An Agent-Based Approach for a Smart Transport System

C. Peñaranda[a], J. Aguero[a], C. Carrascosa[a], M. Rebollo[a], and V. Julian[a]

[a]Computer Sciences and Systems Department. Polytechnique University of Valencia, Camino de Vera s/n, 46022, Valencia, Spain

KEYWORD

Multi-agent systems; Smart cities; Artificial intelligence

ABSTRACT

This paper presents a proposal for a Smart Transport System which is an application that facilitates the interconnection between people (citizens, tourists) and transport providers (Bus, metro, trains, trams), defining the services that everyone can request or offer. The system has been defined as a virtual organization where agents (representing actors of the transport system) can enter or leave into the system consuming or offering services. Due to the fact that modern urban public transport is increasingly an important service used by citizens in current cities, the proposed system will improve the use of resources while also ensuring time flexible mobility solutions for citizens.

1. Introduction

Modern urban public transport is becoming more and more important service used by citizens in cosmopolitan cities. The aims of the governments (both local and central), and Public Transport Agency is to take steps to positively discourage cars coming into city centres and encourage the use of public transport. So, all the public transport providers must to provide a better, more convenient and higher quality of service in the hope of attracting more passengers.

The increase of human mobility and transportation in urban environments presents one ofthe challenges facing our today's society. It is one of the causes of congestion problems,inefficiencies in logistics and energy use, and air pollution in modern cities. To approach thischallenge, innovative solutions for communication networks, information processing, andtransport are required in order to assure a more efficient use of resources (vehicles, energyresources, roads, etc.) while also ensuring time flexible mobility solutions for citizens andbusinesses (Billhardt and et al., 2016).

The current technology for public transport can be used for improving the quality of service and therefore its attractiveness to passengers. This technology provides public transport operators with real-time information on the operational status of the public transport system, which allows them to more effectively manage their fleet (Lujak M., 2015), and to take remedial actions if disruptions occur. In addition, these systems can provide real time information to passengers (Adam et al., 2012), both on the vehicle and at bus stops and form the basis of an *Smart Transport system* (Chamoso and de la Prieta, 2015).

This paper proposes an approach for a *Smart Transport System* which main goal is the development of an integrated solution for the management,coordination and regulation of urban transport in a city. The proposed solution will be based on aset of informatics tools and components (sensor networks, models, mechanisms, techniquesand algorithms). The system is mainly based on agent technology, which, although still immature in some ways, allows the development of systems that support the requirements of applications of this kind (Isabel and Fernandez, 2015). Specificallyagent technology allows the formation and management of systems where the main componentscan be humans and software agents providing services to humans or other agentsin

an environment of whole integration. The proposed *Smart Transport System* allows mobile users (passengers) to know the service offered by the Public Transport Agency based on context information such as the user preferences, ie, the user can receive/request information for possible tourist routes, personalized news services, transport ETA (Estimated Time Arrival), shorter routes to the destination, etc.

The rest of the paper is structured as follows: Section 2 describes the main features of the proposed Smart Transport System; Section 3 introduces the mobile services offered into the system; Section 4 shows how the system has been designed as a virtual organization; Section 5 gives details about the implementation of the system; and, finally, Section 6 summarizes some conclusions.

2. Smart Transport System

The *Smart Transport System* is an application that facilitates the interconnection between passengers (citizens, tourists) and transport providers (Bus, Subways, Trains, Trams); delimiting services that each one can request or offer. The system controls which services must be provided by each agent. Internal functionality of these services is responsibility of the provider agents. However, the system imposes some restrictions about service profiles, service requesting orders and service results.

So, first the system is described from the physical viewpoint. The proposed system provides wireless data services, which allow mobile devices to communicate with different provider servers. The network architecture provides access to wireless services for users equipped with mobile wireless devices, via a set of access points deployed in key points around of the city (usually in Bus Stops which offer information services). This architecture is built upon a number of wireless communication standards (by example: WIFI, Bluetooth) which are employed to deliver these services to the passengers. A graphical view of this system (and network architecture) is shown in Figure 1.

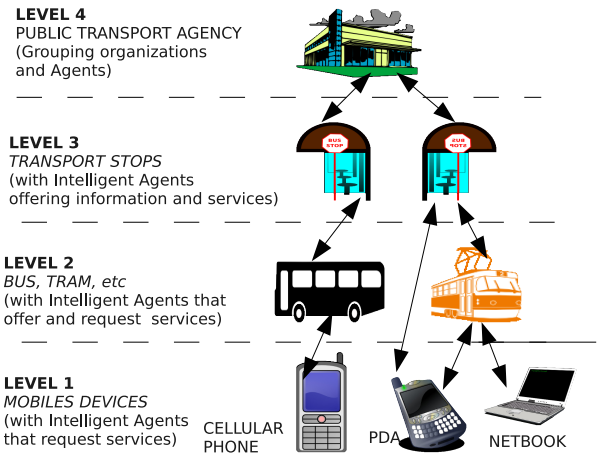

Figure 1: Smart Transport System architecture.

The **first level** is composed of mobile devices equipped with intelligent agents that act as Personal Assistants (PA) for the passengers. This PA is used to obtain information of public transportation. This information can be requested directly by the passenger, for instance, the user can ask to the bus stop the best route to reach a specific destination. Also, the system can make recommendations adapted to the user profiles, depending of their preferences, for instance, if the user is inside the bus, the system can automatically adjust the bus environment,

the daily news showed on the screens are adjusted according to the preferences of the passengers. In this layer only can coexist agent passengers (mobile devices), as shown in Figure 1. These agents may only requests services to the upper layer (to the stops, buses, trains, etc.), but they can not provide services to the upper layer. However, the system allows that the passengers provide services among them.

The **second level** is mainly formed by the bus agents, which act as an access point or gateway for service providers. The Bus agent controls all peripherals of the vehicle such as: ticket machines, passenger displays, destination displays, etc., thus allowing the driver to fully concentrate on the traffic. This Bus agent is located inside the vehicle and only can be accessed by passengers that are inside the bus. In addition, this agent stores data of the route that the bus is serving, including the distance between each stop, the names of the stops and the scheduled running times. The Bus agent shows in the displays the name of each bus stop, the daily news, ETA and the weather (only outdoor temperature). Moreover, this agent has the ability to provide services to passengers and request services of the stops or the central agency.

The **third level** is formed by the Bus Stops (public transport stops) which show to passengers the destination of the next bus together with its ETA. This agent can display relevant information to the passenger as routes, daily news (which are request to the Public Transport Agency). Passengers with their PA can download relevant information from the Bus Stop agent. Also, the display shows the name of each bus stop, as the bus approaches it and indicates -in real-time- any deviations from the schedule. Furthermore, this agent allows the positive location of vehicles taking into account any location error.

The **fourth level** is composed of the Public Transport Agency which acts as a "container" of different Public Transport organizations (Passengers, Buses, Trams, Stops, etc.). But its main goal is to allow the creation and registration of agents, transport units and their services, and, in addition, to control and synchronize the information of the system. So, the primary function of the control center staff is to ensure that buses run on time and that services are efficient and reliable. They must ensure that disruptions of any service are quickly identified and that corrective actions are taken as soon as possible.

2.1 Smart Transport through AmI

Environment is a notion that is fundamental to Ambient Intelligence (Augusto, 2007), as well as associated disciplines such as ubiquitous computing and pervasive environments. In each case, the Smart Transport ,the physical environment is composed of a vast and rich infrastructure of hidden electronic devices and components (and purpose) of which may differ significantly. Thus an unfortunate but realistic case could arise in which certain geographic environments are endowed with certain technologies, but the applications that use these are not compatible with it. In essence, this is a question of interoperability, but how to ensure that all AmI applications are interoperable is a difficult, if not impossible task.

A first way to solve interoperability problems is adopting a rigorous and formal standardization. However, while this approach may seem promising, there are some problems to take into account, for instance, this process has its own time-consuming and therefore may not respond quickly to technological changes and commercial developments. A second and more flexible approach consists of obtaining interoperability in a similar way as the traditional Web Services on Internet. This approach uses different technologies, however adopting open technologies (and standards, for instance, standard communication protocols) reduces the complexity of any interoperability issues that may arise in the future.

Therefore, AmI environments (like the proposed in this work) will support a number of services that would ultimately enhance the quality of the user's experience (in this case passengers). Finally, there exist three fundamental characteristics of the AmI environments (Brønsted et al., 2007). These characteristics are the promotion mechanisms of the services, the access mechanisms and the intelligent mechanisms used in the environment. Figure 2, shows how the AmI mechanism are integrated and interact in the proposed environment.

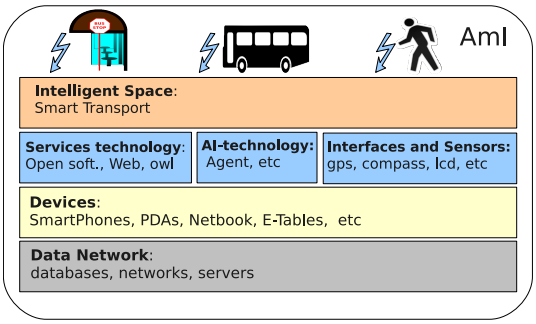

Figure 2: Smart Transport as an AmI.

- The promotion mechanisms, in this case the environment perceives itself as a mediator or facilitator of transport services. The AmI environment would mediate between the key actors, namely the passengers, stops and transport vehicles. The stops are the fixed components of the AmI environment and the transport vehicles and the passengers form the mobile components. The AmI environment has a dynamic model of the passengers and transport vehicles. Passengers can directly subscribe and use the offered services. However, the AmI environment also has some characteristics of proactivity, since not only the passenger's has the responsibility of starting all the transactions. The stops and the transport vehicles can start some transactions in order to invoke some services.

- The access mechanisms, the AmI environment provides an open interface to the transport framework as well as facilities to its use. Also, it is included a sensorization interface in order to improve the environment monitorization. Therefore, the passenger would have the option of using this interface to communicate with either stops or transport vehicles that have services that the passenger wants to use. AmI may provide intelligent and intuitive interfaces in order to cover all the necessities, in this sense:

  - Standard smartphones, the software interface would be on the passenger's device. The main difficulty here is how to ensure that the software will run on a greatest range of devices, given the large number of combinations that exist.
  - Standard interfaces, use of fixed terminals or LCD displays inside the transport vehicles or in stops (in outdoor environments), which support multimedia information. These interfaces does not really complement the ubiquitous nature of the system, but they will be managed by intelligent agents.
  - Standard embedded sensors, use of GPS sensors that allow enabling intuitive and transparent interactions. The implementation of suitable interactions is , of course, a mandatory goal of the systematic application of technology known as HCI (Human-Computer Interaction).

- The intelligent mechanisms, the AmI environment involves the use of intelligent software. However, the classic AI-technology, can be computationally expensive requiring reasonably sophisticated hardware. Though advances in mobile technology have been significant in recent years, they are still not adequate for running AI-based applications. Nevertheless, intelligent agents have been demonstrated that can be executed on mobile devices. These agents are known as embedded agents. Thus, in the proposed AmI scenario, intelligence can be distributed on the fixed components of the architecture (on the stops of bus, metro, train, etc) and, also, on the mobile components (transport vehicles and passengers). The use of intelligent agents will provide characteristics of autonomy, reactivity, proactivity and mobility which can be beneficial into an AmI scenario (Agüero et al., 2012).

2.2 Smart Transport Scenarios

The transport system is organized in such a way that, when a mobile user enters within the range of a public transport providers (bus, metro and stops), the intelligent agent installed in the embedded device allows the provider mutually discover each other. This interaction within the framework of the transport system (physical viewpoint) is illustrated in Figure 3.

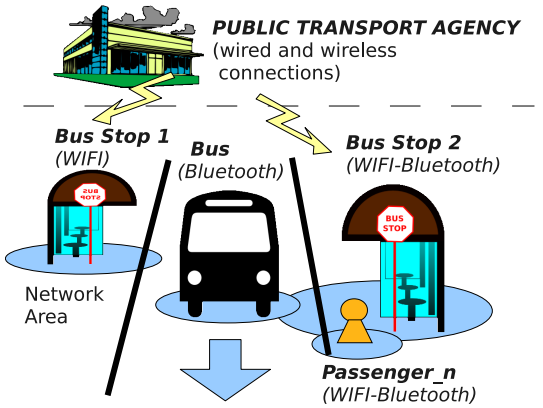

Figure 3: Structure of the Smart Transport System.

This interaction raises at least three possible scenarios, depending on the role played by the participant agents in the system. That is, there are cases where a particular agent plays the role of service provider, but at another time the same agent has the role of client service.

The first scenario assumes that there is a passenger who has to use a public transport and therefore moves to a specific bus stop. In this case, the passenger will consume services (client role) that offers the own stop (provider role) in order to improve the user experience. This client-server relationship between agents is typically controlled by a sequence of events and facts, which can be summarized in the following explanation:

- A passenger equipped with a typical smartphone is making their way towards a transportation area. The smartphone is loaded with the agent-based transport system while the providers (bus, metro, tram, etc.) are equipped with the necessary infrastructure to support such passenger and form individual nodes on the smart transport system network. As can be seen from Figure 3, the passenger is currently approaching a bus stop.

- On entering in the area network, a number of processes start. First of all, the stop (an embedded agent) detects that a passenger is inside its area. As there is a service list in the stop to be offered, the stop agent requests the user profiles to the PA of the passenger. This is followed by a registration process and a profile analysis.

- The Personal Assistants sends the user's information. This information includes a description of the mobile device currently being used and the user preferences. The stop agent requests assistance to the Agency Transport network node, with the user profile.

- When the Agency Transport gives its approval and the user is successfully registered. The user profile is analyzed by the Agency Transport for current user preferences and device capabilities. Then the Agency Transport compiles a list of applicable services and products from its Service/Products Catalog.

- After this, the Agency Transport sends the results to the stop agent for its knowledge.
- The Personal Assistants shows the information regarding these services and products to the user who makes a choice and selects (makes a request for) the service/products he wishes to use. The PA forwards the user service request to the stop, which instantiates the service.

The second scenario assumes that the bus acts as a client and the stop is the service provider. The services used in this scenario allow the synchronization of the ETA of the system, because the stops do a crawl of the buses (or another transport system: trams, trains) to know their position, and so, to calculate the ETA. Finally, the third scenario specifies that the user is into the transport unit (bus, trains, etc.) and he/she is using the services provided by the transport unit, including notification and information services. Similar to the first scenario, there are a number of facts and events that trigger different operating activities for the clients. The description is very similar to the first scenario, so it is omitted.

3. Mobile Service

The system can provide various types of services to facilitate and improve the quality of the public transport system. The services are used to solve the difficulties found (due to the complexity of the problem) and try to achieve the goals of the Transport Agency, which are improving the currently operating transport routes (and opening new routes) reflecting the actual position of journeys, with emphasis on improving and upgrading the service in order to raise the satisfaction rating of public transport users. In addition, the Transport Agency wants to provide the user of Real-Time Passenger Information, which provides passengers in the public transport stations (metro and bus stops) of instant information about the state of the transport system. According to this, the types of services offered by the system can be grouped into three categories (depending on the type of service, the role of client and provider may change): (i) Localization Service; (ii) Instant Information Service (Real-Time Information); (iii) Adaptive Service.

- *Locator Service*, is a service offered by the system for the position detection and vehicle tracking. The system is designed to track buses and trams running scheduled services along known routes within urban areas. At the start of each execution cycle the position of the bus is confirmed automatically by a location beacon. Once the vehicle has started, its position is tracked along the route by measuring the traveled distance. In this case, the bus stop (and roadside beacon) acts as a service provider, because they can monitor the position of the buses in real-time, compare their actual positions with the schedule and take appropriate actions in the case of problems. In contrast, the Bus agent and PA act as clients of the system, since they receive the changes of itineraries/routes that are provided by the stops
- *Real-Time Information service*, it shows information continuously to the user (to the waiting passengers in the stops or inside the public transport). This information can be presented in two ways, visual and audible, and must be updated in real time. Typically, the bus stop displays show information of: bus destination, ETA or scheduled departure time, connections, messages of possible disruptions and some special information (breaking news). Furthermore, the users can download this information in their mobile devices and then they can read it later.
- *Adaptive Service*, it is another service that adapt the "environment" according to the group preferences, using an analysis of the profiles of the passengers. For instance, the temperature inside the bus could be controlled adjusting the air conditioner, to an average value of the desired value by the group. Furthermore, it is possible to adjust the news items shown in the LCD (LCD located in the metro stops or bus stop) according to the passengers preferences, through a simple voting process.

Once described the general types of services into the Smart Transport System, we can now focus on specific services that are provided to passengers by the transport units (buses, trains, trams, etc) and public transportation stops.

3.1 Vehicle Transport Services

First we describe the services offered by the transport unit when the passenger is inside the bus, tram, etc. These services are: (i) Display Information; (ii) Payments; (iii) Stop Notification; (iv) POI'S; (v) Climate Adaptation.

- *Display Information* It provides information (mainly news) on screens adapted to the profiles of the users, that is the preferences and tastes of the users known by the PA. This service can be illustrated in Figure 4, where the transport agent receives the information of the user preferences, and through a voting process, as the proposed in (Davidsson et al., 2005), the most relevant news are selected to be shown on the screens.

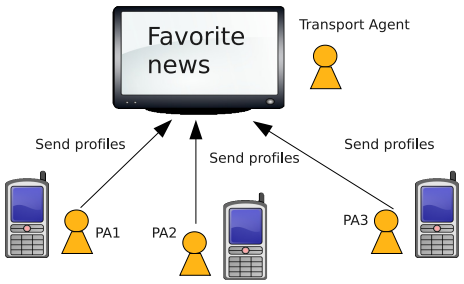

Figure 4: Display Information Service.

In addition, the service also allows to download the most relevant news in the mobile phone of the user. This allow the user to read later the news with more calm and not on the LCD of the transport vehicle. Finally, figure 5 shows the simplified protocol used for this service.

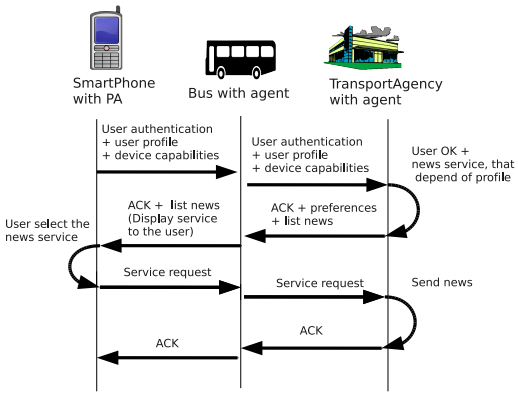

Figure 5: Display-info Service protocol.

- *Payments* it allows to automatically pay tickets when users are boarding the bus or tram. This service allows the user agent to pay the necessary amount when the agent is requested by the transport vehicle.

This service means that the PA agent has the necessary tickets (bus pass or similar) to pay the amount of the bus service. The bus pass must be perviously purchased by the user in the Transport Agency and it must be stored in the database of the PA. Figure 6 shows the simplified protocol used for this service.

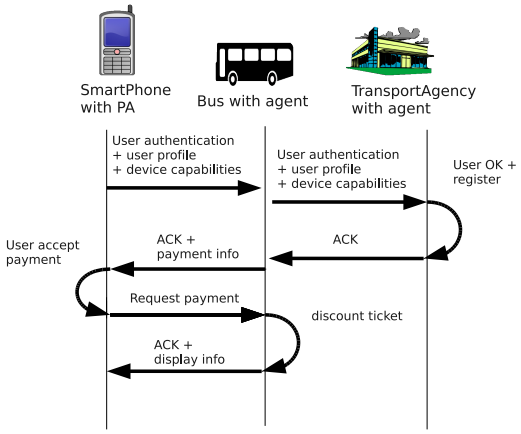

Figure 6: Payment Service protocol.

- *Stop Notification* notifies to the PA that its destination is the next stop. This notification is performed by displaying a message on user's mobile screen accompanied by a vibration or sound to alert the user.
- *POI'S* is a notification server of important places (such as touristic places or restaurants) close to stops or stations. This service is optional to be used by the PA, and allows to inform the user about important places to visit in the city. These POI'S are easily accessible when using public transportation.
- *Climate Adaptation* this service allows to adjust the climatization system (air conditioning or heating) depending on the preferences of users that are in the transport unit. This service reads user profiles to meet the preferred temperature of each user. This is similar to the behavior of the *Display Information* service. The transport vehicle agent asks the desired temperature to the PA (at each stop) and adjusts the thermostat by calculating a simple average.

3.2 Stop Services

This section describes the services provided by the stops. The amount of services offered may vary depending on the physical environment where it is located. The services provided by the stop agent to the PA are the following: (i) Display Information; (ii) Route Search; (iii) Transport Notification and ETA; and POI'S.

- *Display Information* adjusts the information on the station screens according to user profiles. This is similar to the service offered into the transport vehicles. Besides, this service allows to download the preferred news (according to the user profile) to a mobile phone.
- *Transport Notification and ETA* reports about transport units that are arriving to the stop and, also, shows the estimated time of arrival.
- *Route Search* calculates a transport route from the user position to their destination, indicating the possible transhipments that the user may need. This is done by planning a possible route that the user should follow

to reach its destination. This service used a similar technics to another similar works which has been explained in detail in (Mees, 2000). These proposals are similar to our approach to provide Tracking and Planning/Scheduling services, and may be adopted in our approach without problems, only making small changes.

- *POI'S* is a service that informs to the user of important/interesting places (such as touristic places or restaurants) close to stops or stations. This service shows the direction to follow to reach the point of interest on the mobile device.

4. Smart Transport as a MAS

This section describes the proposed system as a virtual organization where designed agents (representing actors of the transport system) can enter or leave into the open system consuming or offering services. In recent years, some works on agent-based systems have focused on providing procedures and methods of designing open MAS, where agents may have self-interested behavior or be selfish. The open MAS should also permit the participation of heterogeneous agents with different architectures and even different languages(Dignum, 2003). Thus, in order to support open MAS, developers have focused on the organizational aspects of the society of agents, to lead the system development process using the concepts of organization, rules, roles, etc. This has led to a new approach called *organization-oriented* methodologies (*Virtual Organization*)(Zambonelli et al., 2003).

Virtual Organization is considered to be a social entity that consists of a specific number of members that carry out different tasks or functions. The main aspects of an organization are the *Structure*, *Functionality*, *Dynamic*, *Normative* and its *Environment*. Therefore, to model theses characteristics, five key concepts are used: *Organizational Unit*, *Service*, *Environment*, *Norm*, and *Agent*(Argente et al., 2008).

4.1 Smart Transport as a Virtual Organization

Our proposal defines a set of models which provide the necessary concepts and components to describe a Smart Transport system as a Virtual Organization. This set of models is based on π**VOM** (*Platform-Independent Virtual Organization Model*)(Agüero et al., 2009) and is divided in different views: the Structure (topology), Functionality, Dynamism, Environment, and Normative behaviors of the organization. Thus, the Smart Transport definition corresponds to the identification of different entities in different views. These entities are agents, roles, organizational units, norms, etc. The following subsection describe in detail the different views for the proposed system.

4.1.1 Structural View

The system is modeled as an organization ()TransportAgency) where we can identify two organizational units (StopUnit and TransportUnit) that represent a group of agents. Each unit is dedicated to represent the *Transport Stops and Stations* or *Transport Provider*, respectively. Two kind of roles can interact in the Transport Agency: Clients and Providers. The *Client* role requests system services and the *Provider* role is in charge of performing services. due to space constraints, it is not possible to describe all the components.

The organization model (the Transport Agency model), which permit implementing the Smart Transport System (as an organizational structure that provides services) is show in Figure 7, with its units, roles and relationships between them. Entities are described as UML components with their corresponding stereotypes.

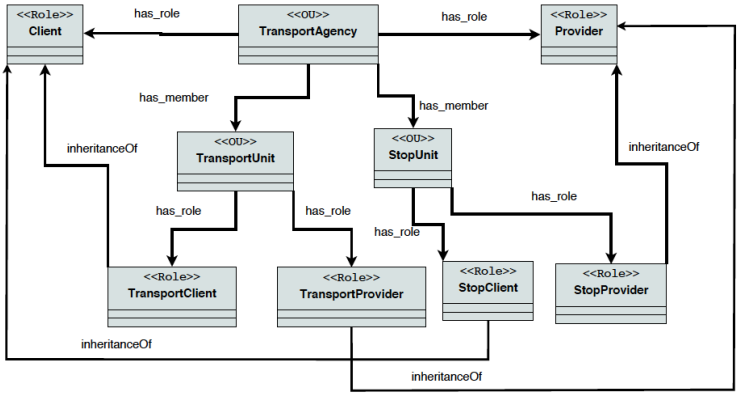

Figure 7: Structural model of TransportAgency.

4.1.2 Functional View

This view specifies the global objectives of the organization; the services and the features that it offers; the objectives pursued by the different entities of the organization; and the tasks and plans that it must follow in order to achieve them.

The Smart Transport can provide various types of services to improve the quality of the public transport system (as explained above in section 3). These services are used to achieve the goals of the Transport Agency, which are improving the currently operating transport routes and to increment the use of the public transport. The Transport Agency organization offers general services: Search, Information, Tracking, Scheduling/Planning and Payments. Those services are specialized for each unit, according to their types of providers (Transport and Transport Stop). Due, to the extent of the problem this paper will focus only on services that are provided to the passenger.

In this case Figure 8 shows only the functional view of the StopUnit. Offered services are described and the roles that request and provide these services are indicated.

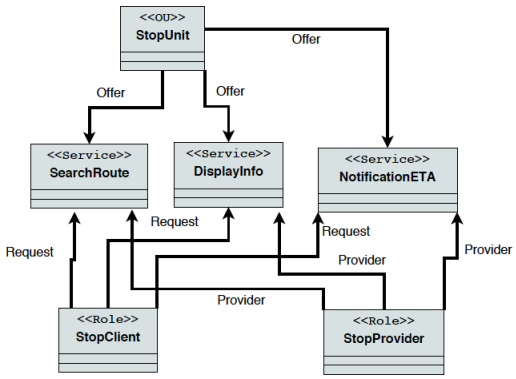

Figure 8: Structural model of StopUnit.

4.1.3 Environment View

This view includes the resources on which the organization depends, e.g., the suppliers of the resources, the customers, and other stakeholders of the organization. This view details the elements that represents a point of interaction between the entity and other elements of the model serving as an interface to the real world. Figure 9 shows the environment model used in the proposed TransportAgency.

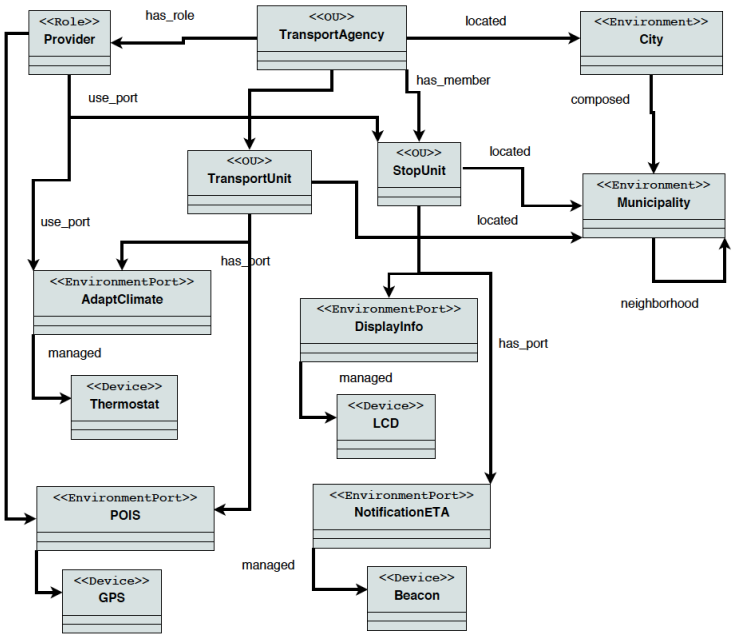

Figure 9: Environment view of TransportAgency.

4.1.4 Normative View

This view assumes that the coordination between agents is achieved through the use of *social norms*. These norms describe the expected behavior of the members, i.e., what actions are permitted, required or necessary and which have to be avoided. *Norms* are used as mechanisms to limit the autonomy of the agents in large systems and to solve complex coordination problems. This view specifies the set of rules and actions defined to control the behavior of the members of the organization, specifically the *Roles* of the organization. Figure 10 shows the environment model used in the proposed TransportAgency.

4.2 Agents

An *Agent* is the basic entity of a MAS, it is considered as a member of the organization and employs a set of interaction protocols in order to be coordinated with the rest of the members. The Agent view is a set of interrelated components, each showing a specific function for the agent definition. The main components are: *Behaviors*, *Capabilities*, and *Tasks*(Agüero et al., 2008b). The agent design starts by defining the different components used in each agent. These components are part of the π**VOM** meta-model defined in (Agüero et al.,

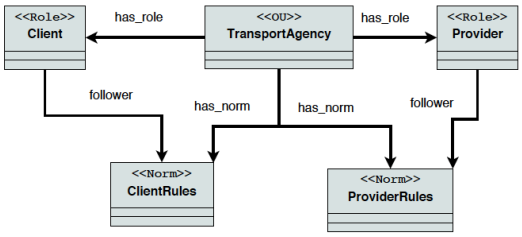

Figure 10: Normative view of TransportAgency.

2009). This section briefly describes the main agents identified in the system, that is: Personal Assistants or users, Transport Agents and Stop Agents.

4.2.1 Personal Assistant

This agent model represents a passenger in the system. This agent has different behaviors that allow managing services offered by the transport provider. For reasons of brevity this paper only explains the three most important behaviors of this agent. These three most important behaviors allows the agent to acquire the role of client, to discover the available services and to share the user profile. In Figure 11 these three behaviors can be observed. the `NetDiscover` *Behavior* allows the agent to perform the necessary activities to find a transport provider (Bus, tram, train, etc.) or a stop provider. This allow to begin the registration process of the agent and then send the user profile. By the other hand, the `StopClient` *Behavior* does the necessary tasks to employ the services offered by the StopUnit. In a similar way the `TransportClient` *Behavior* is used to manage the services offered by the TransportUnit.

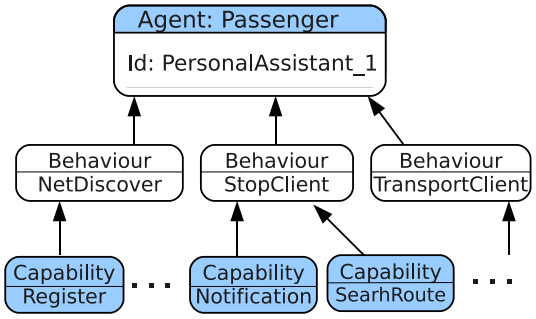

Figure 11: PA Agent Model.

These behaviors are formed by some individual capabilities (see Figure 11), which depend of the services offered or requested by the agent or by the organization. As an example the `NetDiscover` *Behavior* includes the `Register` *Capability*. This *Capability* does the actions of looking for a Transport Provider (or Stop) and enables the registration of the corresponding agent. The `StopClient` *Behavior* includes the necessary *Capabilities* to manage the different stops or stations services. These *Capabilities* can be seen in Figure 11. For instance, the `SearchRoute` *Capability* allows the passenger to request a route (to a stop agent) to a specific destination using the public transport.

4.2.2 Transport Agent

This agent must perform several roles at the same time in its life, so this is one of the most complete agents of this proposal. The Transport Agent must behave as a client agent to update the information that stops or central stations can provide, such as rerouting or notifications over the proposed route. In addition, the agent must provide services to users currently traveling within the transport unit.

One of the most important Transport Agent behaviors is the `NetDiscover` *Behavior* which allows to find customers and another service providers. Figure 12 shows this behavior, which is activated when the transport unit arrives at a station or stop. Then, the Transport Agent acquires the client role. The capabilities of this behavior depend on the services that the Transport Agent may request. The `Notification` *Capability* implements an important service which allows to receive relevant information for the transport unit and for travelers. The `TransportProvider` *Behavior* actives a the service provider role in the Transport Agent. In Figure 12 we can see at least two *Capabilities*, the *Notification Capability* and the *POIS Capability*. The *Notification Capability* is employed to notify the approaching stop to the user. The *POIS Capability* informs about the exixtence of interesting places to visit at each stop.

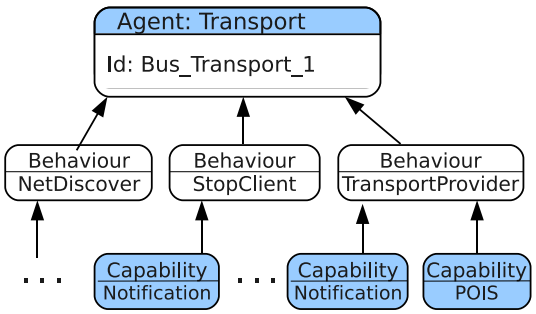

Figure 12: Transport Agent Model.

4.2.3 Stop Agent

The Stop Agent is in charge of providing services in stops or stations. These services are provided to the users (PA agent) or to the transport unit (Transport Agent) when they establish some kind of communication with the Stop Agent. We can see the agent model in Figure 13.

This agent has several behaviors, the most important ones are showed in Figure 13. The `NetDiscover` *Behavior* is employed to find possible customers or users (travelers or transport units). Another interesting behavior is the `StopProvider` *Behavior* that allows the agent to assume the provider role.

Three *Capabilities* denote the different services that this agent can offer. The first is the *Notification Capability* that notifies about routes and ETA of the different transports that reach the stop or the station. The second is the *Display Capability* that allows to adjust the information displayed on the screen according to the profile of the users. The last one is the *SearchRoute Capability* that allows to calculate and plan a route from the stop or the station to the desired destination.

Finally, we want to stand out that there may be one or more agents within each organizational unit. Moreover, each agent can offer one or more services provided by the unit. So, one agent doesn't necessarily have all the services defined in its unit (except when the agent is alone in the unit). Then, the agent may have more or less *Behaviours* and *Capabilities*, according to the roles that have (or may have) active.

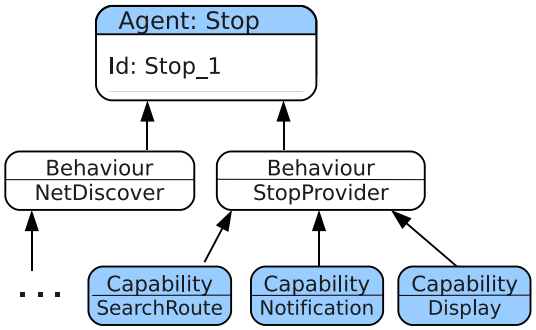

Figure 13: Stop Agent Model

5. Implementation

The Smart Transport System, implemented as an agent-based AmI environment, must allow an easy integration of the mobile services offered by the embedded agents. The proposed framework employs different technologies of hardware and software that are described in the following sections.

5.1 Agent technologies required

After describing the Smart Transport system and the services it can offer, it is necessary to explain the agent platforms used for the implementation of the Smart Transport example. The Bus is designed as a virtual organization supporting an open agent society. The THOMAS platform is used as a platform where agents can enter and leave the organization at any time. Agents using the services of the system have been designed as embedded agents over mobile phone by means of ANDROMEDA[1] and JADE. These platforms are briefly described below.

5.1.1 Embedded Platforms: ANDROMEDA and JADE

- **ANDROMEDA** (ANDROid eMbeddED Agent platform)(Agüero et al., 2008b; Agüero et al., 2008a) is an agent platform specifically oriented to embedded agents on the *Android*[2] operating system. *Android* is a software system specifically designed for mobile devices which includes an operating system, a middleware and key applications. Applications are written using the Java programming language and they run on Dalvik (the *Android* Virtual Machine), a custom virtual machine designed for embedded use, which runs on top of a Linux kernel. ANDROMEDA platform includes all the abstract concepts of the *agent-*π meta-model. The implementation was done using the main API components of *Android*. The agents run as any application of *Android*, because ANDROMEDA platform can be interpreted as a new layer that is inserted into the *Android* architecture, as shown in Figure 14, which is a modification of the original architecture that is shown in (Android, 2014).

- **JADE**[3](Bellifemine et al., 1999) is one of the most popular or relevant platforms that supports the agent execution. This platform is widely used because it provides programming concepts that simplify the

[1] http://users.dsic.upv.es/grupos/ia/sma/tools/Andromeda
[2] Android System, http://code.google.com/android/
[3] http://jade.tilab.com/

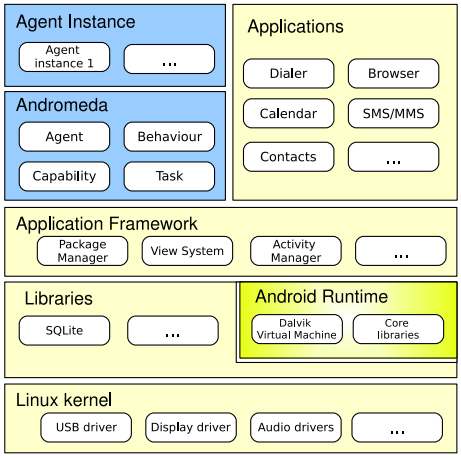

Figure 14: ANDROMEDA *platform over Android architecture.*

MAS implementation. JADE is FIPA compliant in the communication infrastructure between agents. **JADE-Leap** (JADE Light Extensible Agent Platform) is a JADE version used to implement embedded agents, which are executed over mobile phones or devices with limited resources. Figure 15 shows the agent technologies used and the possible process of dialog and interactions between agents and different platforms.

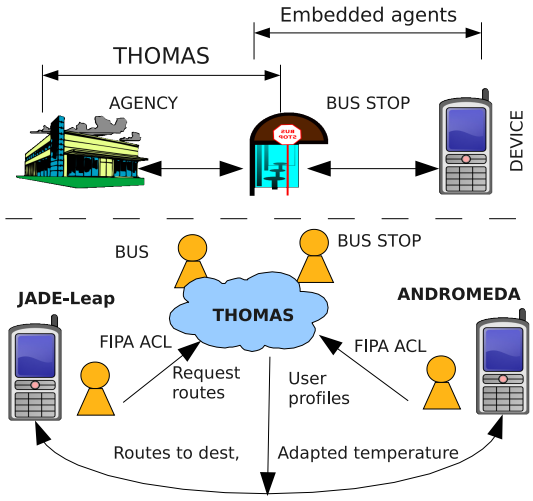

Figure 15: Agent interactions in the case of study.

5.1.2 Organization Platform

THOMAS[4] (MeTHods, Techniques and Tools for Open Multi-Agent Systems)(Carrascosa et al., 2009), is a new open Multi-Agent System architecture consisting of a related set of modules that are suitable for the development of systems applied in environments that work as a "society". Due to the technological advances of recent years, the term "society", where the Multi-Agent System participates, needs to meet several requirements such as: distribution, constant evolution, flexibility to allow members to enter or leave the society, appropriate management of the organizational structure that defines the society, multi-device agent execution including devices with limited resources, and so on. All these requirements define a set of features that can be addressed through the open system paradigm and virtual organizations.

THOMAS architecture basically consists of a set of modular components; the main components are the following: **Service Facilitator (SF)**, this component offers simple and complex services to the active agents and organizations. Basically, its functionality is like a yellow page service and a service descriptor in charge of providing a green page service. **Organization Management System (OMS)**, it is mainly responsible for the management of organizations and their entities. Thus, it allows creation and management of any organization. **Platform Kernel (PK)**, it maintains basic management services for an agent platform. Finally, THOMAS platform used CASE tools that support it: **EMFGormas**. This is an organization-based CASE tool that allows modeling agent and virtual organizations and generates their code. Figure 15 shows how THOMAS is used as the central organization of the proposed Smart Transport System.

5.2 Hardware Components: Devices

Once we have described the agent platforms used to implement the intelligent bus system, we are going to describe the different hardware and software devices used to support these agents. The components forming part of the system are diverse, such as: routers, hubs, LCD screens, mobiles and embedded computers.

The embedded computers are located at stops, stations and transportation units. These computers correspond to the electronic card known as Beagleboard[5], which is a low-cost single-board computer based on Texas Instruments' OMAP device family. The BeagleBoard is a pocket-sized reference board containing a Texas Instruments OMAP3530 system-on-a-chip (SoC) processor (ARM Cortex A-8 core) and it has all the functionality of a basic computer, includes a DSP for accelerated video and audio decoding, a GPU to provide accelerated 2D and 3D rendering that supports OpenGL ES 2.0. The video is provided through separate S-Video and HDMI connections. It has a single SD/MMC card slot supporting SDIO, a USB On-the-Go port, a RS-232 serial connection, a JTAG connection, and two stereo 3.5mm jacks for in/out audio. Additionally, the memory is provided through a PoP chip that includes 256MB of NAND flash memory and 256MB of RAM. Figure 16 shows the hardware architecture used by the BeagleBoard.

Beagleboard implements embedded agents which are developed on **ANDROMEDA**, because it allows to execute Android code into the Beagleboard. This was achieved by modifying the previous work done by EMBINUX[6] and 0xLAB[7]. We obtained that Android was compatible with all electronic components of Beagleboard and other components that we used. Figure 17 shows the software architecture that gives support to Android over Beagleboard.

Another important hardware are mobile phones which run embedded agents for each user. We have used two different devices. The first is the HTC Hero that is based on the Android system and it is able to run **ANDROMEDA** agents. The second is a Galaxy Nexus that allows to run JADE agents.

[4]http://users.dsic.upv.es/grupos/ia/sma/tools/Thomas
[5]http://www.beagleboard.org/
[6]http://labs.embinux.org/
[7]http://0xlab.org/

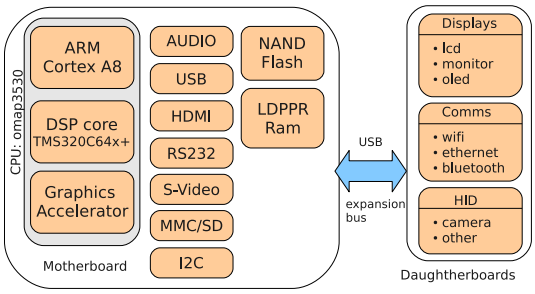

Figure 16: Hardware of Beagleboard.

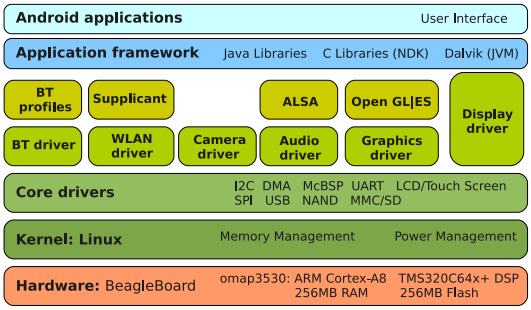

Figure 17: Android architecture over Beagleboard.

Finally, we use a netbook model LG-X110 which runs Android (and ANDROMEDA) as operating system, from the modification of the work done by the Android-x86[8] group. The selected version of Android is able to run on processors Intel-X86 (specifically Intel Atom), as the original version of Android only runs on processors with architecture ARM-V5 or higher.

5.3 Sotfware Components

There are also different software components such as several operating systems, IDEs, ML languages or middleware agent. But for brevity only we discuss the most important for the proposed system. One of the important components to note are different Java Virtual Machines (JVMs) used, such as: J2SE, J2ME and Android. The embedded agents which run over the HTC Hero and the Beagleboard employ the **ANDROMEDA** middleware which at the same time uses Android Dalvik. The agents which run over the Galaxy Nexus and the JADE middleware, employs a Jave ME virtual machine. On the other hand, agents which are located in the Transport Agency use the THOMAS platform that it is supported by J2SE. These last agents run on desktop computers. Finally, Figure 18 shows the different software/hardware components used in this proposal.

Another important software component is the Eclipse IDE which allows to use the EMFGormas tool to implement the system using UML components and, besides, it allows to obtain code templates using plugins for the MDD (Model-Driven Development).

MDD is a fairly new resource in the software engineering field. The main point in MDD is to define a process that is guided by using models, where visual modeling languages are used to integrate a huge diversity of

[8] http://www.android-x86.org/

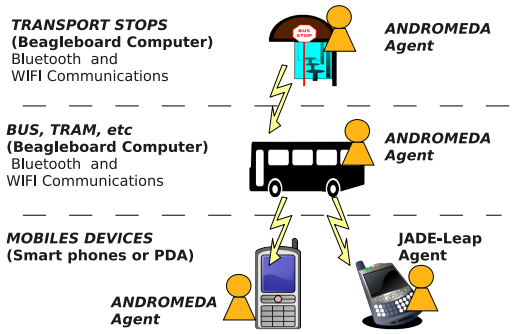

Figure 18: Technologic required in the Smart Transport System.

technologies applied in computing system design (Kleppe et al., 2003). The objective of MDD is to build models that are readable by computers, which can be understood by automatic tools in order to generate templates, code, test models, and can integrate the code into multiple platforms and technologies.

MDD proposes transferring the philosophy of object-oriented programming to one based on models, changing the software artifacts of the programming technology: from "all is an object" to "all is a model"(Bézivin, 2005). MDD approximation uses and creates different models at different abstraction levels to fuse and combine them when needed to implement the application. The transformations allows the models to be automatically converted, i.e., a model with a given abstraction level, to become another one with a different level of abstraction. Transformations are relational entities that describe mapping rules between concepts of two different meta-models.

6. Implementing agents using MDD

Once we know the different components of the system, the design continues with the creation of embedded agents and the Transport Agency system, which is modelled as an organization that will be executed over the THOMAS platform (Carrascosa et al., 2009). The agent implementation starts by defining the different roles and activities to be played by the agents. In this case, for reasons of brevity we only describe agents running on mobile phones (the PA agents) using ANDROMEDA or JADE platforms.

This step is accomplished using the **EMFGormas Toolkit** that allows users to draw/design (UML-Like) all the agent components according to the agent model of π**VOM** (explained in section 4.2). Once the agent model is designed, EMFGormas Toolkit allows verifying the designed system (in a semantic and syntactic way). Figure 19 shows a snapshot of how to design (draw) the components of the client agent model and his correspondence with the *agent-*π meta-model.

Next step is the generation of agent code using this toolkit. The designer must decide in which mobile platform will be executed each agent. In this case, the *user* agent is selected to be executed over an agent platform: ANDROMEDA or JADE. The **ANDROMEDA** platform has been designed using the same concepts of the *agent-*π model. So, an agent in ANDROMEDA is implemented using the same concepts employed in the abstract model. This design greatly simplifies the automatic transformation between models, because the ANDROMEDA models are very similar to the *agent-*π model. In this case, the needed transformation rules are mainly model-to-text, generating directly ANDROMEDA code which can be combined with additional user code. The code template generated by the transformations is shown in Figure 20. A more detailed explanation of this process can be found in (Agüero et al., 2013)

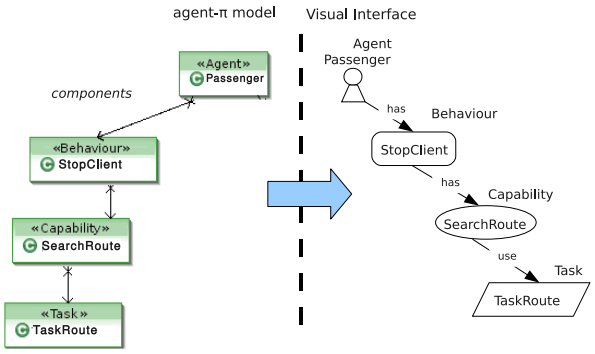

Figure 19: Illustration of how to design an agent using the GUI.

```
public class Passenger extends Agent {
     public void init(){
        . . .
      //define agent components
                                                    Behaviour
      Behaviour StopClient= new Behaviour("StopClient");
                                                    Capability
      Capability SearchRoute =  new Capability("Route");

       Task TaskSearch =new Task("TaskSearch");

        . . .
        //add objects into of the agent components
        SearchRoute.addTaskRun(TaskRoute);
        StopClient.add(SearchRoute);
        addbehav(StopClient);
    }  // ----------- end init()
                                                    Task
    class TaskSearch extends Task{
       doIt {
         //here the user write the  code
       }  // ----------- end doIt
       . . .
    }  // ----------- end Class Task

    . . .
}  // ----------- end Class Agent
```

Figure 20: Code template of ANDROMEDA *Agent.*

The transformation from agent models to **JADE** code must be done employing two phases: (i) the first phase translates from a platform independent model to a JADE-dependent model, and allows to obtain a correspondence among the abstract concepts of the agent-π model to JADE concepts; (ii) the second phase allows to translate the obtained JADE models into executable JADE code. For instance, these transformation processes convert a `Behaviour` of na agent-π into a `ParallelBehaviour` of JADE. An example of the code template generated after these processes is shown in Figure 21.

Finally, the code generated automatically by the tool can be edited by the user to complete the *user* agent development with the specific details of the execution platform.

```
public class Passenger extends Agent {
    ...
   protected void setup(){                                    Behaviour
      ParallelBehaviour StopClient =
         new ParallelBehaviour( ParallelBehaviour.WHEN_ALL );

      StopClient.addSubBehaviour( new SearchRoute(this) );
      ...
      addBehaviour(StopClient);
    }  // ----------- end setup()
                                                              Capability
      class SearchRoute extends OneShotBehaviour {
          public SearchRoute(Agent a) {
            super(a); }

          public void action() {
            ... //check condition == true
            addBehaviour(new TaskRoute(this) );
          }
      } // ----------- end OneShotBehaviour

      ...                                                     Task
      class TaskRoute extends Behaviour {
          public TaskRoute(Agent a) {
             super(a); }

          public void action()  {
              //...this is where the code Task goes !!
          }
      } // ----------- end Behaviour

} // ----------- end class Agent
```

Figure 21: Code template of JADE-Leap Agent.

7. Conclusions

This paper have shown how can be modeled and implemented a Smart Transport System with the goal of improving the interconnection between people and transport providers in a city. The system has been defined as a virtual organization where agents can enter or leave into the system consuming or offering services. In this sense the scenarios have been modeled using platform-independent models. Subsequently, the unified models were transformed into platform-specific implementations. The proposed system have shown how the deployment framework can integrate into its layers different environment devices, different agent platforms and different implementation frameworks.

8. Acknowledgment

This work was partially supported by TIN2015-65515-C4-1-R project of the Spanish government.

9. References

Adam, E., Strugeon, E. G.-L., and Mandiau, R., 2012. MAS architecture and knowledge model for vehicles data communication. *ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal*, 1(1):23–31.

Agüero, J., Rebollo, M., Carrascosa, C., and Julián, V., 2008a. Developing intelligent agents in the Android platform. In *Sixth European Workshop on Multi-Agent Systems (EUMAS 2008)*, volume CD Press, pages 1–14. Bath, England.

Agüero, J., Rebollo, M., Carrascosa, C., and Julián, V., 2008b. Does Android Dream with Intelligent Agents? In

International Symposium on Distributed Computing and Artificial Intelligence 2008 (DCAI 2008), volume 50, ISBN: 978-3-540-85862-1, pages 194–204. Salamanca, Spain.

Agüero, J., Rebollo, M., Carrascosa, C., and Julián, V., 2009. Developing Virtual Organizations Using MDD. In CEUR-WS, editor, *CAEPIA09 Workshop on Agreement Technologies(WAT 2009)*, volume 635, ISSN: 1613-0073, pages 130–141. Sevilla, Spain.

Agüero, J., Rebollo, M., Carrascosa, C., and Julián, V., 2012. Developing Pervasive Systems as Service-oriented Multi-Agent Systems. In *7th International ICST Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services (MobiQuitous 2010)*, volume 73, ISBN 978-963-9995-20-8 of *Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*, pages 78–89. Springer Berlin Heidelberg, Sydney, Australia.

Agüero, J., Rebollo, M., Carrascosa, C., and Julián, V., 2013. Towards the development of agent-based organizations through MDD. *International Journal on Artificial Intelligence Tools (IJAIT)*, 2(2):1350002:1–1350002:35. doi:10.1142/S0218213013500024.

Android, 2014. The Android Software Development Kit (SDK).

Argente, E., Julian, V., and Botti, V., 2008. MAS Modelling based on Organizations. In *9th Int. Workshop on Agent Oriented Software Engineering (AOSE08)*, pages 1–12.

Augusto, J., 2007. Ambient intelligence: the confluence of ubiquitous/pervasive computing and artificial intelligence. *Intelligent Computing Everywhere*, pages 213–234.

Bellifemine, F., Poggi, A., and Rimassa, G., 1999. JADE - A FIPA -compliant agent framework. In *Proceedings of the Practical Applications of Intelligent Agents*.

Bézivin, J., 2005. On the unification power of models. *Software and Systems Modeling*, 4(2):171–188.

Billhardt, H. and et al., 2016. Towards Smart Open Dynamic Fleets. *Multi-Agent Systems and Agreement Technologies. LNCS*, 9571:410–424.

Brønsted, J., Hansen, K., and Ingstrup, M., 2007. A survey of service composition mechanisms in ubiquitous computing. In *Second Workshop on Requirements and Solutions for Pervasive Software Infrastructures (RSPSI)'at Ubicomp*.

Carrascosa, C., Giret, A., Julian, V., Rebollo, M., Argente, E., and Botti, V., 2009. Service Oriented Multi-agent Systems: An open architecture. In *Autonomous Agents and Multiagent Systems (AAMAS)*, pages 1–2.

Chamoso, P. and de la Prieta, F., 2015. Swarm-Based Smart City Platform: A Traffic Application. *ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal*, 4(2):89–97.

Davidsson, P., Henesey, L., Ramstedt, L., Törnquist, J., and Wernstedt, F., 2005. An analysis of agent-based approaches to transport logistics. *Transportation Research part C: emerging technologies*, 13(4):255–271.

Dignum, V., 2003. *A model for organizational interaction: based on agents, founded in logic*. Phd dissertation, Utrecht University.

Isabel, A. F. and Fernandez, R. F., 2015. Simulation of Road Traffic Applying Model-Driven Engineering. *ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal*, 4(2):1–24.

Kleppe, A., Warmer, J. B., and Bast, W., 2003. *MDA Explained: The Model Driven Architecture: Practice and Promise*. Addison-Wesley Professional.

Lujak M., O. S., Giordani S., 2015. Route guidance: Bridging System and User Optimization in Traffic Assignment. *Neurocomputing*, 151(1):449–460.

Mees, P., 2000. *A very public solution: Transport in the dispersed city*.

Zambonelli, F., Jennings, N. R., and Wooldridge, M., 2003. Developing multiagent systems: The GAIA methodology. *ACM Trans. Softw. Eng. Methodol.*, 12(3):317–370. ISSN 1049-331X.