodriguez, "Antarctica: A multiagent system for internet data services in a wireless computing framework," in *NSF Workshop on an Infrastructure for Mobile and Wireless Systems, Scottsdale, Arizona, USA*, October 2001.

[14] E. Pitoura and G. Samaras, *Data Management for Mobile Computing*. Kluwer Academic Publishers, 1998.

[15] C. Bobed and E. Mena, "Towards the ubiquitous computer: A mobile agent approach," in *II Symposium on Ubiquitous Computing and Ambient Intelligence (UCAmI'07), Zaragoza, Spain*. Thomson, ISBN 978-84-9732-605-6, September 2007, pp. 143–151.

[16] S. Ilarri, E. Mena, and A. Illarramendi, "A system based on mobile agents to test mobile computing applications," *Journal of Network and Computer Applications, ISSN 1084-8045*, vol. 32, no. 4, pp. 846–865, July 2009.

[17] N. Mitrovic, J. Royo, and E. Mena, "Adus: Indirect generation of user interfaces on wireless devices," in *Fifteenth International Workshop on Database and Expert Systems Applications (DEXA'2004), Seventh International Workshop Mobility on Databases and Distributed Systems (MDDS'2004)*. IEEE Computer Society, ISBN 0-7695-2195-9, ISSN 1529-4188, September 2004, pp. 662–666.

[18] Y. Villate, A. Illarramendi, and E. Pitoura, "Keep your data safe and available while roaming," *International Journal of Mobile Networks and Application (MONET), Special Issue on Pervasive Computing*, vol. 7, no. 4, pp. 315–328, August 2002.

[19] O. Urra, S. Ilarri, E. Mena, and T. Delot, "Using hitchhiker mobile agents for environment monitoring," in *Seventh International Conference on Practical Applications of Agents and Multi-Agent Systems (PAAMS'09), Salamanca, Spain*, ser. Advances in Intelligent and Soft Computing, vol. 55. Springer Verlag, ISSN 1867-5662, ISBN 978-3-642-00486-5, March 2009, pp. 557–566.