J. Carbo, N. Sanchez-Pi & J. M. Molina (2018) Agent-based simulation with NetLogo to evaluate ambient intelligence scenarios, *Journal of Simulation*, 12:1, 42-52.

Agent-based Simulation with Netlogo to Evaluate AmI Scenarios

Javier Carbo[1], Nayat Sanchez-Pi*[2], and José Manuel Molina[3]

[1] *Department of Informatics, Universidad Carlos III de Madrid,, Colmenarejo, Madrid, Spain. javier.carbo@uc3m.es*
[2] *Computer Science Department, Institute of Mathematics and Statistics, Universidade do Estado do Rio de Janeiro, Rio de Janeiro (RJ) Brazil. *corr: nayat@ime.uerj.br*
[3] *Department of Informatics, Universidad Carlos III de Madrid,, Colmenarejo, Madrid, Spain. molina@ia.u3m.es*

Dated: March 7, 2016

Abstract

In this paper an agent-based simulation is developed in order to evaluate an AmI scenario based on agents. Many AmI applications are implemented through agents but they are not compared to any other existing alternative in order to evaluate the relative benefits of using them. The proposal simulation environment developed in Netlogo analyse such benefits using two evaluation criteria: First, measuring agent satisfaction of different types of desires along the execution. Second, measuring time savings obtained through a correct use of context information.

So, here, a previously suggested agent architecture, an ontology and a 12-steps protocol to provide AmI services in airports, is evaluated using a NetLogo simulation environment. The present work uses a NetLogo model considering scalability problems of this application domain but using FIPA and BDI extensions to be coherent with our previous works and our previous JADE implementation of them.

The NetLogo model presented simulates an airport with agent users passing through several zones located in a specific order in a map: passport controls, checking boxes of airline companies, boarding gates, different types of shopping. Although initial data in simulations are generated randomly, and the model is just an approximation of real-world airports, the definition of this case of use of Ambient Intelligence through NetLogo agents opens an interesting way to evaluate the benefits of using Ambient Intelligence, which is a significant contribution to the final development of them.

Keywords: Agents, Ambient Intelligence, Context-Aware, Ubiquitous Techniques, Software Simulations

1 Introduction

Virtual simulations frameworks, as (Serrano et al., 2014), have been widely used to evaluate emergency plans in indoor environments. But there are also other scenarios that are very complex to evaluate and this is the case of Internet of Things (IoT) scenarios. Electronic sensors, that act as autonomous computational devices (smartphones, cameras, i-watches, thermical, infrared sensors, drones, etc.), are rapidly becoming ubiquitous capturing daily life activities in all kind of environments (at home, at the office, and even in a bigger scale such as the so called smart cities). The Ubiquity of sensors makes possible the idea envisioned by Weiser in 1991, that presents a world where computers would be embedded in everyday life where people could communicate with computers providing customized services in a way where the network infrastructure should be transparent to the user itself (Weiser, 1991). This idea is mostly known as Ambient Intelligence (in advance AmI). AmI emphasizes on greater user-friendliness, more efficient services support, user-empowerment, and support for human interactions. In this vision, people will be surrounded by intelligent and intuitive software entities embedded in everyday sensors around us recognizing and responding to the particular needs of individuals in an invisible way (Kovács, Kopácsi, 2006).

AmI represents, in other words, a new generation of user-centered computing environments aiming to find new ways to obtain a better integration of the information technology in everyday life activities obtained by ubiquitous sensors. Ideally, people in an AmI environment does not notice these sensors, but they will benefit from the services they are able to provide. Such sensors are aware of the people presence in those environments by reacting to their gestures, actions and context (Aarts et al., 2001). AmI environments are then integrated by several autonomous computational devices of modern life ranging from consumer electronics to mobile phones. AmI has several spheres of application like: Transportation (for instance providing adaptive bus routes or adaptive traffic lights), Health (predicting heart attacks, providing faster ambulance calls, etc), Home (providing more efficient energy-uses), etc. Recently the interest in AmI Environments has also been focused on demanding highly innovative services in critical areas like airports and train stations in order to increase security, to reduce the length of rows and to better provide updated travel information.

In order to work efficiently, software running on these sensors may have some knowledge about the user, it means that they need to cooperate with other sensors sharing knowledge about the user without interfering with user's daily life activities. Due to highly dynamic properties of the above introduced environments, the software system running on sensors has to face problems such as: user mobility, service failure, resources or goal changes which may happen in any moment. To cope with these problems, such system must senses the environment, and acts on it, over time in pursuit of its own benefit.

That is why there is a need of special kind of software that should combine ubiquity, context-awareness, intelligence and natural interaction in an AmI environment. The system has also to adapt not only to changes in the envi-

ronment, to be autonomous and self-managed but also to the user requirements and needs. The kind of software that fits with such requirements is Agent technology. Agents intend to reproduce human behaviour through abilities such as autonomy, proactivity, adaptability, planning, and so on (Wooldridge, Jennings, 1995). Agents adapt not only to changes in the environment, to be autonomous and self-managed they also adapt to the user requirements and needs. This is the underlying foundation of the concept of agent, computer systems capable of independent actions on behalf of its user (Durfee, Rosenschein, 1994).

We have been working in the confluence of both research areas for years. Specifically we have developed a distributed agent-based platform to provide AmI services to users in an airport domain (Sánchez-Pi et al., 2008),(Sanchez-Pi et al., 2008). But we found out that evaluation of AmI systems is a difficult problem and seldom tackled in literature because of the privacy issues, hardware costs and the open and dynamic nature of this kind of systems. Then, instead of universal real-life evaluations, the most popular way to evaluate them is to observe their performance in particular application scenarios through virtual simulations as (Serrano et al., 2014) shows defining a complex and complete framework to evaluate emergency plans in indoor environments. This is also our case.

This paper presents a two-fold criteria evaluation of the benefits of using AmI in a particular domain application where we have previously worked in: an airport. We use NetLogo to simulate particular and collective behaviors in an airport. This NetLogo simulation has the objective of comparing user satisfaction due to the delays of agents in rows with and without AmI. We are trying to find out how AmI could help when a high number of agents are accessing different services through rows, and through the use of location indications as it happens in real-life airports. Particularly, we use simulations to compare extra time-savings and the level satisfaction of agent goals provided with AmI and without it. Such goals are for instance, avoiding missing the plane (it provides a major satisfaction), meeting shopping interests (it provides minor satisfaction) and reducing time spent waiting in rows (it also provides minor satisfaction).

The rest of the paper is structured as followed: section 2 presents a contextualized related work; section 3 is where our previously defined ontology, protocols, agent architecture and airport scenario is summarized; section 4 describes the main contribution of this paper: the coherent adaptation of the elements presented in section 3 to a NetLogo model and the simulation experiments results. At last, conclusions are laid down.

2 Foundations

In the literature, there are several approaches developing platforms, frameworks and applications for offering context-aware services where agent technology has been applied (as we do) in order to provide the right information at the right time to its users. These applications also include location-based services as our work uses Aruba technology to do such task) pushing information and events

to the user (Poslad et al., 2001).

Application domains of such combination of the three elements: AmI, Agents and Location Technology, they are: TeleCARE project for supporting virtual elderly assistance communities (Afs, 2003); a planning agent AGALZ using case-based reasoning to respond to events and monitor Alzheimer patients' health care in execution time (Corchado et al., 2008); SMAUG (nie, 2004) is a multi-agent context-aware system that allows tutors and pupils of a university to fully manage their activities; AmbieAgents (Lech, Wienhofen, 2004) proposes an agent-based infrastructure for context-based information delivery for mobile users; There is also a case study that consists in solving the automation of the internal mail management of a department that is physically distributed in a single floor of a building plant (restricted and well-known test environment) using ARTIS agent architecture Bajo et al. (2008); and an AmI architecture to provide an agent-based surveillance system applying an agent-orientated methodology (pav, 2007). None of them, however has been applied to an airport domain as we are doing the past years.

On the other hand, as agents seem to be the appropriate solution for AmI environments since they provide autonomy and proactivity. In (O'Hare et al., 2004), O'Hare et al. advocate the use of agents (as well as we do) as a key enabler in the delivery of AmI. It could be assumed that agents are abstractions for the interaction within an AmI environment, and the single aspect that agents need to ensure is that their behaviour is coordinated. This assumption leads to the use of very simple reactive agents without any cognitive capability (Brooks, 1985). But depending on the domain, agents reproducing intelligent behaviours need decision rules that take into consideration complex context information (location, user profile, type of device, etc.) that has to be interpreted in which these interactions take place. Complex knowledge processing is required to offer, provide and consume services on behalf of humans. We need agents for helping humans in their knowledge-related tasks. Agents that can some how understand people's emotions and rational behaviour, or that can at least attempt to process complex information on our behalf. Taking care this, the so called cognitive architectures accomplishes not only the task to regulate the interaction but also to manage complex decision making. The most extended and promising cognitive architecture is based on the Belief-Desire-Intention paradigm. These three levels of knowledge allow agents to cope with complex decisions supposedly as humans do, following a particular reasoning algorithm (Rao, Georgeff, 1995). Communications between agents aslo have the intention to emulate human dialogs through the use of predefined sequences of linguistic performatives as IEEE-accepted FIPA communication standards define them (FIP, 1997).

BDI-based agent platforms such as JADEX (Pokahr et al., 2003) or JASON (Bordini, Hbner, 2005), and FIPA-compliant platforms as JADE (JAD, 2010) produce agents that are often conceptually heavy models and intensive CPU consuming implementations. This makes them difficult to use as simulation tools when a relatively high number of agents are involved, as we observed with our JADE implementation of the airport AmI application.

Therefore an alternative is the use of lightweight-agent paradigm extended in

simulation research area, known as Multi-Agent based Social Simulation MABS, which is largely used in economics, traffic flow, etc... It allows the analysis of complex interactions with heterogeneous individuals (Sichman et al., 1998), and it typically represents agents in a very simplistic or atomistic approach. This simplification is needed to avoid the complexity of BDI-based, FIPA-compliant agents. This kind of simple agents are produced by platforms as MASON (cs.gmu.edu/ eclab/projects/mason), RePast (repast.sourceforge.net), SMNP (www.monfox.com/dsnmp_sim.html) and Netlogo (ccl.northwestern.edu/netlogo). Some approaches try to address this limitation through the inclusion of cognitive skills in MABS platforms such as (Caballero et al., 2011) did with the integration of MASON and JASON, and as the proposed FIPA and BDI extensions (Sakellariou et al., 2008)to NetLogo do in our work.

Specifically, NetLogo is a programmable modeling environment for simulating natural and social phenomena. It is particularly well suited for modeling complex systems developing over time. Developers can give instructions to hundreds or thousands of independent agents all operating concurrently. This makes it possible to explore the connection between the micro-level behavior of individuals and collective behavior that emerge from the interaction of many individuals. Two approaches very close to ours use NelLogo models to crowd evacuation in emergency situations are (Wagner, Agrawal, 2014) and (Dawson et al., 2011).

A wide variety of computational approaches have been proposed for simulation of collective behavior (Pan et al., 2007). In this work, authors define three classification categories (1) fluid or particle systems, (2) matrix-based systems, and (3) emergent systems, but there are also specific AmI simulators designed to evaluate general AmI systems such as UbiWise (Barton, Vijayaraghavan, 2002), Tatus (O'Neill et al., 2005) and UbiReal (Nishikawa et al., 2006), but they are focused on the interaction of a real user with the system, they are not designed to develop and run fully automated executions of a particular AmI scenario as we do.

3   Problem definition

In this work, the scenario is defined by a 2D grid of pixels, where special rooms are represented by a pixel accessible from any neighbour pixel. Each individual is represented by an autonomous entity, an agent, whose main goals is to either taking the plane or recovering baggage and exit. Many individuals can be located in the same pixel but in each iteration just one of them is interacting with the services/information provided by the room. Several agents are defined with cognitive capacity based on BDI model. This means that each agent has a set of beliefs that include the relevant locations (pixels) the agent has visited, personal beliefs about itself, and beliefs based on information/services received from other agents. This belief set changes while it is moving through the grid and when new information/services arrived from other agents.

3.1 AmI in an airport

AmI has application into different sectors in daily life. One important sector is Transportation and specifically Airports. AmI intelligence can be presented into this domain as an information system to offer customized services to different type of users (agent roles): passengers, crew and airline staff. We are familiar to this specific problem because we have been working during years with this application domain (Sánchez-Pi et al., 2007) . We previously developed a centralized system using Appear Networks Platform (www.appearnetworks.com) and Aruba Wi-Fi Location System (www.arubanetworks.com) and later we developed a distributed agent-based platform using the same technology (Sánchez-Pi et al., 2008), (Sanchez-Pi et al., 2008) . In both approaches, we assume an initial minimal profile known of the user: name (identifier), agent role, passport data (nationality, physical aspect), carrying suitcases, shopping interests and travel info (flight numbers, companies, origin and target) in order to suggest the best-fitting services.

Involved knowledge in order to provide context-awareness in an airport was also defined in an ontology. For building the ontology, we have followed Noy and McGuiness proposal which consists on an iterative process based on the methodology proposed by Gruninger and Fox (Grninger, Fox, 1995) who defined the competency questions used in the scope and goal step, and the development of the classes hierarchy based on Top-Down and Bottom-Up strategies. In our previous works (Fuentes et al., 2006), we defined the problem of context definition in ubiquitous applications. The high level ontology definition that we have described follows the categorization defined by Schilit (Schilit et al., 1994), that divided contextual information in computing context (network, devices, etc.), user context (preferences, location etc.) and physical context (temperature, traffic, etc.). The ontology definition gathers these concepts and their properties and relationships for accomplished this contextual definition. An important contextual information about the user to take into account is the location. In order to acquire location information we use Aruba Networks which is a location tracking solution which uses an enterprise wide WLAN deployment to provide precise location tracking of any Wi-Fi device in the research facility. The RF Locate application can track and locate any Wi-Fi device within range of the Aruba mobility infrastructure. Using accurate deployment layouts and triangulation algorithms devices can be easily located include PDAs, rogue APs/Clients, VoWLAN phones, laptops, Wi-Fi asset management tags. Although many alternatives exist, most successful indoor location techniques are based on the RSSI triangulation method. But basic RSSI triangulation does not provide sufficient accuracy for many of the users of location information. While techniques such as analysis of building material and walk around calibration can improve the accuracy of RSSI measurements, they add considerable expense and complexity to the network installation, and the improvement in accuracy erodes over time, as the environment changes. WLANs are cellular, where neighbouring APs operate on different RF frequencies (channels) to avoid interference. The Wi-Fi medium access control layer (MAC) allows any station
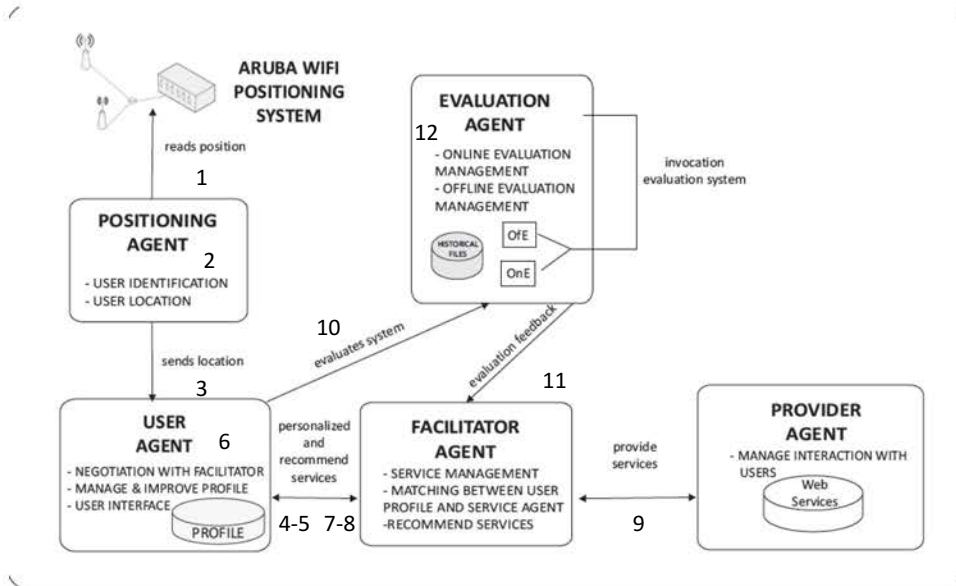
Figure 1: Schema of the multi-agent architecture.

in a basic service set to transmit at any time. Therefore all stations (including the AP) should be listening on the cells RF channel all the time, to avoid missing transmissions. The above said, explained the use of time-stealing APs to monitor other channels while nominally providing coverage of their own cell. An alternative technique is to deploy dedicated RF monitors named Air Monitors (AMs). AMs are identical to APs (the same hardware and software), but they are configured permanently in listening mode. This is a very useful capability, because the AMs contribute not just to location accuracy but also by improving security coverage, detecting RF sources that may be security risks or interferers. The drawback of using dedicated AMs is that they add to the capital costs of the network.

When a wireless device enters the network, it immediately is established the position of the client device. Once the client localized, he can negotiate the set of applications depending on his physical position. In our distributed approach based on agents, entities are in charge of distribute contextual information in order to access the information in a more efficient way.

3.2 Agent System architecture

The proposed agent-based architecture manages context information to provide personalized services to users. As it can be observed in Figure 1, it consists of five different types of agents that cooperate to provide an adapted service. *User agents* are configured into mobile devices or PDAs. *Provider Agents* supply the

different services in the system. A *Facilitator Agent* links the different positions to the providers and services defined in the system. A *Positioning Agent* communicates with the Aruba positioning system (Sánchez-Pi et al., 2007) to extract and transmit positioning information to other agents in the system. Finally, an *Evaluator Agent* stores log file in order to acquire a future evaluation criteria of the MAS system developed for AmI scenarios.

Eight concepts have been defined for the ontology of the system. The definition is: *Position* (XCoordinate int, YCoordinate int), *Place* (Building int, Floor int), Service (Name String), *Product* (Name String, Characteristics: List of Feature), *Feature* (Name String, Value String), *Context* (Name String, Characteristics: List of Features), *Profile* (Name: String, Characteristics: List of Features). Our ontology also include six predicates with the following arguments:

Our ontology also includes five predicates and an action with the following arguments: *HasLocation* (place, Position, AID), *HasServices* (Place, Position, List of Services), *isProvider* (Place, Position, AID, Service), *HasContext* (What, Who), HasProfile (Profile, AID), and *Provide* (Product, AID).

The interaction with the different agents follows a process which comprises the following phases:

1. The ARUBA positioning system is used to extract information about the positions of the different agents in the system. This way, it is possible to know the positions of the different User Agents and thus extract information about the different Providers Agents that are available for this location.

2. The Positioning Agent reads the information about position (coordinates x and y) and place (building and floor) provided by the Aruba Positioning Agent by reading it from a file, or by processing manually introduced data.

3. The Positioning Agent (Positioning Agent.Send
Location) communicates the position and place information to the User Agent.

4. Once a User Agent is aware of its own location, it communicates this information to the Facilitator Agent in order to find out the different services available in that location.

5. The Facilitator Agent informs the User Agent about the services available in this position.

6. The User Agent decides the services in which it is interested.

7. Once the User Agent has selected a specific service, it communicates its decision to the Facilitator Agent and queries it about the service providers that are available.

8. The Facilitator Agent informs the User Agent about the identifier of the Provider Agent that supplies the required service in the current location.

9. The User Agent asks the Provider Agent for the required service through the Facilitator Agent.

10. Once the interaction with the Provider Agent has finished, the User Agent provides the evaluation information to the Evaluator Agent.

11. The Evaluator Agent updates the contents of the user profile with the evaluation information and send this information to the Evaluator Agent.

12. The Evaluator Agent stores this user profile for future further analysis.

The corresponding number of each phase is shown in figure 1 to facilitate the understanding of the communication flow between agents to request a particular service. So with this agent definition, ontology and protocols, we have completely defined the AmI agent-based application domain we are going to evaluate using a simulated model of an airport.

4   Agent simulation with NetLogo

In this section we define (using NetLogo) a simulated scenario where the already described in section 3 MAS architecture for context-aware problems can be applied. This scenario would allow us to consider two evaluation criteria that would become more discriminant when there are many agents in the system: First, satisfaction provided by Ambient Intelligence, which is linked to the accomplishment of agent's goals through an appropriate use of time. This concept is computed according to three satisfaction evaluation criteria: whether we achieved the main goal (to avoid missing the plane) or not, how much we met desired activities (shopping) and how much we avoided undesired activities (time spent in queues/rows). For instance an agent nor satisfied at all would have missed the plane, and an agent would be mostly not satisfied if it did not buy any gift according to its shopping interests of if it spent many time in rows.

Second, time savings obtained through the use of context information. A correct use of information in our domain stands for avoiding going to the information panels of the airport, and avoiding going around while shopping (through the use of location indications). For instance, an agent did not save any time if it would have walked to the flight information panel and to the boarding information panel, furthermore it would have taken a detour (instead following a straight course) to reach the provider that fits its shopping interests. These time savings can be obtained using information provided by AmI. The corresponding difference in the steps followed by ingoing agents with and without AmI in Figures 2a and 2b are:

1. request the service from Boarding Info.

2. request the service from Checkin Box.

3. request the service from Passport Control.

4. request the service from Shops (until finding the one that matched with the shopping interest of the agent).

5. request the service from Boarding Gates.

The corresponding followed steps by outgoing agents with and without AmI in Figure 3a and 3b are:

1. request the service from Baggage Info.

2. request the service from Baggage Belt.

3. request the service from Shops (until finding the one that matched with the shopping interest of the agent).

4. request the service from Passport Control.

5. going outside.

So, this NetLogo model of an airport includes several types of User and Provider Agents (besides the already mentioned Positioning, Facilitator and Evaluator agents of our MAS architecture for context-aware problems). User agents may be passengers, crew and staff, but additionally they may be of two types (outgoing and ingoing) passing through several services located in a specific order in a map:

- Outgoing agents go through main entrance, flight information panel, checking box, passport control, shops, boarding information panel and boarding gates.

- Ingoing agents go through boarding gate, baggage information panel, baggage belt, shops, passport control and main entrance.

The eight concepts and six predicates that formed the ontology of the system were used in the FIPA communications in NetLogo. The equivalent OWL ontology can be obtained using the OWL-API (3.1.0) (Polhill, 2015) that extracts state and structure ontologies from an existing Netlogo model. We can observe how the elements of the ontology were used in the next couple of FIPA communications examples of our model:

```
(turtle 51):
["inform" "sender:0" "receiver:51" "content:" "isProvider (Place
(Building Airport ; Floor 0); Position: Belt (patch 18 6) ;
AID: 51 ; Service (Name: Baggage-Delivery) )"]

(turtle 2):
["request" "sender:51" "receiver:2" "content:" "Provide (Product
(Name: Baggage-Delivery ; Characteristics: Baggage-Number 1 ) ;
AID: 51 )"]
```

We assume that each of the user agents has defined a particular predefined profile (traveling profile and personal profile), corresponding to the features of the profile concept of our ontology, that gives values to the next attributes:

- How much interest the agent has in each type of shop.

- How much baggage is carrying (number of suitcases).

- How much estimated danger perception may produce to external observers (due to physical aspect, nationality, etc.).

- Flight number.

User Agents go shopping if they have enough (estimated) time to do it. We use randomly generated initial data of passenger profiles, so the model is just an approximation of real-world airports.

The concept Service is instantiated with Airport services that are provided by Agent Providers:

- Checking box.

- Passport control.

- Shops.

- Baggage belt.

- Boarding gate.

Furthermore queues are formed in services (checking boxes, passport controls, shops, baggage belts and boarding gates), and User Agents have to wait until the Agent Provider is not busy. We assume that information panels do not consume time and do not produce any row.

In order to evaluate the benefits of using context with our MAS architecture, there will be some User Agents that use AmI and others do not. Information panels are Facilitator agents for the agents using Am and Provider agents for the agents not using AmI. Each of these agents using AmI would be executing the communications with Positioning and Facilitator agents (included in the 12-steps protocol described in section 3), and we assume that such communications also involve a relative small elapsed time and also form rows to attend User Agents. But on the other hand, for instance, User Agents using context do not require to pass through information panels, and they know the exact location of the most interesting shops (for that particular agent) thanks to the communications with Facilitator agents, avoiding a random walk through the shops that the users that do not use AmI have to take. We also assume that moving through the map requires time (agents move 1 position per iteration) and providing services has an estimated time (random distribution of different types that depends on features of the profile: the more baggage, the more time in checkin box, the more danger perception, the more time in passport control). Since the same instance of our user agents do not repeat model executions,

Evaluator agent has no sense in this simulation. Otherwise, evaluator agent would allow agents using AmI to know a priori what checkin box (because the user always travels with the same company) or the boarding gate (because the user always travels to the same destination) or the shops to purchase in (since it knows the shopping preference) whithout the participation of the Facilitator agent.

We can observe these difference in the sequence of intentions (coded in reversed order) that the four types of agents execute. For instance, ingoing agents that do not use Ambient Intelligence have to execute intentions for moving to the baggage info screen in order to know the belt number corresponding to its flight, and move through different shops until it finds out the most interesting shop it was looking for.

```
add-intention "move-to-output" "in-output"
add-intention "pass-control" "past-control"
add-intention "move-to-control" "in-control"
add-intention "shopping" "shopped"
add-intention "move-to-shops" "in-shops"
add-intention "collect-baggage" "baggage-collected"
add-intention "move-to-belt" "in-belt"
add-intention "ask-baggage-info" "informed-belt-baggage"
add-intention "move-to-baggage-info" "in-baggage-info"
```

On the other hand, agents that use Ambient Intelligence do not require to move to the baggage info screen, and they move directly to the most interesting shop as the next Netlogo code shows:

```
add-intention "move-to-output" "in-output"
add-intention "pass-control" "past-control"
add-intention "move-to-control" "in-control"
add-intention "shopping" "shopped"
add-intention "move-to-interestingshop" "in-interestingshop";;
add-intention "collect-baggage" "baggage-collected"
add-intention "move-to-belt" "in-belt"
add-intention "ask-baggage-info" "informed-belt-baggage"
```

Outgoing agents show similar differences according to the use/not use of Ambient Intelligence. Outgoing agents that do not use Ambient Intelligence require Netlogo moving intentions towards checkin and gate info screens in order to know the assigned checkin box and boarding gates. Additionally these agents would move around shops until they find out the most interesting shop they were looking for.

```
add-intention "move-to-gate" "in-gate"
add-intention "query-gate" "informed-gate"
add-intention "move-to-gate-info" "in-gate-info"
add-intention "shopping" "shopped"
add-intention "move-to-shops" "in-shops"
```
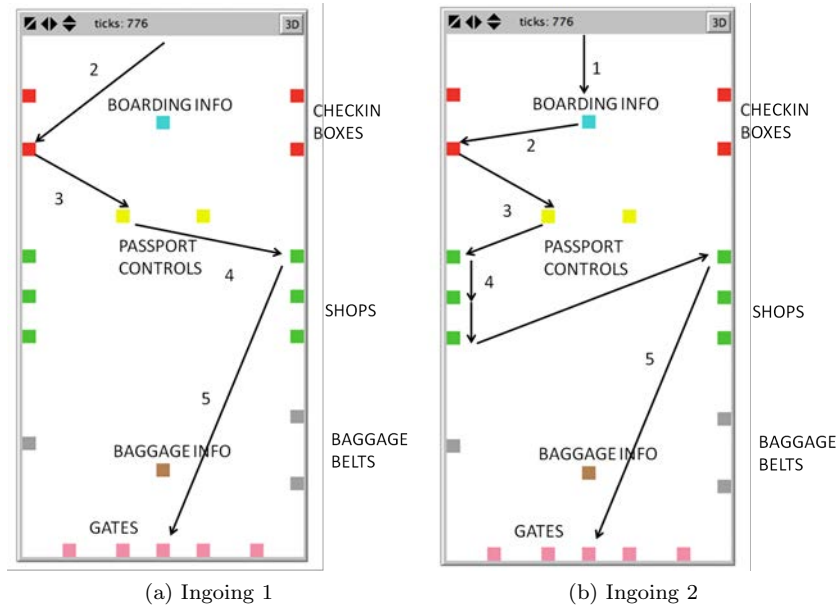
(a) Ingoing 1　　　　　　　　(b) Ingoing 2

Figure 2: Followed steps by ingoing agents with AmI

```
add-intention "pass-control" "past-control"
add-intention "move-to-control" "in-control"
add-intention "request-checkin" "done-checkin"
add-intention "move-to-checkin" "in-checkin"
add-intention "query-checkin" "informed-checkin"
add-intention "move-to-checkin-info" "in-checkin-info"
```

While outgoing agents that use Ambient Intelligence would not require going to the info screens, and they move directly to the most interesting shop as it shows the next code corresponding to their Netlogo intentions to be executed in reversed order:

```
add-intention "move-to-gate" "in-gate"
add-intention "query-gate" "informed-gate"
add-intention "shopping" "shopped"
add-intention "move-to-interestingshop" "in-interestingshop"
add-intention "pass-control" "past-control"
add-intention "move-to-control" "in-control"
add-intention "request-checkin" "done-checkin"
add-intention "move-to-checkin" "in-checkin"
add-intention "query-checkin" "informed-checkin"
```

The definition of this model allows us to simulate several runnings of high-populated agent systems moving from airport main entrance to boarding gates

13

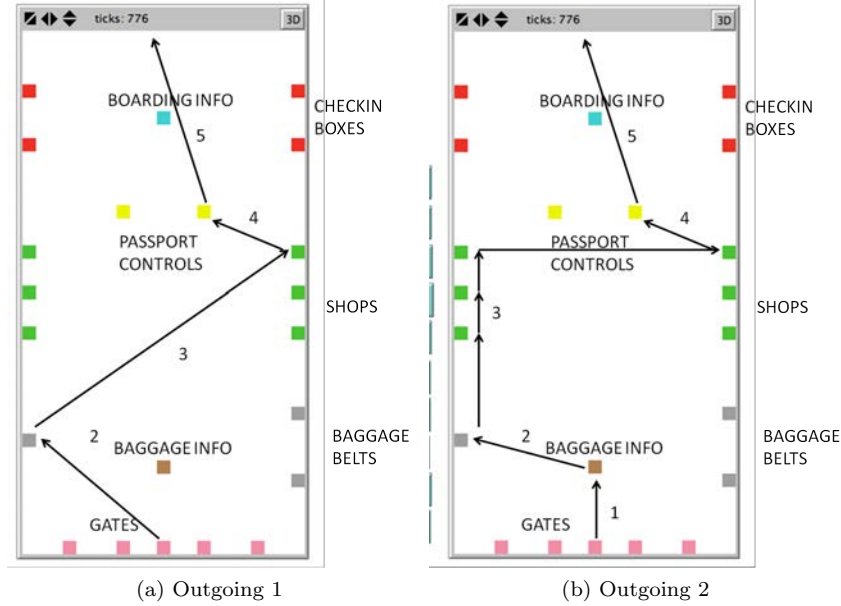(a) Outgoing 1   (b) Outgoing 2

Figure 3: Followed steps by outgoing agents with AmI

and the opposite. The first evaluation criteria consists in comparing satisfaction provided by the activities carried out by agents in the airport, although it is subjective we quantified it assigning satisfaction values to the next circumstances as follows:

- Avoiding missing the flight (high positive value).

- Shopping pleasure (low positive value).

- Time spent in rows (low negative value).

On the other hand, the second criteria is measured with the average time-spent in the airport.

Initial setup parameters of each simulation running are:

- Number of ingoing agents who do not use AmI.

- Number of ingoing agents who use AmI.

- Number of outgoing agents who do not use AmI.

- Number of outgoing agents who use AmI.

- Number of iterations required to avoid missing the flight.

- Number of passport controls.

14

- Number of checking boxes.

- Number of shops of different type.

- Number of boarding gates.

- Number of baggage belts.

Different values of these initial parameters would setup models of different types of (small and big) airports.

Further details of the implementation can be observed since the code can be downloaded at sourceforge:
`https://sourceforge.net/projects/netlogo-bdi-fipa-airport-model/`.

Additionally we have already uploaded our NetLogo model into the official NetLogo library at:

`http://ccl.northwestern.edu/netlogo/models/index.cgi`

4.1 Results

Figure 4 shows a caption of the NetLogo agent simulation that describes the elements that represent rooms of the airport: where Red points represent checkin boxes, Cyan point represents flight information panel, yellow points represent passport controls, black line represents a wall, green points represent shops, brown point represents jointly boarding information panel and baggage belts information panel, and finally pink points represent boarding gates.

The figure 4 generated by NetLogo show the definition of the initial parameter setup that would define the scale of each execution of the model, while 5 shows the relevant output variables that a NetLogo execution shows: total satisfaction of agents not using AmI, total satisfaction of agents using AmI, the average-time spent in the airport of AmI agents and average-time spent in the airport of nonAmI agents. These are the values we were trying to obtain in order to evaluate the benefits of using AmI in a context-aware scenario. Finally a curve of the evolution of both total satisfaction values can also be observed in 6. These curves show how satisfaction is very similar in the beginning of each simulation run, but as service rows increase and user agents miss their flights, satisfaction gets then reduced, but always in bigger amounts for non AmI agents. Values of average time also get worse when a very high number of agents are included in the simulation. So both evaluation criteria (time savings and agent satisfaction) show the potential benefits of using AmI when the number of agents is big enough.

According to, http://ccl.northwestern.edu/netlogo/docs/faq.html , the FAQ section of the official website, we have tried 30 runs with the shown initial parameter setup in NetLogo version 5 which runs models in a scientifically reproducible way. Table 1 shows about a 18% of time saving improvement, and about a 40% of more agent satisfaction for agents using AmI. Other simulations with different initial parameter setup fit approximately with these pattern while the same number of AmI and nonAmi agents (50% each) are participating in the simulation.
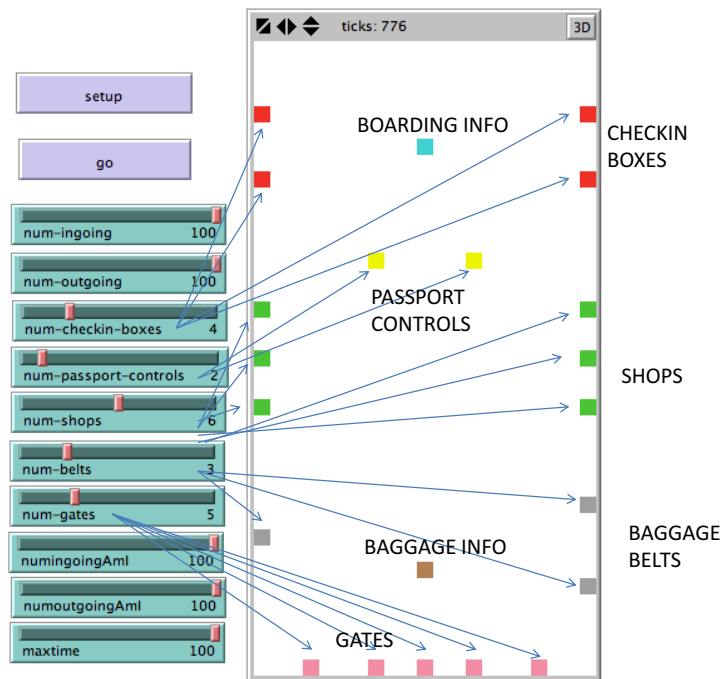
Figure 4: Description of elements in our NetLogo Model and initial parameter setup
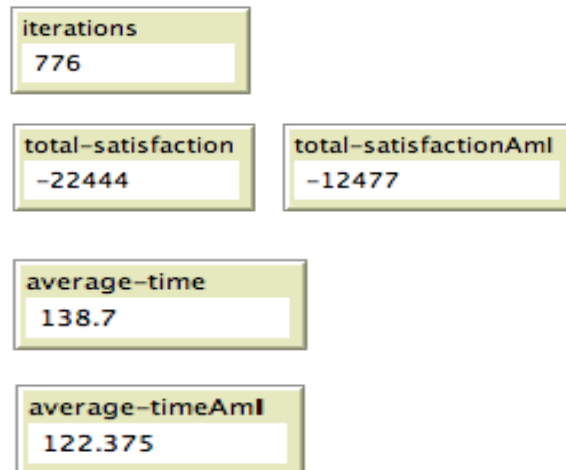

Figure 5: Final outcome of a NetLogo simulation of the airpot

Figure 6: Evolution of satisfaction values with and without AmI

Table 1: Results of 30 NetLogo executions of airport model

Name	Average	Standard Deviation
total-satisfaction	-21858,51	1415,67
total-satisfactionAmI	-13596,34	1937,87
average-time	145,31	7,26
average-timeAmI	122,87	4,93

5   Conclusions

In this contribution we looked forward to estimate the potential benefits of using an already defined by us AmI application of agents into an airport domain. Since we have previously suggested an agent architecture, an ontology and a 12-steps protocol to provide AmI services in such domain, we were interested in transforming such issues into a simulation that could easily visualize and compute such benefits when the number of agents is high enough. Although we had a JADE implementation of this model (Sánchez-Pi et al., 2008), we observed that a NetLogo model could achieve these goals. Since our initial proposal included FIPA messages and BDI reasoning agents, we used both NetLogo extensions to satisfy both requirements. We also by simplicity re-introduce an equivalent of our (small-sized) ontology into NetLogo instead of using an external already defined protege ontology. Although initial data in simulations are generated randomly, and the model is just an approximation of real-world airports, initial parameters allow the representation of both small and big airports through different values in number of boarding gates, shops, check-in boxes, baggage belts, etc. The definition of this case of use opens an interesting way to evaluate agent approaches dedicated to AmI, which is a significant contribution to the final development of AmI. In spite of the interaction complexity (12 steps protocol to provide services in AmI), we use a very limited number of options, so internal reasoning of agents is very straightforward, this is a limitation imposed by Netlogo simplicity. But this platform allows to test the consequences of using different interaction protocols when the number of involved agents is high, ignoring or simplifying the computational overhead that BDI reasoning and FIPA protocols impose over other alternative agent implementations.

Our proposal address the three most common shortcomings of AmI simulations according to (Serrano et al., 2014):

- simulation are closed, and can not be parameterized.

- experiments are not reproducible

- source code is rarely given

The showed simulation results help us to establish and quantify the potential benefits of using AmI. It also provides us with an estimate for an Airport scenario of where we should put the effort depending on if it is useful to use AmI for in this kind of scenario. Experiments and graphics resulting of computing multiple runs with equal numbers of each agent type, let us conclude about the posible relevance of counting on the facilities of AmI in this environment. Therefore, through a context-specific model we have measured the benefits of using AmI, this evaluation task is innovative, particularly because we did it without an oversimplification that would requiere removing BDI or FIPA in our model. As future work we want to include experiments with different agent system architectures, and very different population composition (proportion of AmI vs. NonAmI agents) of this airport scenario. Additionally we plan to characterize agents in a richer way, including features such as: excitement, anxiety,

urgency and fatigue jointly with types of agents different from ingoing/outgoing passengers: staff, tourists, business people, groups, etc. Finally, more complex airport maps based on real airports could increase the realism of simulations.

Acknowledgements

References

Foundations for intelligent phisical agents specification. Ginebra, Suiza, 1997.

Virtual Comunity Support In Telecare // Proc. of the fourth IFIP Working Conference on Virtual Enterprises (PRO-VE'03). 2003.

Implementation and Evaluation of a Location-Aware Wireless Multi-agent System // Proc. of Embedded and Ubiquitous Computing (EUC'04). 2004. 528–537.

Development of intelligent multi-sensor surveillance systems with agents // Robotics and Autonomous Systems. 2007. 55, 12. 892–903.

JADE. 2010.

Aarts E., Harwig E., Schuurmans M. Ambient Intelligence in Denning, J.(edt.): The Invisible Future. 2001.

Bajo J., Julian V., Corchado J.M., Carrascosa C., Paz Y. De, Botti V., Paz J.F. De. An Execution Time Planner for the ARTIS Agent Architecture // Journal of Engineering Applications of Artificial Intelligence. 2008. 21, 8.

Barton John J., Vijayaraghavan Vikram. UBIWISE, A Ubiquitous Wireless Infrastructure Simulation Environment // HP LABS. 2002.

Bordini Rafael H., Hbner Jomi F. BDI Agent Programming in AgentSpeak Using Jason // IN: PROCEEDINGS OF 6TH INTERNATIONAL WORKSHOP ON COMPUTATIONAL LOGIC IN MULTI-AGENT SYSTEMS (CLIMA VI). VOLUME 3900 OF LNCS. 2005. 143–164.

Brooks Rodney A. A Robust Layered Control System For a Mobile Robot. Cambridge, MA, USA: Massachusetts Institute of Technology, 1985.

Caballero Alberto, Botía Juan, Gómez-Skarmeta Antonio. Using Cognitive Agents in Social Simulations // Eng. Appl. Artif. Intell. X 2011. 24, 7. 1098–1109.

Corchado J.M., Bajo J., Paz Y. de, Tapia D.I. Intelligent environment for monitoring Alzheimer patients, agent technology for health care // Decision Support Systems. 2008. 44, 2.

Dawson R. J., Peppe R., Wang M. An agent based model for risk-based flood incident management // Natural Hazards. 2011. 59. 167–189.

Durfee E.H., Rosenschein J.S. Distributed problem solving and multi-agent systems: Comparisons and examples // 13th International Workshop on Distributed Artificial Intelligence. 1994. 94–104.

Fuentes Virginia, Carbó Javier, Molina José Manuel. Heterogeneous Domain Ontology for Location Based Information System in a Multi-agent Framework // IDEAL. 2006. 1199–1206.

Grninger M., Fox M. S. Methodology for the Design and Evaluation of Ontologies // International Joint Conference on Artificial Inteligence (IJCAI95), Workshop on Basic Ontological Issues in Knowledge Sharing. 1995.

Kovács G.L., Kopácsi S. Some Aspects of Ambient Intelligence // Acta Polytechnica Hungarica. 2006. 3, 1. 35–60.

Lech T.C., Wienhofen L.W.M. // Proc. of the Second European Workshop on Multi-Agent Systems. 2004.

Nishikawa Hiroshi, Yamamoto Shinya, Tamai Morihiko, Nishigaki Kouji, Kitani Tomoya, Shibata Naoki, Yasumoto Keiichi, Ito Minoru. UbiREAL: Realistic Smartspace Simulator for Systematic Testing. 2006. 459–476.

O'Hare G. M. P., O'Grady M. J., Kegan S., O'Kane D., Tynan R., Marsh D. Inteligent Agile Agents: Active Enablers for Ambient Intelligence // ACM's Special Interest Group on Computer-Human Interaction (SIGCHI), Ambient Intelligence for Scientific Discovery (AISD). Vienna, April 2004.

O'Neill E., Klepal M., Lewis D., O'Donnell T., O'Sullivan D., Pesch D. A testbed for evaluating human interaction with ubiquitous computing environments. 2005. 60–69.

Pan Xiaoshan, Han Charles S., Dauber Ken, Law Kincho H. A Multi-agent Based Framework for the Simulation of Human and Social Behaviors During Emergency Evacuations // AI Soc. X 2007. 22, 2. 113–132.

Pokahr Alexander, Braubach Lars, Lamersdorf Winfried. Jadex: Implementing a BDI-Infrastructure for JADE Agents // EXP - in search of innovation (Special Issue on JADE). 9 2003. 3, 3. 76–85.

Polhill J. Gary. Extracting OWL Ontologies from Agent-Based Models: A Netlogo Extension // Journal of Artificial Societies and Social Simulation. 2015. 18, 2. 15.

Poslad S., Laamanen H., Malaka R., Nick A., Buckle P., Zipf A. CRUMPET: Creation of User-friendly Mobile Services Personalized for Tourism // Proc. of the second International Conference on 3G Mobile. 2001.

Rao Anand S., Georgeff Michael P. BDI Agents: From Theory to Practice // ICMAS. 1995. 312–319.

Sakellariou Ilias, Kefalas Petros, Stamatopoulou Ioanna. Enhancing NetLogo to Simulate BDI Communicating Agents // Artificial Intelligence: Theories, Models and Applications. 5138. 2008. 263–275. (Lecture Notes in Computer Science).

JADE/LEAP Agents in an Aml Domain. // . 5271, 5271. 2008. 62–69. (LNAI).

Sanchez-Pi Nayat, Carbo Javier, Molina Jose Manuel. Analysis and Design of a Multi-Agent System Using Gaia Methodology in an Airport Case of Use // IBERAGENTS 2008. 7th Ibero-American Workshop in Multi-Agent Systems. In Conjunction with IBERAMIA'08. 2008.

Sánchez-Pi Nayat, Fuentes Virginia, Carbó Javier, Molina Jose Manuel. Knowledge-based system to define context in commercial applications // Proc. of the 8th ACIS Conference SNPD'07. Tsingtao, China, 2007. 694–699.

Schilit B., Adams N., Want R. Context-Aware Computing Applications // Proceedings of the 1994 First Workshop on Mobile Computing Systems and Applications. Washington, DC, USA: IEEE Computer Society, 1994. 85–90. (WMCSA '94).

Serrano Emilio, Poveda Geovanny, Garijo Mercedes. Towards a Holistic Framework for the Evaluation of Emergency Plans in Indoor Environments // Sensors. 2014. 14, 3. 4513–4535.

Sichman J.S., Conte R., Gilbert N. Multi-agent systems and agent-based simulation // 13th International Workshop on Distributed Artificial Intelligence. 1998. 94–104.

Wagner Neal, Agrawal Vikas. An agent-based simulation system for concert venue crowd evacuation modeling in the presence of a fire disaster // Expert Systems with Applications. 2014. 41, 6. 2807–2815.

Weiser M. The Computer of the 21st Century // Scientific American. 1991. 265, 3. 66–75.

Wooldridge M., Jennings N. Agent theories, architectures and languages: a survey // Lecture Notes in Artificial Intelligence. 890. 1995. 1–39.