PUMAS: a Framework based on Ubiquitous Agents for Accessing Web Information Systems through Mobile Devices

Angela Carrillo-Ramos, Jérôme Gensel, Marlène Villanova-Oliver, Hervé Martin
Laboratoire LSR - IMAG
B.P. 72 - 38402 Saint Martin d’Hères, CEDEX, France
33 4 76 82 72 80
{carrillo, gensel, villanov, martin}@imag.fr

ABSTRACT
In this paper, we describe PUMAS [3], a framework based on Ubiquitous Agents for accessing Web Information Systems (WIS) through Mobile Devices (MD). The objective of PUMAS is to adapt the information delivered to a nomadic user (who often changes her/his location) according to her/his preferences, intentions and history in the system and, to the limited capacities of her/his MD. We use some AXML (Agent Unified Modelling Language) [11] diagrams for representing the messages exchange between the agents and the PUMAS components.

Categories and Subject Descriptors
I.2.11 [Artificial Intelligence]: Distributed Artificial Intelligence - Intelligent agents.
H.3.3 [Information Storage and Retrieval]: Information Search and Retrieval - Information filtering, Search process.

General Terms
Design.

Keywords
PUMAS, Ubiquitous Agent, Mobile Agent, Framework, WIS.

1. INTRODUCTION
The access to information through Web Information Systems (WIS) has changed a lot due to: the technical advances in Mobile Devices (MDs), e.g. PDA, phones, laptop…; the characteristics of the MD (e.g., reduced capacities like small size of screen, memory, hard disk…); the inherent mobility of nomadic user; the multimedia nature of exchanged data, etc. Additionally, when a nomadic user (user who often changes her/his location) searches, using her/his MD, for information, she/he could get in response a too large quantity of information which is not always relevant for her/him and which is not always supported by her/his MD.

For delivering only the more relevant information (i.e. “the right information in the right place at the right time”) to the nomadic user, it is necessary to take into account that her/his location can change her/his information needs. The work of Cao et al. in [2] presents Location dependent queries which are executed in order to get information according to the current location of the user. For obtaining the user’s location, it is possible to use a GPS device and/or methods like the ones proposed in Nieto-Carvajal et al.[9]: the Signal Strength, SNMP (Simple Network Management Protocol) or the Local Access to the MAC address of the access point.

Additionally, many functional and technical issues have to be considered when modelling a WIS. The system is not only supposed to answer users’ requests, but it should also ideally provide them with information adapted to their needs, constraints and preferences. The underlying challenge for WIS designers is to provide WIS users with useful information based on an intelligent search and a suitable display of the delivered information. In order to reach this goal, the Multi-Agent Systems (MAS) is an interesting approach. El Fallah-Sehrouchni et al. in [4] have defined a MAS as a credible paradigm to design distributed and cooperative systems based on the intelligent agent technology.

The W3C [16] defines an agent as “a concrete piece of software or hardware that sends and receives messages”. These messages can be used for accessing a WIS and for exchanging information. Ramparany et al. [13] have shown the interest of MAS when the Internet is used to access and exchange information through new MDs (“smart devices” like PDA, phones, laptops…). In this case, agents can be useful for representing the user’s characteristics inside the system and MDs can work like “cooperative devices”. Agents can be executed on the MD and/or migrate through the net for searching information on different servers (or MDs) in order to satisfy the user’s requests. This is the underlying idea of the Mobile Agent concept. For Lin et al. [8], a Mobile Agent (MA) is an active autonomous object that can migrate on a heterogeneous network and communicate with a MA manager, different nodes and other MAs.

Rahwan et al. [12] consider an agent as an entity capable of flexible autonomous behaviour where flexible means: Responsive (or reactive) for perceiving the environment and responding in a timely fashion; Proactive for exhibiting goal-directed behaviour and taking the initiative thanks to its knowledge (defined, acquired and inferred) and, Social entity for interacting with other agents or humans. Rahwan et al. [12] recommend the use of the agent technology in MD applications for three reasons. First, a mobile user is usually located in some environment which can be represented in terms of context information, such as time, place...
and tasks. Second, the environment is dynamic, since users may move from one place to another and since some tasks may change. Finally, a mobile computer system must have the ability to be proactive, i.e. to make some reasoning about the user’s goals and how they can be achieved.

The MD applications (and their agents) must allow users to consult data at any time from any place. This is the underlying idea of the Ubiquitous Computing. For the W3C [16], Ubiquitous Computing is an emerging paradigm of personal computing, characterized by first, small, handheld and wireless computing devices; second, the pervasiveness and the wireless nature of devices require network architectures to support automatic and ad hoc configuration and, third, high distribution, heterogeneity, mobility and autonomy of the ubiquitous computing environment.

An agent could also behave independently from the server and other agents. This is the foundation of Peer to Peer Systems (P2P Systems) which are characterized by a direct communication between the peers with no communication needed with a specific server, and by the autonomy that a peer has for accomplishing some assigned tasks. Shizuka et al. [14] consider that P2P computing is one of the potential communicative architectures and technologies for supporting ubiquitous/pervasive computing and that the peers are ubiquitous anywhere. A P2P system can be Pure P2P or Hybrid P2P. A Pure P2P system is a system in which every peer is able to directly connect to any other peer and messages are sent without the mediation of a server. A Hybrid P2P is one in which a peer needs to connect to both an index server and other peers. In this connection, some group administration messages are passed via the server and other messages are directly exchanged between peers.

The goal of our work is to provide relevant information for the nomadic user who accesses a WIS through MDs. Our approach is based on the agent technology. We call an Agent-Based Web Information System (ABWIS) a WIS developed using an agent approach and accessed by users through MDs. Following the P2P approach, an ABWIS has to represent knowledge required by each agent for accomplishing tasks associated with the different roles they can play (client, server, coordinator…). The work of Panti et al. in [10] is an example of Peer MAS whose process is used in our approach in order to define the agents’ activities and the plans they execute in order to accomplish their assigned activities.

In this paper, we describe a framework called PUMAS (Peer Ubiquitous Multi-Agent Systems) that we have first introduced in [3] for designing, developing and deploying ABWIS. Each MD informs the system about the user’s location (using a GPS device) and its connection features (time, connection device, protocol…), stores information and integrates agents having the ability of performing tasks. These agents can, on the one hand, migrate to different servers (or other MD) in order to find the peer agent(s) that will help to answer the user’s requests or, on the other hand, use a central platform in order to communicate with other peer agents for achieving their assigned tasks. In PUMAS, the communication between agents is performed by means of messages using a hybrid peer to peer approach (we follow the Shizuka et al. [14] recommendations who consider it is better to use an Hybrid P2P architecture in order to prevent security issues related to the agent mobility), communication between agents in a point to point or in a broadcast way, management of the agent’s states (connected, disconnected, killed, etc.) and of the services provided by them). On the one hand, users equipped with MDs – peers - can communicate through the PUMAS central platform and on the other hand, the agents (peers) can exchange information in order to achieve their tasks (in an individual or cooperative way) and provide their services to the users or other agents. By definition [12], an agent has the characteristics of a peer since it can play different roles and knows the agents with which can communicate for performing its tasks. Users equipped with MD are peers if mechanisms allow them to know others peers for working together and/or for asking for information, and to communicate with others in order to exchange information. PUMAS provides an infrastructure that offers such mechanisms. Users can also use PUMAS for sending information requests and for receiving the results from different WIIs executed on several servers and/or MD. An ABWIS could be used for different systems like a guided tourist visit, a supply chain, the global traffic control, etc. [7]

Our approach focuses on the definition of a framework based on peer ubiquitous agents for facilitating the access to different information sources using MDs and makes use of ubiquitous agents which are in charge of adapting the information to user needs (constraints, preferences, profile, location, last requests) and the technical capabilities of her/his MD.

The paper is structured as follows: in section 2, we describe the goal, the architecture of PUMAS, our framework and we give some guidelines for implementing the communications between PUMAS agents. We conclude in section 3.

2. PUMAS FRAMEWORK

For defining the architecture of PUMAS, we have studied architectures based on agents like KODAMA [15], MIA [1] and CONSORTS [7], and identified three common levels which form the classical architecture of an ABWIS: First, a Mobile Agent Level which is composed of the MD and the Mobile Agents. The user accesses the system through her/his MD where Mobile Agents are executed. Second, an Intermediate Level which offers services (connection, communication, etc.) in order to communicate with the Information System. Third, an Information System Level represents the services that the system offers to users.

Some characteristics of PUMAS are based on CONSORTS [7]. PUMAS also relies on the three classical levels of an ABWIS: First, the Connection MAS (related to Mobile Agent Level) provides the mechanism for facilitating the connection from different types of MD to the system. Second, the Communication MAS (related to Intermediate Level) ensures a transparent communication between the MDs and the system and, displays the information to user in an adapting way according to the constraints of her/his MD. Third, the Information MAS (related to Information System Level) which receives the user’s request, redirect them to the “best” IS (the nearest, the IS which stores information which answers the user’s request, the more consulted…), takes into account the user’s profile in the system (preferences, history, intentions…) and returns the adapted results to the Communication MAS.

The inherent mobility of the user and of the agents is supported by ubiquitous agents which can be transmitted through the network to retrieve some needed information and which can communicate with other agents for performing tasks. In PUMAS, the ubiquitous agents are organized in a Hybrid P2P Architecture which
overcomes the following issues: security in the applications (security problems inherent to the agent mobility), communication between agents in a point to point or in a broadcast way, management of the agent’s status (connected, disconnected, killed, etc.) and of the services provided by them. PUMAS is based on agent and Hybrid P2P approaches and provides a direct communication between users and agents, agent autonomy for performing their tasks and for playing different roles, agent reaction for responding to environment changes and the agent capabilities for working alone or as a team member.

We briefly explain how PUMAS can be used through the example of a medical information system in a hospital. It shows how to access WIS through MDs and why it is necessary to adapt the information according to the user’s profile, to the technical constraints of her/his MD and to considerations about the connection time and user’s location. When doctors with MDs (e.g. PDAs) visit their patients, they can consult information about the patients’ clinic history, clinical tests, prescribed medicines, etc. With the location of the MD (room, floor, bed, etc. which infers the patient’s location) and the current date, the doctor can identify the patient and know patient’s medical file (i.e. according to the request, the application on her/his MD must consult the different IS of the hospital – pharmacy, patients, doctors, clinical laboratory, …) and/or she/he can communicate with other doctors (peers) through her/his MD in order to get some specific information (e.g. specialist doctors who has examined this patient).

2.1 Architecture of PUMAS

The Figure 1 shows the architecture of PUMAS. It relies on three MAS components: Connection, Communication and Information. Inside the three MAS (Connection, Communication and Information), each agent is a peer ubiquitous agent. Agents are connected to a central platform (Hybrid P2P system) and benefit this way from a mechanism for knowing all the agents and their services and for managing their communications. The agents are autonomous for connecting and disconnecting, for sending messages to a specific agent or a group of agents, and for performing their assigned tasks. Below, we describe the agents’ roles and the functionalities of each component of PUMAS and how the agents adapt the information according to the user’s needs. We also present how to model the interaction (communications) among the agents using Class and Sequence Diagram of Agent UML (Agent Unified Modelling Language) [11]. In [3] we can find other AUML Diagrams of PUMAS.

2.1.1 The Connection MAS

This MAS includes several Mobile Device Agents (MDA) and one Connection Controller Agent (CCA). Each MD can execute different MDA which are transmitted from and towards the WIS which can be distributed on different servers or MD. The knowledge of a MDA is composed of general rules of behaviour and characteristics related to the type of MD (PDA, cellular telephone, etc.) and some specific roles defined according to the application (e.g., this agent is used for transmitting a file). In addition, a MDA must know the communication protocols and mechanisms (connection, protocols, network type, constraints, etc.) shared with the system. A MDA owns data that it manages and shares (e.g. the files, agent services…). In our approach, a MDA is seen as an agent which owns the characteristics of both a Cooperative and a Connection Agent. Both agents are specializations of a Mobile Agent (see Figure 2).

![Figure 1. The PUMAS Architecture.](image1)

![Figure 2. AUML Class Diagram for a Mobile Device Agent.](image2)

The Connection Controller Agent (CCA) detects the MD type (PDA, cellular phones…) using CC/PP [16] files and facilitates its connection. A CC/PP file is a resource of the MDA and it includes some user’s characteristics like her/his user’s location (physical and logical location –city, country, street, IP Address, orientation, motion speed, connection status, bandwidth, latency…), application requirements (hw, sw, browser and WAP requirements), current session characteristics (current user, current device, current application, …) and the user’s profile (current specific user’s requirements, preferences for the current session…) using the extensions introduced by the work of Indulska et al [6]. The CCA serves as an intermediary between the Connection MAS and the Communication MAS. The CCA also checks the connections established by the users and the agents’ states (connected, disconnected, killed, etc.), and links each MDA to its corresponding Proxy Agent (PA) in the Communication MAS (see next section). The CCA introduced by PUMAS allows checking whether the user is still connected. If not, it checks if she/he has willingly disconnected or if the disconnection has been caused by a fault (e.g. system or network problem). The CCA checks out if the user wishes to continue with her/his last session (she/he would be represented by the same PA) or open a new one (she/he would be represented by a new PA).
2.1.2 The Communication MAS

This MAS offers an interface which renders the communication between users transparent and activates the mechanism for displaying the information in an adapted way (according to the MD characteristics). It is composed by several Proxy Agents, one MDPProfile Agent and one Coordinator Agent.

There is one Proxy Agent (PA) for the connection of each MDA (two different users can connect themselves to the system through the same MD and there would be two different PDAs). The main task of a PA is to represent a MDA within the system. In this case, there are two agents, one MDA on the MD and one PA in the system (located on the server or on another MD). The PA has the same properties and behaviour as the MDA except the ones concerning the connection (see Figure 2). Because of MDA has the characteristics of Cooperative Agent and in order to take advantage of the agent mobility, a MDA can also play itself the role of PA. In this case, there is only one agent in the system, the MDA which can be transmitted to the system (or to another MD) and can be executed on the server (or to another MD) playing the role of a PA. One contribution of PUMAS is the representation in both the PA and the Mobile Device Agent (MDA) of the knowledge and behaviour which are inherited from the Cooperative Agent (see Figure 2). First, PA and MDA are able to perform the Coordination, Cooperation, Control and Negotiation (CCCN) tasks in the MAS. Second, they adopt the P2P process explained in [10] for defining the strategy to be applied for achieving the assigned tasks (CCCN or another) and for finding out the peers which can cooperate and work.

The MDPProfile Agent has to check the user’s profile (according to her/his MD) and her/his information needs. In addition, this agent together with the Coordinator Agent checks and establishes the mechanism for sending hypermedia data to the user.

The Coordinator Agent (CA) is in permanent communication with the Connection Controller Agent (CCA) in order to verify the connection state of the agent which searches for information. The CA knows all about the agents connected in the system (using a yellow pages mechanism): their connections, states, services, location and all about user profiles: technical features of their MDs, user’s location and connection time. If there are problems with the CCA (if CCA fails, if there is a lot of connections...), the Coordinator Agent can play the role of CCA up until the problems are solved. At that moment, the CCA and the CA must synchronise the information about the connected agent and check the current connections.

2.1.3 The Information MAS

The Information MAS is composed of one Receptor/Provider Agent, one Router Agent and one or several IS Agent(s).

The Receptor/Provider Agent (R/PA) owns a general view of the whole system. It knows the agents of both the Communication and Information MAS, their services, their locations, their profiles, etc. The R/PA receives all the requests that are transmitted from the Communication MAS and redirects them to the Router Agent (RA) which is in charge of finding the “right” Information Systems (IS) in order to perform the query. This R/PA checks if the query results take into account the user’s profile (preferences, user history, intentions, etc).

To redirect the query to “best” Information System(s), the Router Agent (RA) adopts a strategy which depends on several criteria (location, peers similarity, time constraints, etc.). The strategy can lead to the sending of the query to a specific WIS, to the sending of the query in a broadcast way and/or to the decomposition of the query in sub-queries, each being sent to one or several WIS. The Router Agent is also in charge of compiling the results it obtains from the WIS and of analysing them to decide whether the whole set of results or only a part of it is to be sent to the R/PA.

An ISAgent associated with a WIS receives the user’s query from the Router Agent and is in charge of searching for information. Once a result for the query is obtained, the IS Agent returns it to the Router Agent. An ISAgent can execute the query by itself or delegate this task to the adequate WIS component. This depends notably on the nature of the WIS. Our approach addresses complex and possibly distributed WIS located on server(s) but also very simple WIS which only rely on some files located on a MD. In this last case, one IS Agent may be sufficient to ensure the right functioning of the Information MAS. It is worth noting that, in this case, what we call an “IS Agent” is in fact the MDA of a MD which can play the role of an IS Agent since it has the knowledge required for executing a query on the files stored in the MD. In a complex WIS, the IS Agent can collaborate with other IS Agents (if the WIS has been developed following the MAS paradigm) or with any other WIS component to perform the query. In the case of a non MAS based WIS, our approach only requires that an IS Agent is developed in order to ensure the communication between PUMAS and the WIS.

![Figure 3. The Sequence Diagram for an information request.](image)

The Figure 3 show how an information request is propagated toward the R/PA from the MDA. This figure is an AAML Sequence Diagram which represents the interactions (messages) among the agents. These messages are called Communication Acts (e.g., accept, inform, propose, request, subscribe, propagate ...). There are messages that can be sent in a concurrent way and there are ones executed according to a condition. For instance, the Figure 3 shows how a Mobile Device Agent (MDA) could send a request to the Connection Controller Agent (CCA). This CCA could send a message in order to create a Proxy Agent (PA) (if a PA does not exists in the system for representing the MDA) or, it could send a message for informing the PA that its corresponding MDA needs some information. We can also see how after the Receptor and Provider Agent receives the results of the information request, it can send to MDPProfile Agent a message of “refuse” (e.g., to refuse the request results because of a timeout), or a message of “not understand” (e.g., not to accept these results because of it can not understand them) or a message of “propose” (e.g., redirect the message which is a proposition of answer for the information request).
2.2 Example
Let us now come back to the example of the medical system (see Figure 4). When a doctor arrives to the patient’s room, she/he can ask for information about a patient (e.g., the prescribed medicines for this patient). The MDA executed on her/his MD sends the request to the Connection Controller Agent (CCA) which adds up to the request the room and bed number (MD’s location which indicates the patient’s location). CCA also gets the current date from the system (patient’s location for knowing the patient identity). The request is propagated through PUMAS core from the CCA: it is first transmitted through the Communication MAS agents (Proxy Agent, Coordinator Agent and MDProfile Agent). The MDProfile Agent can add up into the request, the information according to the MD, for example, that this kind of MD can not support video or a particular format of file, only text files (if the doctor asks for the test results, she/he could only get them in a text format). Then the MDProfile Agent sends the request to the Receptor/Provider Agent (R/PA). This agent can add up into the request the doctor’s preferences. For instance, if the doctor asks for prescribed medicines, the system must also provide her/him with the patient’s diet and the results of the patient’s blood tests. Finally, the request arrives to the Router Agent (RA). This agent redirects the request to the IS Agent located in the Information System(s) which manage(s) the information about the patients in the hospital. All the requests follow the same path from the MDA to the RA towards the RA which analyses the requests and redirects them to the most suitable IS(s). For example, if the doctor wants to know the last prescribed medicines to this patient, the RA redirects the request to the IS Agent located in the IS of the hospital’s pharmacy. If the request is for another doctor (peer), the RA redirects the request to the IS Agent located in the peer’s MD. In our example (see Figure 4), RA sends the same request to a doctor’s peer (Doctor 2) and decomposes the request in three sub-requests: the blood tests request (toward the Clinical laboratory IS), the diet request (toward the nutritionist) and the prescribed medicines request (toward Pharmacy IS). A doctor can also ask for information about a specific patient to several of her/his peers. In this case the Router Agent could send the request in a broadcast way or it could decompose the request according to the receiver peer (e.g., requests relates to the heart for the cardiologist…). The retrieved information is organised by the Router Agent (e.g., the list of prescribed medicines to the patient after her/his last entry at the hospital, her/his diet and the results of her/his blood tests, the peer’s answers about this patient …) and it is returned to the doctor who sent the request following the inverse path. Different agents check the results because the doctor could have changed her/his location, or she/he could have had different problems like network problems, or she/he could have disconnected from the system and re-opened her/his session in a new connection, or each connection (or reconnection) could have different network conditions (features like bandwidth, communication protocol, internet service provider, etc), or she/he could consult the information through other MD, etc.

Through this example, we can observe the Hybrid P2P Architecture of PUMAS. The core of PUMAS centralizes the requests: on the one hand, it is in charge of obtaining the more relevant information (which satisfies the user information needs) and, on the other hand, it is in charge of adapting the answers. The main peer characteristics of PUMAS agents are: first, the agents have the autonomy of connecting to and disconnecting from the system; second, a MD can communicate with a specific IS (located on a server or on a MD) passing this information like a parameter of the request and the Router Agent transmits the request to this specific IS (agent to agent communication, for example, communication between two doctors in order to ask for information about a patient). An advantage that PUMAS offers is that it can also help a user who does not know which specific IS to ask for needed information thanks to the Router Agent. In our example, if a doctor needs the patient information but she/he does not know which IS manages it, the Router Agent redirects the request to the “right” IS by means of an intelligent analysis of the request and the help of the ISAgents which achieve an intelligent search inside the different IS (pharmacy, laboratory, patients…).

![Figure 4. Sending an Information request.](image)

2.3 Implementation of PUMAS Applications
For implementing the applications based on PUMAS, we have chosen JADE-LEAP [5], a FIPA compliant platform. We have tested it using two different kinds of MDs of our team (Pocket PCs with Windows CE, using Créme – kVM which is Personal Java compliant- and PDAs with PalmOS using an implementation MIDP 1.0). JADE-LEAP provides an execution environment for the agents, a class package for developing the agents and the graphics tools in order to control and manage the execution of the agents. After these tests, we get some ideas for implementing the PUMAS agents and we found some correspondences between PUMAS’ components and JADE-LEAP ones: we can associate the JADE-LEAP’s AMS (Agent Management System) with our Coordinator Agent, DF (Directory Facilitator) with our MDProfile Agent, Receptor/Provider Agent and Router Agent and, ACC (Agent Communication Channel) with our Connection Controller Agent. JADE-LEAP also proposes a Dummy Agent for sending and receiving the ACL (Agent Communication Languages) messages and, a Sniffer Agent for controlling the communications between the agents. For the knowledge representation, JADE-LEAP uses JESS and a set of classes (JADE Content package) which define ontologies. JADE-LEAP also lets the user define her/his own ontologies and we want to integrate SWRL1 for performing this task. In our tests of the JADE-LEAP examples, we found some problems like a mismatch between the active agents listed in the platform and the agents connected according to the application (some disconnected agents appeared

---

1 SWRL: Semantic Web Rule Language, a W3C Recommendation which combines OWL and RuleML in order to define ontologies and the rules for managing a MAIS. [http://www.w3.org/Submission/2004/SUBM-SWRL-20040521/](http://www.w3.org/Submission/2004/SUBM-SWRL-20040521/)
like connected for the platform). For this reason, we propose the Connection Controller Agent (CCA) (which controls all the connected agents) and the Coordinator Agent (which has a log of the agents of the system and could play the role of CCA if it fails). We could not execute JADE-LEAP in the Stand-Alone way in our Pocket PC. We went back to Split Execution which simulates a Hybrid P2P Architecture with the following advantages: knowledge of active agents, theirs services and capabilities; less traffic in the network and a mechanism of registration (subscription) and authentication which lets to an agent recognizes all its peers.

3. CONCLUSION AND FUTURE WORK
In this paper, we have proposed PUMAS, a framework for modelling, designing and developing an Agent Based Web Information Systems (ABWIS). PUMAS allows the ABWIS to provide nomadic user with the more relevant information adapted according to her/his characteristics, preferences, history, etc. and to the technical constraints of her/his MD. We have described PUMAS components, its agents, theirs roles, the information management (flow, storage, representation, etc. of information) considering aspects as user location changes, communications between users, definition of user’s profiles, etc. Our future work concerns the proposition of a language dedicated to the agents communication (rendering temporal and spatial information availability) and the development of the corresponding API. We must also define a strategy description language for the Router Agent in order to send requests toward the “best” IS and to compile the answers, a description language for the PUMAS Agent in order to define their roles, profiles and rules and, a mechanism for adapting the query results.

4. ACKNOWLEDGMENTS
The author Angela Carrillo-Ramos is partially supported by Universidad de los Andes, (Bogotá, Colombia).

5. REFERENCES


