AGENT BASED ARCHITECTURE IN AN AMBIENT INTELLIGENCE CONTEXT

Nikolaos I. Spanoudakis [a, b] Pavlos Moraitis [b]

[a] *Singular Software SA, Al. Panagouli & Siniosoglou, 14234, Nea Ionia, Greece, nspan@singularlogic.eu*
[b] *Dept. of Mathematics and Computer Science, René Descartes University, 45 rue des Saints-Pères, 75270 Paris Cedex 06, France, pavlos@math-info.univ-paris5.fr*

Abstract

This paper provides an insight on the special requirements of accessibility content and services in an ambient intelligence context and proposes an agent-based approach into a more general service oriented architecture for addressing them. It is based on previous approaches for agent-based information systems regarding infomobility services adding the mobility impaired people special requirements. The result is an architecture with multiple agents on the user's nomad device that address the ambient intelligence issue and a family of dedicated personal assistance agents to each type of mobility impairment whose deliberation provides the best solutions for people that have a combination of impairments. In this paper we pay a particular attention on the task of integrating the multi-agent system in the overall architecture.

1 Introduction

Agent technology has been applied to the infomobility services sector in recent years. Such services include location-based services like mapping and points of interest search, travel planning and, recently, trip progression monitoring and pushing information and events to the user. Works like the ones carried out in CRUMPET [15] and the most recent Im@gine IT [12] projects have addressed the state of the art and even furthered it. Recently, agent technology was also applied for supporting virtual elderly assistance communities, in the context of the TeleCARE project [2].

A recent research has proposed that elderly and disabled people compose a segment of the population that would profit very much from ambient intelligence (AmI), if the latter is accessible [1]. Furthermore, O'Hare et al. [13] advocate the use of agents as a key enabler in the delivery of ambient intelligence. Thus in an AmI framework for servicing elderly and disabled (mobility impaired people) the role of agent technology is crucial.

In this paper we present a part of the work proposed in the Integrated Project (IP) "Ambient Intelligence System of Agents for Knowledge-based and Integrated Services for Mobility Impaired users" (ASK-IT, IST-2003-511298), which aims to offer infomobility services to mobility impaired people and support them while on the move. The work in this project is based on the one proposed in Im@gine IT [12]. However, this project goes further by taking into account the needs of different types (mobility impairments) of users (or possible combinations of these types), and by involving reasoning which uses different (and sometimes conflicting) knowledge describing these types' needs. Moreover, using ambient intelligence, it provides to the impaired people special and context based support, while on the move. In order to address all these challenges we propose an agent-based architecture, including the participating agent types and the agent platform's integration with the OSGi ([6], [14]) middleware. Thus, in what follows, we will provide a general presentation of the multi-agent system (MAS) that we conceived for addressing the ASK-IT challenges and that is the core of the overall architecture. We will particularly focus on the way we integrated this system in the overall ASK-IT service oriented architecture, a process that proved to be an important challenge.

The rest of the paper is organized as follows. In section 2 we present the application requirements and challenges while in section 3 we discuss the background relative to this work. Section 4 presents the ASK-IT architecture and section 5 focuses on the integration of the MAS with the overall ASK-IT architecture. Finally, we conclude in section 6 with a brief discussion.

2 Application Requirements and Challenges

ASK-IT is based on the Im@gine IT project extending it to offer a new generation of services addressing the mobility impaired persons' needs. The ASK-IT detailed requirements and use cases were presented by Simões et al. in [17]. Previous projects (e.g. CRUMPET, Im@gine IT and TeleCARE) scarcely address the case of an elderly or handicapped person requesting infomobility services. This is a new situation, with the following characteristics:

- Personalization doesn't only refer to learning user habits. It encapsulates the need for knowledge regarding the situation that this person is into. Therefore the personal assistant agent must employ powerful knowledge regarding the type of impairment of the person. It is highly different to request for and present a route to a person on wheelchair. The accessibility features of crossroads and of busses or trains must be taken into account. In the case of an elderly or sick person even the weather conditions must be taken into account. Thus, agents with knowledge sufficient to serve each impairment type must be developed. Furthermore (and for the case that a person has more than one types of impairment) these agents must be able to cooperate in order to serve that person. Work within this sector will be wholly innovative and will address the coordination of a team of agents that decide over one subject. Moreover we will have to tackle emerging behaviour, since the combinations of types of impairments and their extent can be limitless.

- The immediate ambient plays a vital role for servicing the user (domotic services, ticketing services, etc.) and all these services must be accessible by the user agent on his/her device. Here the user agent doesn't only use a local service in order to present information to the user. The agent either needs to select and get the local service and then adapt it to the user's needs, or is the actual user of the service (e.g. switch on the air-condition) and must have the relevant knowledge and profile of the user. Moreover, the agent has access to information provided on real-time by sensors on the user's body and must act whenever an abnormal situation is recorded.

3 Background

The background on relevant agent architectures is the FIPA [7] standard and the results of the Im@gine IT project [12] that addressed open issues defined by FIPA.

FIPA specified an architecture for Personal Travel Assistance (PTA) applications. The following agent types were proposed to participate in this architecture:

- The *Travel Service Agent (TSA)* is responsible for accessing services and replying in the requester's ontology. He is proposed to specialize in a domain, like global flight plans and hotel arrangements, for example. This agent type is responsible for maintaining the data access, interpretation and delivery to other service agents that might be implemented as a "wrapper" around legacy databases or web services.

- The *Travel Broker Agent (TBA)* is responsible for locating and contracting TSAs. He can obtain the travel options from several services, filter and select from the alternatives, and legally bind a contract and travel documents based on a final selection. He can schedule and incrementally reschedule the entire travel plan across several service types such as flight, train, hotel and special events.

- The *Personal Travel Assistant (PTA)* acts on behalf of a user and is legally authorized to do so, up to the level allowed by the user. While conceptually seen as one *Personal Assistant* (PA) for each user, the implementation could be assumed to use a multi-user, server-based design. This agent is responsible for remembering and following the user's instructions and learning the user's preferences based on choices or feedback after the trip. A *Mini-Personal Travel Assistant (MPTA)* is a lightweight agent that is typically device-dependent, such as an agent operating on a PDA or laptop computer, where, for instance, bandwidth and modality become special issues. Although this tends to cause a restriction on functionality, many additional functions such as GPS and GSM can be provided in this context.

FIPA also specifies a set of standard interaction protocols such as *FIPA-request, FIPA-query,* etc. that can be used as standard templates to build agent conversations. Moreover the possible heterogeneity of agents points to the use of as much as possible standardized practices. The JADE-Leap ([3], [8]) version allows for light-weight implementations of agents on nomad devices like PDAs, laptops and smart phones. Caire et al. [5] presented a communication protocol for JADE-Leap agents taking into account the inherent problems of an agent platform executing on a dynamic and limited environment like that of a mobile phone. This platform provides the elements needed for the ASK-IT application.

As is obvious, a lot of implementation specific issues are left for the application developer to decide. Also, FIPA admits that this architecture has some deficiencies. The Im@gine IT project [12] extended the FIPA architecture and addressed the open issues within the FIPA report. It introduced the following new agent types:

- *Transport Mode Agent:* This agent is a first attempt to integrate ambient intelligence with MPTA. He monitors the user's active route and tracks his progress notifying the MPTA whenever a route segment has been completed. He also can send information to the user, e.g. about the next bus-stop. It is also used for accessing nearby services supplying the results to the personal assistant for further processing (e.g. can understand when the user enters his car).

- *Interface Agent*: This agent controls the access of the end user to the local main MAS platform. The interface role is a part of a business to customer (B2C) operator site in which the user authentication and profile data reside.

- *Specialised service provider agent (similar to FIPA TSA)* that provides a service to the network advertising it to the geographically closest middle agent. The advertising of the service contains for the first time information that specifies the conditions under which the service will be offered (geographical area that the service can be achieved, price, availability, etc). A provider role can be:

  – *Simple*. The simple service provider role advertises and accepts simple services such as mapping or geocoding.
  – *Complex*. The complex service provider role is capable of synthesizing a complex service requested by the user (e.g. plan a trip), interacting with one or more simple service providers.
  – *Events Handler (subscription service)*. This role accepts events and forwards them to the interested personal assistant agent. The events can be traffic events along a specific route that a certain user has activated and submitted to the events handler.

The Im@gine IT architecture provided solutions for the future personal travel assistant (PTA) developments as they were identified by FIPA. It addressed the challenges: a) of agent mobility in a network (the personal assistant agent delegates the task of filtering huge amounts of data to the complex provider agent by sending parts of the user profile along with the user request), b) travel monitoring (through the concept of the events handler agent) and c) inter-operation between agents and workflow (by allowing the interface agent to manage value flows between the user and the brokers, the latter managing the value flow towards the provider agents). Finally, it provided a complete solution for the nomad devices service provisioning including not only simple services but also the delegation of complex tasks and subscription services. The solution is composed of a protocol, a service profiling scheme and the relevant matchmaking process [18].

4 The Proposed Architecture

Taking all the above into consideration, the proposed ASK-IT architecture will be an evolution of the Im@gine IT architecture in two ways:

a) by integrating ambient intelligence to the *Mini-Personal Travel Assistant (MPTA)*, and,

b) by proposing a server side dynamic coalition formation aiming to serving users with more than one types of impairments.

Service discovery and provisioning will use the middle agent paradigm (see e.g. [11]). This approach assumes that agents that provide services are able to advertise them to middle agents. Moreover, the latter have profiles of available web services. Personal assistants or any service requester agents can then ask the middle agents for service providers that are suitable for the needed task.

The family of agents that will be able to address these challenges will include the following. The definition of these particular types of agents has been based on the requirements document provided by the users:

- The *Personal Wearable Intelligent Device Agent (PEDA*, acting as a Mini-Personal Travel Assistant for persons with impairments) that provides the personalized infomobility services to the user

- *Ambient Intelligence Service Agents (AESA)* that configure the environment of the user according to his habits/needs (new type of agent)

- The *Personal Wearable Communication Device Agent (PWDA)* that monitors the user's sensors and provides information either directly to the user or the Personal Wearable Intelligent Device agent in cases of emergency (new type of agent)

- *Provider Agents* that advertise and offer services to the ASK-IT service network (acting as Travel Service Agents)

- *Middle Agents* that have white and yellow page information about providers agents and cooperate in order to provide all available services through any contact point (acting as Travel Broker Agents)

- *Elderly and Disabled Assistant Agents (EDA)* that specialize in the mobility requirements and needs of any type of handicap (new type of agent)

- *EDA Coalition Creator (EDAC)* that is responsible for accepting requests aimed for the EDA and dynamically forming the coalition of different Elderly and Disabled Assistant Agents experts that will have to deliberate on the user's goal (new type of agent)

Figure 1 provides an insight on the proposed architecture for the nomad device (client side), while Figure 2 presents the ASK-IT server architecture. Arrows define interfaces and technologies that are foreseen to be used in order to realise them. The ASK-IT server has a broker agent and the Elderly and Disabled Assistant agents. Provider agents can be deployed either on the ASK-IT server or in other computers accessible through the Web. The personal space (or personal area network-PAN) includes the devices on the user's person. On the center there is the nomad device that has the connection to the internet and where the Personal Wearable Intelligent Device, Ambient Intelligence Service and Personal Wearable Communication Device agents will be deployed. The body area network (BAN) includes devices that sense the user (sensors). Finally, the immediate ambient (or local area network-LAN) includes services that are available only when a user is in a specific area (e.g. in a building, in his car, etc).

As far as system implementation is concerned, most – if not all – of the related projects use open-source, FIPA compliant development environments, like JADE ([4], [8]). The latter provides standard agent technologies and offers to the developer a number of features in order to simplify the development process:

- Distributed agent platform. The agent platform can be distributed on several hosts, each one of them executes one Java Virtual Machine.

- FIPA-Compliant agent platform, which includes the Agent Management System the Directory Facilitator and the Agent Communication Channel.

- Efficient transport of ACL messages between agents.

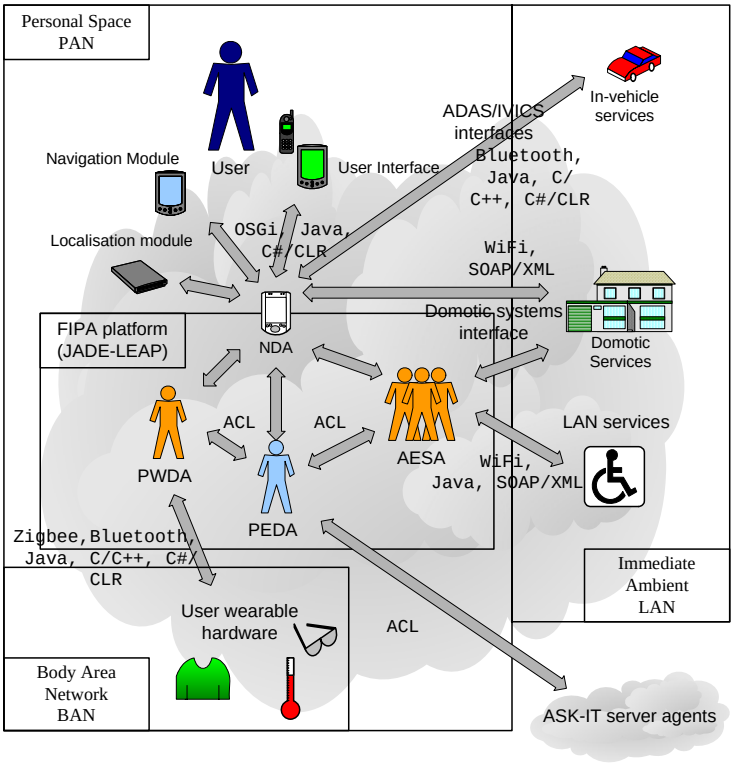

Figure 1: ASK-IT nomad device architecture

In both the server and client side the OSGi technology was selected ([14], we discuss more on this issue on the next section) for connecting the various components and services. Such components on the client and server side include:

- *Nomad Device Application (NDA):* It is the component that controls the device application and provides connectivity and visibility between the different installed components and services. It can start or end available services and components.

- *Localization Module*. It produces accurate coordinates of the user.

- *Navigation Module (NM)*. It navigates the user to its destination.

- *User Interface (UI)*. The ASK-IT Human-Machine Interface.

- *Domotic Services*. These services allow the user to control and monitor devices in his household (heater, air-condition, etc).

- *In-vehicle services*. Interfaces and tools to provide access to In-Vehicle Information and Communication System (IVICS) but also to provide user related information and needs to Advanced Driver Assistance System (ADAS).

- *ASK-IT database (DB).* Retains information about the ASK-IT users.

- *ASK-IT web server (WS).* The user can create an account, monitor services usage and pay for them using an internet browser. Tools for system administrators are also web-based.

- *Data Management Module (DMM).* This is a module utilised by the broker agent. It allows service providers to add their web service in the ASK-IT web services repository and provides a graphical interface where they can match their service Web Service Definition Language (WSDL) files to the ASK-IT ontology.

- *Web services* offered by a big number of external service providers.

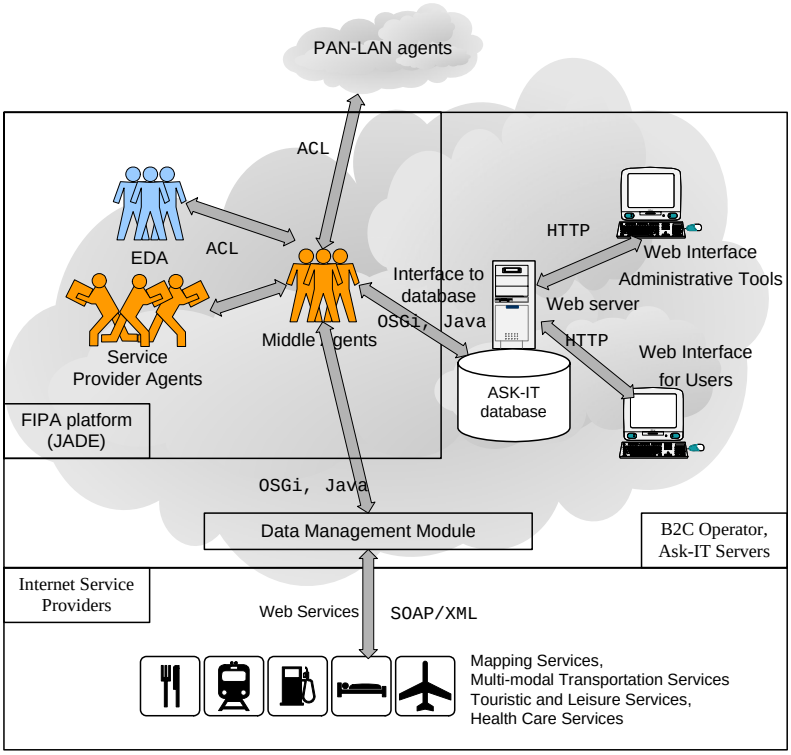

Figure 2: ASK-IT server architecture

At this time, it is important to make some comments on the Elderly and Disabled Assistant agents' functionality and their cooperation with the Personal Wearable Intelligent Device agent. The Personal Wearable Intelligent Device agent acts a Mini-Personal Travel Assistant (MPTA) and has two types of knowledge when he prepares a request for servicing a user:

- His profile data that can be:

  – dynamic (learned behaviour), or
  – static (user defined),

- His current context (on the move, in a building, etc) that is determined by interactions with the Ambient Intelligence Service and Personal Wearable Communication Device agents.

What the Personal Wearable Intelligent Device agent doesn't have is the knowledge regarding the needs of different mobility impaired types with regard to movement. This knowledge can be vast and/or conflicting if it tries to encompass all types of impairments. Therefore, we decided that it will be defined separately for each type of impairment. Then, in the case of a person that has more than one type of impairments, a coalition of agents – each with the knowledge of a different type of impairment – will deliberate on the needs of the user. These Elderly and Disabled Assistant agents will use argumentation [9] in order to resolve eventual conflicts and find a compromise. They will interact through an argumentation-based dialogue protocol like the one proposed in [10]. However, this aspect of our work is out of the scope of this paper and it will be presented in the near future. Thus, whenever the user wants to plan a trip the Personal Wearable Intelligent Device agent will add the context and profile relevant data to the request. Then the Elderly and Disabled Assistant agents on the server side, after a deliberation dialogue, will add the relevant data based to the user's impairment type(s) and make the relevant requests to the broker (who will now find the real service providers to offer their proposals). When the Elderly and Disabled Assistant agents receive the service invocation results from the broker they will sort the proposed routes according to the user's context and type of impairment and send them to the mobile device (again through the broker).

The other novelty of our architecture will emerge through the cooperation of the Personal Wearable Communication Device, Ambient Intelligence Service and Personal Wearable Intelligent

Device agents on the user's personal space (ambient intelligence and personal travel assistance integration). There, the Personal Wearable Intelligent Device agent will inform the Ambient Intelligence Service agents about the user's context (e.g. returning home) and the Ambient Intelligence Service agents will act on the user's environment according to his habits (e.g. turn on the heater so that the user finds it hot when he arrives at home), and vice-versa, the Ambient Intelligence Service agents will be able to inform the Personal Wearable Intelligent Device agent that the user is now indoors (e.g. he arrived at home) thus providing information regarding the user's context.

Moreover, the Personal Wearable Communication Device agent will be able to provide to the Personal Wearable Intelligent Device agent information about the health status of the user or about other user context information like outside temperature. Through this interaction the Personal Wearable Intelligent Device agent may learn user habits also based on his personal and environment context. Finally, the Personal Wearable Communication Device agent can also act when the user is incapacitated or in grave danger (based on sensor information) by proactively calling for help.

5 The Agent Platform – OSGi Integration

In ASK-IT not all cooperating software components are agents, therefore one main challenge of our work was related to the way to connect all the identified modules and agents. The Open Services Gateway initiative (OSGi) technology provides a service-oriented, component-based environment for developers and offers standardized ways to manage the software lifecycle. It was chosen for integrating all the participating components because of its offering the following possibilities:

a) A bundle (that is how the OSGi components are named) can import but also export java packages when installed. In Java, there is normally a single classpath that contains all the classes and resources. The OSGi framework caters for controlled linking between different software modules, as they are installed on runtime. This allows, for example, the exportation of the ontology by one bundle and its use by all others that are installed afterwards.

b) In the same virtual machine, bundles can be started or stopped dynamically on runtime. This feature allows for the best utilization of resources of nomad devices (low computing power).

c) The bundles can locate and invoke services offered by other bundles. This allows for dynamic interoperability between our different components. Services definition and advertisement is easy and intuitive (as the paragraphs below will demonstrate).

In the next paragraphs we firstly present the methodology for integrating the FIPA agent platform in the OSGi framework and then the integrated architecture for the ASK-IT server and client. In the figures the light grey packages depict the ASK-IT defined components, while the dark grey ones depict the open source libraries used (i.e. the knopflerfish OSGi framework, the JADE-Leap framework and the Jena semantic web framework).

5.1 The Integration Methodology

The methodology that we used for defining the bundles that participate in the architecture is the following:

a) Define the ontology that will be common for all bundles and used for defining the services signatures (in the case that all the signatures will use simple Java classes like `Integer` and `String` there is no need for an ontology)

b) Define the java interfaces for the services that will be offered by each bundle (all interfaces definitions may import the ontology bundle).

c) Define the different bundles, each implementing the relevant interfaces (all bundles dynamically import the ontology and service descriptions bundles)

5.2 ASK-IT Client Architecture

The client architecture is presented in Figure 3 as a UML deployment diagram [19]. It is a simplified version of the client containing only the UI module (ASK-IT HMI bundle) and the multi-agent system (MAS) part that runs on the client side (ASK-IT client MAS bundle). The `org.ask_it` package contains three important sub-packages:

a) `org.ask_it.ontology`: It contains the java bean classes of our ontology. They have been created using the Protégé tool [16] and the JADE beangenerator add-on. It is dependent on the JADE framework, which is why it is not a separate bundle.

b) `org.ask_it.client_interfaces`: It contains the interfaces definitions for all ASK-IT OSGi modules (different class for each module). In this case the HMI bundle and the ASK-IT client MAS bundle.

c) `org.ask_it.abstraction`: It contains important classes for achieving interoperability between OSGi and JADE-Leap agents and also the various constants that we are going to use.

The first two packages are common for all ASK-IT client bundles. The third package is common for the agents built within the ASK-IT MAS client bundle and contains the classes needed for communication between the agents and the OSGi implementation class (`MASServiceImpl`). Using the `org.ask_it` packages each developer can create its bundles and services.

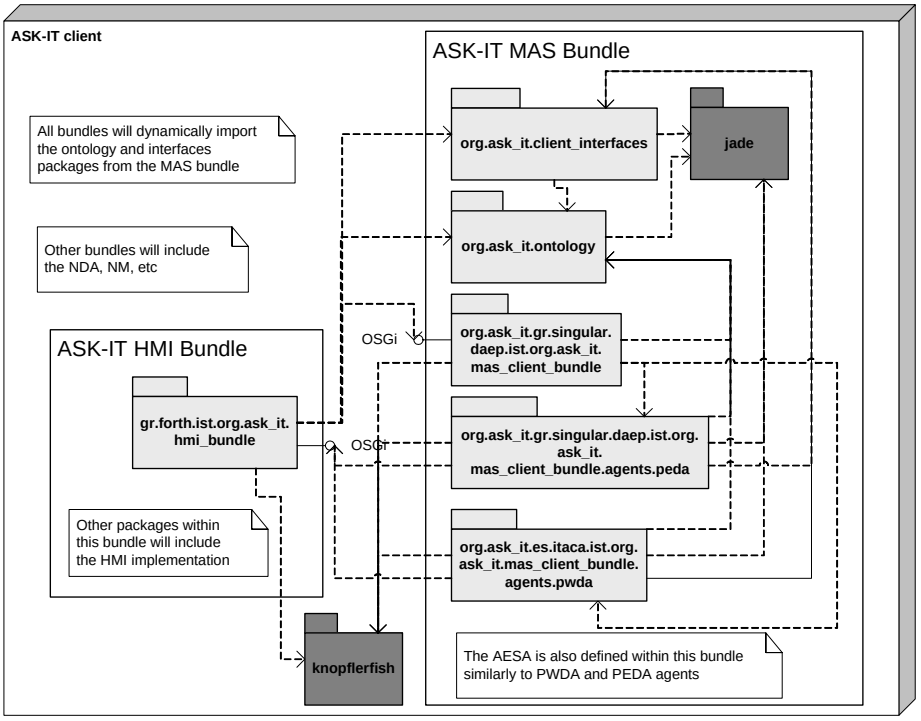

Figure 3: ASK-IT Client Architecture

Figure 4 shows the contents of the `org.ask_it.abstraction`, `org.ask_it.client_interfaces` and the OSGi bundle definition and Personal Wearable Intelligent Device agent definition packages. The ontology package is not presented because it contains a large number of classes.

In Figure 5 we present the interface for MAS client. Some of the services defined are the request to plan a trip (`planATripRequest`) and to create a map (`createMap`). Note that the interface imports the `org.ask_it.ontology` package and uses concepts from there in order to define the method signatures (e.g. `Coordinates` and `Address` concepts).

Figure 4: The ASK-IT client MAS bundle (including the Personal Wearable Intelligent Device agent) classes

```
package org.ask_it.client_interfaces;

import org.ask_it.ontology.*;

public interface MASService {
    public Route[] planATripRequest(Coordinates origin, Coordinates destination);
    public GeocodedAddress[] geocodeAddress(Address address, int maximumReturnedAddresses);
    public void callForHelp(Coordinates origin, String[] userCategories,String language, String[] userMedicalCondition);
    public Map createMap(ScreenSize screenSize, BoundingBox boundingBox, POIForMap[] showPOIs, Line[] routeLines, String routeMapID);
    public POIForMap[] proximitySearch(BoundingBox boundingBox, String[] POITypes, String language);
    public POI[] getPOIInfo(String[] POIName, String language);
    public String getMaid(String language, String start, String finish,
            Address address, int desiredAge) throws NullPointerException ,RuntimeException;
}
```

Figure 5: The `MASService` interface definition

The java files outlined in Figure 6 realise the interface between the JADE-Leap platform and its agents with the OSGi platform. The ASK-IT MAS client bundle services are defined in package `gr.singular.daep.ist.org.ask_it.mas_client_bundle`. The `Activator` class starts and stops the bundle services by declaring that the `MASServiceImpl` class will define the methods declared in the `MASService` interface and expose it as an OSGi service (or bundle of services). The `MASServiceImpl` implementation class's attributes include the reference to the agent platform, that it starts and controls (`ContainerController cc`), and to the agents that it creates (Personal Wearable Intelligent Device, Personal Wearable Communication Device and Ambient Intelligence Service agents). Then, it implements all the methods defined in the `MASService` interface and one more, for shutting down the platform (`shutdown`).

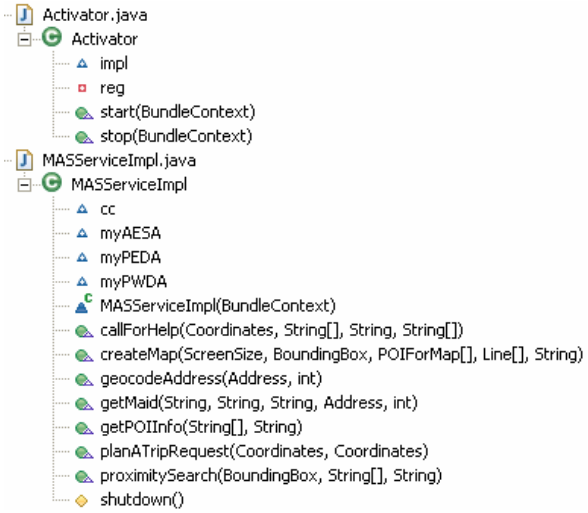

Figure 6: The classes realising the OSGi interface to the agents

The OSGi bundle `Activator` class is the one executed by the OSGi framework. It starts and stops the bundle according to the OSGi administrator wishes. Any OSGi bundle can execute a service offered by the MAS client bundle simply by invoking a service described in the `MASService` interface. The `MASServiceImpl` is responsible for selecting the agent that will service each service. It contacts the agent using the `putO2AAgent` method of the JADE framework by means of which Java objects can send

other objects to the agent as events. The agent gets the event, initialises the relevant protocols and at the end of his work he notifies the event sender of its results. The agent accesses services offered by other bundles using the OSGi `BundleContext` object that is passed to it as a parameter when created by the `MASServiceImpl` class.

5.3 ASK-IT Server Architecture

The ASK-IT server (see the relevant UML deployment diagram in Figure 7) follows the same architecture as the client. The ASK-IT server defines two types of servers for interacting with other ASK-IT servers or the ASK-IT MAS clients:

a) an HTTP server (using the JADE HTTP Message Transport Protocol [8]). If the client is an independent platform then it uses this server in order to transmit new ACL messages (PC client). Other ASK-IT servers also use the HTTP server in order to connect.

b) a JICP server (using the JADE Internal Communication Protocol, [5]). If the client is a mobile phone or a nomad device like PDA (split-container platform) that operates without having a static IP address (its IP changes over time) then it uses this server in order to transmit new ACL messages.

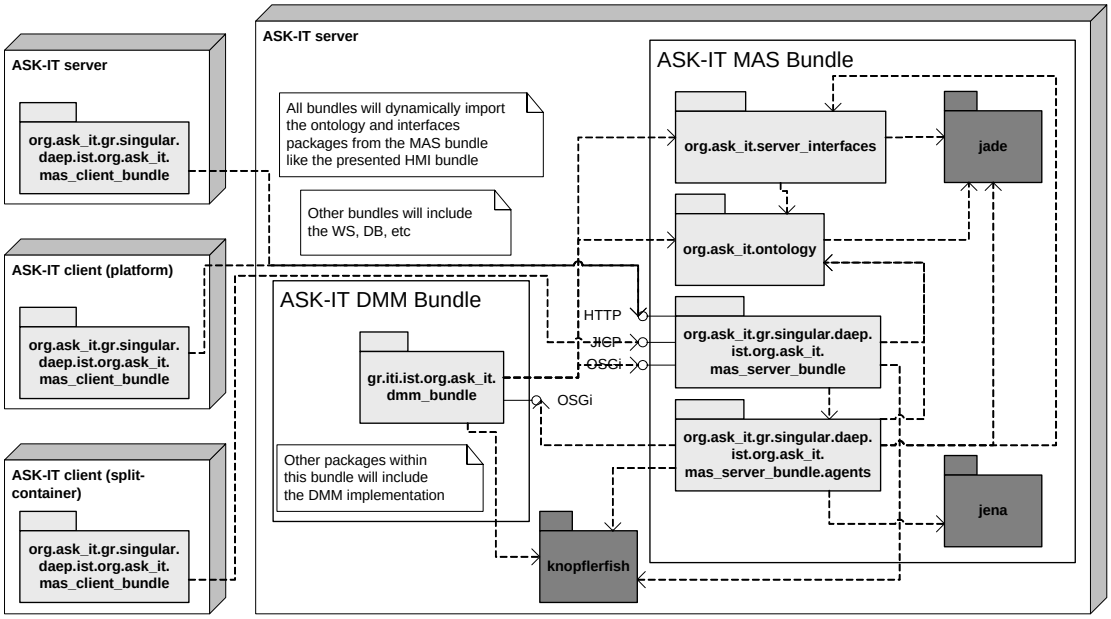

Figure 7: ASK-IT Server Architecture

6 Discussion

In this paper we presented an agent-based system, integrated in a more general architecture, for the personal travel assistance domain that addresses a) the need for integration with ambient intelligence and b) the need to service mobility impaired users, allowing for the usage of an extensive knowledge for each type of impairment and also the service of combinations of impairments.

An important novelty of our work, in which a particular attention is payed in this paper, is the integration of the Open Services Gateway initiative with a typical FIPA compliant agent platform, the JADE-Leap framework. We provide guidelines sufficient for any agent application developer to integrate his agents with other software modules using the modern service-oriented OSGi technology.

By concluding, we can say that the proposed work extends the state of the art with regard to the FIPA PTA standard as well as the work done in previous projects on this domain. Specifically, it proposes that the personal assistant agent on the user's nomad device cooperates with ambient intelligence related agents that can monitor and act on the user's environment, or read information available through different sensors on his person. Moreover, it proposes that the user's context and profile knowledge is taken into account when forming service requests and that his impairments-related

knowledge is also taken into account before finally searching for the relevant service to his needs. The latter step takes place on the server side due to the limited capabilities of processing on the nomad devices. Finally, the service results are filtered and sorted according to the user's needs, also on the server side, so that he gets only the needed information on his device, thus overcoming possible bandwidth problems but also saving time.

Herein, we have, therefore, addressed some important issues for modern agent technology based applications, that could be summarized as follows: a) how to integrate personal travel assistance applications with ambient intelligence, b) how to allow for the use of knowledge regarding the needs of different user groups when a user may belong to more than one group, and, c) how to integrate an agent platform in a service oriented architecture.

One important aspect of our future work in this project is the proposal of an argumentation based deliberation dialogue among the different involved Elderly and Disabled Assistant agents when a user has more than one types of impairment (e.g. a person on wheel chair also having heart problems).

References

[1] Abascal, J.. Ambient Intelligence for People with Disabilities and Elderly People. ACM's Special Interest Group on Computer-Human Interaction (SIGCHI), *Ambient Intelligence for Scientific Discovery (AISD) Workshop*, Vienna, April 25, 2004.

[2] Afsarmanesh, H., Guevara Masís, V., Hertzberger, L.O., Virtual Community Support In Telecare. *Forth IFIP Working Conference on Virtual Enterprises, PRO-VE'03*, Lugano, Switzerland, October 2003.

[3] Bauer, B. and Bonnefoy, D. and Bergenti, F. and Evans, R., The Lightweight Extensible Agent Platform (Software Demonstration). *Fifth International Conference on Autonomous Agents*, Montreal, 2001.

[4] Bellifemine, F., Bergenti, F., Caire, G., Poggi, A.. JADE - A Java Agent Development Framework. Multi-Agent Programming: Languages, Platforms and Applications, Bordini et al. (eds.), ISBN: 0-387-24568-5, pp 125-147, 2005.

[5] Caire, G., Lhuillier, N. and Rimassa G., A communication protocol for agents on handheld devices. *Proceedings of the first International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS2002)*, 2002.

[6] Cervantes, H. and Hall, R.S., Autonomous Adaptation to Dynamic Availability Using a Service-Oriented Component Model. *Proceedings of the 26th International Conference on Software Engineering (ICSE 2004)*, Scotland, May 2004.

[7] FIPA: Personal Travel Assistance Specification. Foundation for Intelligent Physical Agents, XC00080B, http://www.fipa.org, 2001.

[8] JADE-Leap. Java Agent Development Environment – Lightweight Extensible Agent platform, http://jade.tilab.com

[9] Kakas A., Moraitis P., "Argumentation Based Decision Making for Autonomous Agents". *Proceedings secondnd International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS'03)*, pp. 883-890, Melbourne, Australia, 2003.

[10] Kakas, A., Maudet, N., Moraitis, P. "Modular Representation of Agent Interaction Rules through Argumentation", *Journal of Autonomous Agents and Multi-Agent Systems, (JAAMAS)*, Springer, vol. 11, no. 2, pp. 189-206, 2005.

[11] Klusch, M. and Sycara, K., Brokering and Matchmaking for Coordination of Agent Societies: A Survey. Coordination of Internet Agents, Omicini et al. (eds.), Springer, 2001.

[12] Moraitis, P., Petraki, E. and Spanoudakis, N., An Agent-Based System for Infomobility Services. *The third European Workshop on Multi-Agent Systems (EUMAS2005)*, Brussels, Belgium, December 7 - 8, 2005.

[13] O'Hare, G. M. P., O'Grady, M. J., Keegan, S., O'Kane, D., Tynan, R. and Marsh, D.. Intelligent Agile Agents: Active Enablers for Ambient Intelligence. ACM's Special Interest Group on Computer-Human Interaction (SIGCHI), *Ambient Intelligence for Scientific Discovery (AISD) Workshop*, Vienna, April 25, 2004.

[14] OSGi, Open Service Gateway initiative, http://www.osgi.org

[15] Poslad, S., Laamanen, H., Malaka, R., Nick, A., Buckle, P. and Zipf, A., CRUMPET: Creation of User-friendly Mobile Services Personalised for Tourism. *Proceedings of the second International Conference on 3G Mobile Communication Technologies*, London, UK, 2001.
[16] Protégé, An Ontology Editor and Knowledge Acquisition System. http://protege.stanford.edu
[17] Simões, A., Gomes, A., Bekiaris, E. et al., ASK-IT Use Cases. Deliverable D1.1.2, IST-511298 project ASK-IT, http://www.ask-it.org, 2006.
[18] Spanoudakis N. and Moraitis, P., Engineering a Brokering Framework for Providing Semantic Services to Agents on Lightweight Devices. *Proceedings of the ECAI'06 Workshop on Context and Ontologies: Theory, Practice and Applications (C&O'06)*, Riva del Garda, Italy, 2006.
[19] UML, Unified Modeling Language, http://www.uml.org