A Framework for Distributed Interaction in Intelligent Environments

Dario Di Mauro[1], Juan C. Augusto[2], Antonio Origlia[1,3], and Francesco Cutugno[1]

[1] University of Naples "Federico II", Naples, Italy
[2] Middlesex University, London, UK
[3] University of Padua, Padua, Italy
{dario.dimauro,cutugno}@unina.it, j.augusto@mdx.ac.uk, antori@gmail.com

Abstract. Ubiquitous computing is extending its applications to an increasing number of domains. "Monolithic" approaches use centralised systems, controlling devices and users' requests. A different solution can be found in works proposing "distributed" intelligent devices that communicate, without a central reasoner, creating little communities to support the user. If the former approach uses all the available sensors being more easily context-aware, the latter is scalable and naturally supports multiple users.
In this work we introduce a model for a distributed network of entities in Intelligent Environments. Each node satisfies users' requests through Natural User Interfaces. If a node cannot produce the expected output, it communicates with others in the network, generating paths where the final target is undetermined and intermediate nodes do not understand the request; this is the focus of our work. The system learns parameters and connections in the initial topology. We tested the system in two scenarios. Our approach finds paths close to the optimum with reasonable connections.

1 Introduction

The human world is more and more interconnected. People have access to many computing devices and use them to communicate with each other. As a matter of fact technology is used to manage daily tasks more and more frequently and in a increasingly wider part of population: talk with friends, require information, enjoy art and manage appliances. Literature and market offer many solutions and, in order to maximise performances, they need to be contextualised in some domain, being more aware of how each system is changing the surrounding environment [2, p.1]. The approaches proposed in literature are mainly divided in two groups; the former is based on a central reasoner that collects all the data coming from the available sensors and takes actions considering a monolithic core. The latter approach uses a distributed network of interacting devices; they have their own sensors and reasoners and are focused on smaller targets, usually related on the set of categories they belong to. If the first case is more easily

aware of what is happening, the last is more flexible, and especially useful in dynamic contexts [25], where many users interact with the available technology together; moreover, the current market offers more and more devices focused on simple tasks, but that can be exploited to make the overall environment more intelligent. In all the adopted strategies, Human-Computer Interaction (hereafter HCI) plays a fundamental role. It is essential to make the users feel like *managers* of the environments they are in, moreover it influences the naturalness of the interaction with the system; it strictly depends on the domain, on the choices made concerning devices and design; however, considering interaction from a more abstract point of view, HCI, such as human-human and machine-machine communications, is performed by exchanging messages and data, also in a multimodal way; domain and tasks infer more specific details. From this point of view, designing and developing different kinds of interaction requires to choose different sets of modalities, input devices, multi-modal fusion approaches and input managers, but the main need does not change: strive for a better communication. This concept is stronger in an Intelligent Environment (hereafter IE) [3], because different types of interaction are needed and, in a pervasive system, each entity in the world interacts with others in a specific way. In order to design a new IE, the needed steps are: modelling the environment and developing best interaction modalities for each chosen task. Modelling is always necessary to manage domain-related possibilities, constraints and involved actors; it is usually managed by experts of that domain and, for IEs, they should provide a set of sensors, devices and network protocols to reach some goals [8]. The latter part, indeed, is time-consuming and repetitive in many cases. Moreover, multiple users are supposed to independently use the network at the same time, so the system should be powerful enough to manage a large number of dialogues together. In a distributed system the resulting organisation can be seen as partitioned in communities that depend on the current interaction, and users are *leaders* of each set. The considered scenario is typical in smart museums, where works might talk to users: visitors follow their path and interact with a few entities per time; meanwhile their interaction is independent from other users and the rest of works of art. A centralised system cannot support all the visitors at the same time, while a distributed system can, but it should interact with others system nodes to process unknown requests.

In this work we will present a model for a distributed network of devices and a framework for HCI in IE, focusing on Natural User Interfaces (NUI) [21] and with a specific preference for natural language dialogues supported by gestures and augmented reality approaches; this work is in the direction of basic, natural interaction design thought to improve user experience quality. The framework provides a communication layer between machines, offering a pervasive/ubiquitous infrastructure and a comfortable environment to support Internet Of Things (IoT) guidelines.

This paper is organised as follows: Section 2 reports works related to PHASER, that is the model presented in Section 3. In Section 4 we provide a solution for a problem related to distributed systems, named the “Navigation Problem”. From

Section 5 we focus on a Smart House, showing some representative scenarios and reporting Experiments and Results in Section 6. Section 7 concludes the paper.

2 Motivation and Related works

Our basic idea is to provide a distributed network of entities, where each node interacts with the user through multi-modal interaction following NUI guidelines. Knowledge is local to the node and limited to its own services. If the node is not able to produce the expected output for a request, it sends the received message to the others, without a prior determined target node. The Ambient Intelligence perceived by the users is built upon a collection of partial nodes' intelligence.

Other works facing the same problem have been found in literature. One of them is i*Chameleon [16], where the authors show an interaction among multiple devices, but the described system only supports a limited number of connected nodes. Moreover, the authors limit the work to exchanging signals, clearly defining starting (that sends requests and commands) and ending nodes (that receives and acts), but without considering a particular context, as differently proposed in this work. Peña-Ríos et al. [22] propose a framework to simply develop smart devices and test them by mixing real-life and simulated scenarios. That system presents interesting innovation points, but its use is limited to expert users only. An interesting system has been proposed by Dooley et al. [10]; it is a conceptual work, but it models ubiquitous computing in a smart environment aiming at supporting the user in IEs. The proposed model separates the world in spaces; each space is supposed to be an environment where people interact with distributed devices but their "reasoners" are still centralised. Considered spaces are homes, offices, local transports; different domains where technology can help people in their daily lives. Our model is complementary to Dooley's one, focusing on a distributed version of a single domain, supporting NUI.

A Multi-Agent System (MAS) point of view of an IE has been already explored in other studies [14, 12]. Loseto et al. [17] propose a flexible multi-agent approach for smart-environments. Their work is based on the discovery of semantic resource and orchestration, including negotiation techniques between user and smart devices. Sun et al. [23] present a multi-agent design framework for a smart-home and home automation applications. Their work is a Belief-Desire-Intention model [5] for agent individual behaviour design and a regulation policy-based method for multi-agent group behaviour design. Valero et al. [26] propose a system based on Magentix2; differently from others, the authors introduced multiple users' roles and access policies based on that. MAS paradigms are very valid solutions for smart-environment support, but the listed works are usually directly connected to the right device or use a central system that collects the provided services. A distributed context provides flexibility and adaptability [25], while a central "service manager" presents some limitations, especially in very popu-

lated environments; however MAS solution of Valero et al. [26] inherits benefits by well known models such as JADE[4] and Jason[5].

3 PHASER

In this section we will present our model and its implementation in a framework, called PHASER (Pervasive Human-centred Architecture for Smart Environmental Responsiveness). In our concept, PHASER gives a role to each entity that interacts with others. Possible entities are *objects* and *people* that interact with those objects as well. For this reason, we define a node in an abstract way to include the needs of both entities. Each node interacts with others, offering services and responding to requests. We define a single node as a tuple:

$$N(\iota, Cnf_\iota, ClosePeers_\iota, DiscoveredPeers_\iota, oBC_\iota)$$

where ι is a unique identifier of the node in the environment; $ClosePeers$ and $DiscoveredPeers$ are sets of related nodes in the environment: ι interacts with those nodes. Details later in this Section. oBC collects partial information about the connected peers, acting as Business Cards; they include information gathered from a shared ontology (explained later in this Section) and the accepted inputs.

A configuration Cnf determines the behaviour of ι in the environment. It comprises inputs, outputs and the behaviour towards other nodes. In details:

$$Cnf_\iota = (name_\iota, type_\iota, class_\iota, env_\iota, I_\iota, O_\iota, P_\iota) \tag{1}$$

where *type*, *class* and *env* determine the role of ι in the environment according an ontology, *name* is chosen by an interaction designer; details are in Section 3.4. I and O represent input and outputs respectively; they divide data into channels as in Equation 2 for multi-modal interaction, where c_x is a channel code and $RG_{c_x} = \{r_{i_1}, r_{i_2}, \ldots, r_{i_{c_x}}\}$ is a set of regular expressions. If N_{i_ι} and N_{o_ι} are the number of input and output channels, we define I_ι and O_ι in Equation 3.

$$Ch_j = \left(c_j, RG_{c_j}\right) \tag{2}$$

$$I_\iota / O_\iota = \bigcup_{1 \leq x \leq N_{i_\iota / o_\iota}} \{Ch_x\} \tag{3}$$

Eventually P_ι, briefly described here, is a set of parameters for each node that determines how it reacts to connections and interaction requests.

Connections. Each node can interact with other peers and their reference is stored in $ClosePeers$; they compose the initial topology designed by a domain expert. However, sometimes new unforeseen connections can be discovered and

[4] jade.tilab.com retrieved on December 2016
[5] jason.sourceforge.net retrieved on December 2016

included in $DiscoveredPeers$. New arcs may increase the power of the interaction for the user, because they create other bridges in the network, and if a node frequently connects to a discovered peer, this is automatically promoted to $ClosePeer$.

$$DiscoveredPeers = \bigcup(\kappa, c_\kappa, T_\kappa)$$

is a set of discovered nodes κ - we refer to them as "partial" connections -, where $c_\kappa \in [0..1]$ is the probability of making this connection fixed: κ will be included in $ClosePeers_\iota$ as $c_\kappa = 1$. T_κ is the last activity of this connection. As ι interacts with κ at time t, $c'_\kappa = UPC(c_\kappa, \Delta)$, where $\Delta = t - T_\kappa$. Details about the discovery of a node are in Section 4.2.

$$UPC(x, y) = x + \frac{x}{y\lambda(y)} \tag{4}$$

$$\lambda(x) = \phi_0 - (\phi_0 - \phi_1)\frac{1}{1 + e^{k_1 - x}} - (\phi_1 - \phi_2)\frac{1}{1 + e^{k_2 - x}} \tag{5}$$

$\phi_{0,1,2}$ and $k_{1,2}$ in Equation 5 are constants that derive on the activity of the interaction. According to the activity of the interaction, $\lambda(x)$ assigns a factor ϕ. $\phi > 0$ means that the connection is strengthened, but for $\phi < 0$ it is discouraged. Typical relations are $1 \geq \phi_0 > \phi_1 > \phi_2$, with $\phi_2 \leq 0$ and $k_1 \ll k_2$.

Open connection. As two nodes, ι and κ, open a connection, they share part of their local information composing a personal Business Card (BC):

$$BC_{\iota/\kappa} = \left(name_{\iota/\kappa}, type_{\iota/\kappa}, class_{\iota/\kappa}, env_{\iota/\kappa}, I_{\iota/\kappa}\right) \tag{6}$$

On open, $oBC_\iota = oBC_\iota \cup \{BC_\kappa\}$ and $oBC_\kappa = oBC_\kappa \cup \{BC_\iota\}$. On startup, ι asks a connection to each $p_i \in ClosePeers_\iota$. Partial connections will be opened following details shown in Section 4.2; in the latter case, collected BCs will be stored in the same way. Moreover, local information need to be updated:

$$DiscoveredPeers_\iota \cup \begin{cases} X_\kappa(t) & \text{if } (\kappa, c_\kappa, T_\kappa) \in DiscoveredPeers_\iota \\ \{(\kappa, 0, t)\} & otherwise \end{cases}$$

where $X_\kappa(t) = \{(\kappa, UPC(c_\kappa, t - T_\kappa), t)\} \setminus \{(\kappa, c_\kappa, T_\kappa)\}$ and t is the current instant. However, $X_\kappa(t')$ is added on every interaction between ι and κ, where t' are the considered next instants.

3.1 States

Each node in our model has a state. It determines which kind of work a node is doing and if it can accept other requests. A node starts in the *Idle* state; Table 1 summarise the states, while Figure 1 reports all the allowed transitions; details about Forwarding in Section 4. The connection *Managing machine interaction (MMI)* $\rightarrow$ *Idle* is a "forced" transition and it happens if node ι is waiting for node κ for a *MMI*. If ι remains in *MMI* it would not be able to receive a response from κ, because it would be recognised as busy.

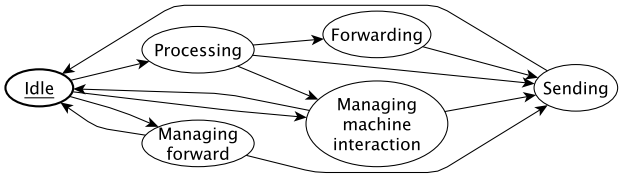

Fig. 1. Possible states and Transitions for a node in PHASER

Table 1. Description of states of PHASER

State	Description
Idle	initial state
Processing	the node is processing a request
Forwarding	the node is forwarding an unknown request
Managing Forward	the node is processing a received forward
Sending	the node is replying or re-forwarding a request
Managing machine interaction	two nodes are co-working for a request

3.2 Network of PHASERs

In order to better support the communication, two nodes ι and κ share their Business Card as seen before. The result is a network where each node has partial information about its local connections. This is a real distributed context because there are no entities that collect all the information. Users interact with each node in the environment, but the actual topology is hidden for them. They, indeed, perceive the network as a compact system because each node involves other parts as in a centralised system, but PHASER is more flexible than a centralised system.

During a communication, ι may send a request to κ. A request R is a snapshot of the input for ι represented by $R = \left\{(c_j, r_{c_j}), 1 \leq j \leq N_{i_\iota}\right\}$, where N_{i_ι} is the number of input channels for ι, c_j is a channel and r_{c_j} is the value of the request on c_j. κ receives R, but it is able to accept it just if it represents a valid input for κ; for an element in the request $R_x = (c_x, r_{c_x})$ this is true if $\exists (c_x, R_c) \in \{Ch_{x_\kappa} \subseteq I_\kappa \mid r_{c_x} \text{ matches on } R_c\}$. "$x$ matches on X" means that $\exists X_i \in X$ so that x complies on the format of X_i. This is wrapped in:

$$m\left(R_x, \kappa\right) = \begin{cases} 1 & \text{if } R_x \text{ is a valid input for } \kappa \\ 0 & otherwise \end{cases} \tag{7}$$

A request R is fully accepted by κ if $\sum_{1 \leq j \leq R} m\left(K_j, \kappa\right) = |R|$. We consider $|R|$ and not $|N_{i_\kappa}|$ because the request could not provide information for some channels. However, a fully accepted request is candidate to be manageable, but ι cannot take for granted that κ will process it successfully.

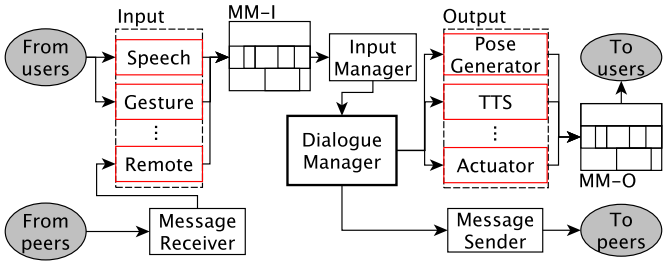

Fig. 2. The architecture of PHASER

3.3 Architecture

The model presented above has been implemented and Figure 2 presents the architecture of a node in the framework; it is an extended version of what already seen in [9]. The architecture supports multi-modal dialogues (manly speech+voice) and is designed as connected modules; it represent the skeleton of each node in the network of PHASER. MMI stores multi-modal input signals, in which a set of devices writes data in separated channels. Input data use timestamps gathered from the same hosting machine. *Input Devices* (IDs) implement an interface, in order to abstract from the used technology and represent data in an hardware-independent form. The diagram in Figure 2 shows examples of IDs and ODs; their actual activation depends on configuration discussed above. Similar reasoning is replicated on *Output Devices* (OD): MMO collects multi-modal output data coming from ODs and presents their fission to users. An *Input Manager* (IM) manages the fusion of data taken by the MMI structure and passes their classification to the *Dialogue Manager* (DM); this is, here, just an interface towards a real dialogue manager; its behaviour, in fact, highly depends on the particular node and cannot be included in the overall description. The DM is mainly based on OpenDial [15], including it as an external tool, but other DMs may be integrated in PHASER. PHASER offers to the Dialogue Managers a support for network interaction, and for processing of requests as explained in Section 4.

The *Remote* module, instead, is used to communicate with non-human peers by standard protocols. They can be robots, smart-devices, technologically enriched works of art, etc. Relationships among these entities create a PHASER network in which each node has an internal logic. By including the *Remote* module as ID, we are able to equally manage human - through active IDs/ODs - and artificial entities - through *Remote* -. This aspect improves the user-centred point of view, because connections are hidden and users perceive the world as a single block, where parts of it process each request. With the proposed architecture, PHASER offers a ubiquitous infrastructure and a comfortable framework for an Internet Of Things (IoT). The powerful aspect of this architecture is that its overall behaviour is not related to a single entity nor it is domain-specific but,

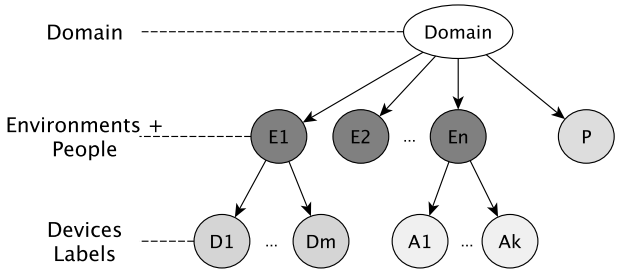

Fig. 3. The structure of a ontology accepted in PHASER

with proper I/O devices and a DM, it allows to easily prepare an IE, concentrating efforts on each entity. Furthermore, if an environment is considered as "entities offering services" and by them in the Business Card, each system will be able to opportunely contact nodes to solve internal tasks. This is a typical concept in AI agent-based approaches, but we are proposing it in a multi-domain - interaction-oriented - abstract architecture.

3.4 Ontology

PHASER uses an ontology that represents both the space and the devices. It currently relies on a new developed representation of an environment, but integration with well-known ontologies in this field is under investigation: we are considering soupa [7] and iot-lite[6]. An example of the ontology adopted in PHASER is in Figure 3. Its structure is as follows:

- a node represents the domain;
- a number of environments are connected to *domain*: they represent physical rooms in the environment;
- each room contains devices. Multiple labels are supported (here with different grey levels);
- particular nodes are directly connected to the domain, because they have not a semantic relation with a single environment: devices that move through the rooms, or personal devices are considered in this class;
- devices working in multiple rooms (i.e. lights) are connected to multiple environments. The current room, that depends on where the device is, must be specified in the configuration, otherwise a random room will be selected at run-time.

Each instance of PHASER is an entity which belongs to a leaf of the tree. By specifying a class, PHASER derives at run-time information such as the environment and other devices in the same room. The selected class, is a parameter reported in the configuration each node requires, as seen before.

[6] http://iot.ee.surrey.ac.uk/fiware/ontologies/iot-lite retrieved on December 2016

4 Navigation of a request

As a node interacts with a user, it tries to locally process the requests. If the node is not able to do it, the system could deliver an error message or share the request within the network. It may broadcast the data, being sure to reach at least one valid node, if it exists, but if multiple available nodes arrive, the starting node should be able to know which is the best one. Moreover, in large networks, many nodes that broadcast information may overload the network itself [11]. A second possible strategy is based on a more intelligent routing process [4], where the node can iteratively forward the request, and ontologies, history and context-awareness [1, 20] could help enriching system capabilities.

By relying on a common ontology and a dynamic topology as shown in Section 3, each node knows the business card of the adjacent ones. The current node could easily find out how much others can successfully process the request and who they are. The approach we propose to solve this problem is a depth-first-search in a distributed graph where a greedy part chooses the local best nodes as first. Considered parameters are: current request, past interactions and context-awareness.

As a greedy method on a distributed system, the current node that is not able to locally process a request, sorts its adjacent entities in decreasing order, comparing them with Equation 8.

$$Comp(s, c, n, R) = M(R, n) + Toll(s, c, n) + Friend(s, n) \tag{8}$$

where R the current request, in the form presented in Section 3.2, and s, c, n are respectively the starting, current and the next node in the path; the starting node is who received the user's request. The navigation ends if either a node provides a response or too many hops have been done. $M(R, n)$ is the match degree of the current request with the n's accepted inputs, where $M(R, n) = \sum_{0 \leq i < |R|} m(R_i, n)/|R|$. The higher $M(R, n)$ is, the more probably n can understand the request R. $M(R, n) = 1$ is a perfect match. R and m have both been presented in Section 3.2.

$Toll(s, c, n)$ represents a toll to pay in changing the environment. In our case:

$$Toll(s, c, n) = (-1)^{(E_c - E_n)(E_s - E_n)} \tau\left(E_c, E_n\right) \tag{9}$$

where E_A is an integer for the environment of node A, $E_A - E_B = 0$ *iff* nodes A and B are in the same environment and $\tau\left(x, y\right)$ is a function representing a toll going from x to y. Basically, the current node prefers to send the request in its environment, but if the request changes context, it is difficult to fall into the starting environment again.

If needed, $Friends$ assigns a bonus ϕ to requests coming from similar devices. Assuming that T_A is the type of device A in the ontology,

$$Friend(s, n) = \begin{cases} \phi & \text{if } T_s = T_n \\ 0 & otherwise \end{cases} \tag{10}$$

4.1 Sorting nodes

If a node x needs to send a request, it uses $Comp(s, x, n, R)$ in the Equation 8 to sort the connected nodes. Let s be the environment where the request started, x the current node, n one of the adjacent node and R the current request. Iteratively applying $Comp$ we obtain a sequence as in Equation 11; oBC_x has been introduces in Section 3.

$$Sorted_x = \left(p_1, p_2, \cdots, p_{|oBC_x|}\right) \tag{11}$$

where $Comp(s, x, p_i, R) \geq Comp(s, x, p_{i+1}, R)$, $\forall i \in [1..|oBC_x| - 1]$. x will forward R to the peers in the order of $Sorted_x$. At step i, peer p_i will be selected if p_{i-1} has failed at step $i - 1$; a final *"local fail"* is arisen by x if $p_{|oBC_x|}$ fails. In a centralised reasoner, the problem of forwarding is not relevant, because all the devices run in the same cluster. However, PHASER is flexible because the algorithm involves just active connections and it is recalled at each request.

4.2 Network adaptability

On forwarding, a node x selectively chooses the nodes that could reply to the user on the submitted request. x has no knowledge about the identity of the "target" t - a node that is able to process the request - *a priori*. However, PHASER optimises the navigation of the request in two ways: *(i)* by using the tolls, PHASER learns that some directions could provide a response, fostering paths that worked in the past; *(ii)* if the forward successfully reaches a target t, the topology changes because t is a discovered peer and s opens a partial connection towards it.

The new connection is added to $DiscoveredPeers_s$ shown in Section 3. As well as permanent links, s and t share their BC, adding them in oBC_s and oBC_t respectively. A discovered connection is a temporary link, and it lasts just for the current session; they are not affected in the sort process because they are directly chosen with a highest priority by the forwarding algorithm. In reaching a target node t, nodes within followed path do not open new connections, but they update the related tolls of a constant μ, limited to a maximum level of τ_{max}. Possible tolls towards new environments will be included. For their value we assign a constant τ_i. τ_{max}, μ and τ_i are empirically defined.

5 Case study in Smart house

Current technologies in the field of IE in Smart Houses rely on a single user interaction; this choice reduces the system complexity and optimises the ability of the system to understand users' needs according to their activities[24]. The most used communication device in this environment is the smart-phone. However, more than one person usually lives in a house and some studies have shown that people share the smart-phone [6, 13, 18]. On the contrary, recent solutions, such

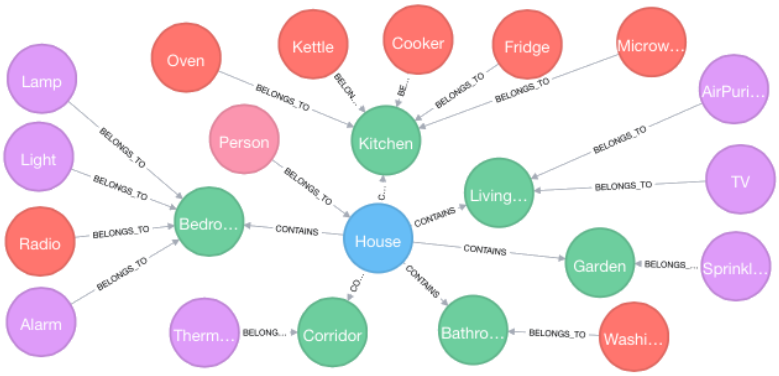

Fig. 4. An example of Ontology for Smart houses used in PHASER

as Amazon Alexa[7], support multiple users and foster the interaction through a completely shared device connected to everything and everyone. Although it is possible to use more than one smart device in the house, Alexa, the virtual "assistant", centrally controls all processes.

PHASER should be able to work in different domains just with a limited number of interventions on its configuration, however this paper is focused on the Smart House domain. Each device in the House gathers information about the context from an ontology, structured as in Section 3.4 and represented in Figure 4. Each node interacts with the others as seen in Section 3. The final goal of our approach is that each device propagates messages in the network as explained in Section 4. Users' perceives themselves as being able to control everything from everywhere, but each device is just expert of the actions it can process. Given this scenario the next challenge is making the interaction as more natural as possible; comments on this issue are reported in the end of Section 6. In the next Section, we report some relevant scenarios PHASER can tackle.

5.1 Scenarios

Scenario 1. *John is in the living room, watching TV is on and he is reading a book. Meanwhile, his wife is finishing the housekeeping, interacting with the washing machine and using the radio and their son is in the bedroom playing the guitar. The oven is working to prepare the dinner and John needs to check it, so he asks the TV if the oven is still cooking. The TV checks and answers "no, it has finished". Then the family can have dinner. Later on, it's time to go to bed, so John sets the alarm and falls asleep; the child does the same. The alarms can look the people in the house and can monitor how people are sleeping and, as it is almost time to wake-up, they switch the heating on in the proper bedrooms. John asks to boil the water in the kettle, then takes the breakfast and goes to work.*

[7] https://en.wikipedia.org/wiki/Amazon_Alexa retrieved on February 2017

Scenario 2. *It is Sunday and Mark is taking a shower; he will reach some friends later. He wants to mow the lawn, with the just bought automatic lawnmower, but it has not been programmed to start yet and, at least on the first time, he does not want to leave it working alone. So Mark asks some device there to start the lawnmower. The lawnmower starts.*

The first scenario is divided in two phases. Multiple users are interacting with the house; in the former part, they are operating independently, in the latter the devices collaborate to responds to users' needs. The second scenario, instead, presents the inclusion of a new device.

6 Experiment and Results

Single PHASER node interaction has been tested in Wizard-of-Oz modalities, where we tested internal dynamics reproducing recorded dialogues. We also conducted stress tests by simulating from one up to 80 input devices. Details are in [9]. In this Section we propose tests focused on network communication with the aim of proving that the algorithm proposed in Section 4 reaches the target node in a reasonable time and steps number. Another test, in simulation, uses a realistic network for a smart house. In all the presented results, the target is the node able to provide the desired output. This section ends presenting experiment and results with real users, meeting the scenarios presented in Section 5.1.

Simulation. In the considered case of study number of required nodes rarely goes beyond tens, and searching algorithms do not work in very challenging situations, where undetermined solutions are possible. Nevertheless, in order to test convergence features, quality and processing times of our algorithm, we generated networks with 10^3 devices. The network was divided in sub-networks with random arcs: connections within the same area are more probable than connections between different sub-networks. We generated networks with $2^{0..10}$ sub-networks with a number of nodes from 22 to 4134. By defining both the request and the target, we simulated an interaction, starting from a random node, using the algorithm of Section 4 as a measure of quality; μ, τ_{max}, τ_i and ϕ have been set to 0.1, 0.5, 0.3 and 0.05 respectively. Figure 5 shows the collected lengths of paths. The first and second columns contain the observed lengths at the first iteration and after 30 adaptation steps, also adding new relevant links; starting from a reference set of N nodes in a graph with $N^2 * 30\%$ connections in total, in repeated simulation, we added up to 7% of the connections. The third column shows the shortest path length on the same initial topology. In the last column we report a stability test; it is calculated by removing both important nodes in the network - after the adaptation - and the new introduced arcs.

Realistic case scenario. With the same approach, we generated a realistic network with more realistic connections distributions responding to a possible scenario in the House. We considered 3 bedrooms, 1 bathroom, 1 living room and

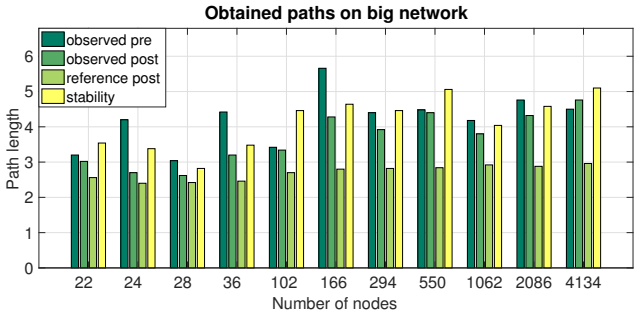

Fig. 5. Observed paths pre and post the adapting phase, comparing post with the shortest path and including stability check

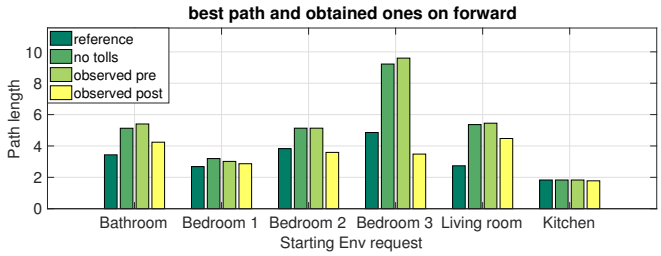

Fig. 6. A comparison in realistic situation with 28 devices and reasonable connections

1 kitchen with 28 nodes in total. The nodes were pseudo-randomly connected, as connections within the same area and limiting inter-rooms connections were preferred; as far as connections between rooms concerns, once again randomisation was chosen. The more connections the nodes share, the faster could be the response, because each node is more probably connected to the target. However, a full-connected graph is not always feasible because of infrastructure limits, especially with large networks; for this reason it is reasonable that nodes in the same sub-network, that could be *"semantically linked"* are highly connected because it is probable that they will interact. In this case, μ, τ_{max}, τ_i and ϕ have been set to 0.1, 0.5, 0.3 and 0.05 respectively.

The system was tested as follows: similarly as in previous tests, we choose the target node T. A random node R_1 from each room was picked and we calculated the shortest path between R_1 and T; used as reference in comparison with the forwarding algorithm without tolls, with $toll = 0.1$, and the adaptation process as explained in Section 4. In the second step, the resulting network is tested picking another node R_2 in the same area. The network was trained in seven iterations. Resulting data are showed in Figure 6. In order to be statistically relevant, each single experiment has been repeated 50 times; the whole process has been performed on 10 generated topologies having the same structure, but connections among nodes are potentially different.

Smart house. We conducted these experiments at the "Smart Space Lab" at Middlesex University, London. We proposed the scenarios seen in Section 5.1 for a network of 8 devices. Each user interacted with the devices through a web-page, where an image represents the associated intelligent device; an avatar has been used as a "personal assistant" for web-pages accessed through a smart-phone. All the nodes run on the same machine, but they can be remotely accessed. The tests have been performed by 10 people. They all perceived the system as a compact block, similarly to a centralised system, but volunteers, asked to express their preference for distributed vs. centralised systems, preferred a distributed system in 8 cases on 10.

Discussions In the first experiment, we noted that, after the learning phase, the system reached the request in fewer steps than *"observed pre"*, also in big networks. The *t-test* confirmed the hypothesis that "*post* data are lower than *pre* ones", also in the biggest topology, where the *post* average distance is slightly higher than *pre*. Moreover, the removed nodes affected the stability of the result, but the system was able to reach the target anyway. We measured that the reaches the target in 25 milliseconds, even on the biggest network. We just considered time required for the navigation of the request on the same machine; at run-time, average transmission delays and the Dialogue Manager of each node may take additional time.

It appears that our model works better when connections reflect a "semantic links" between two devices, because increases the possibility to use that connection. Two nodes will collaborate if they are semantically connected and if they offer "shared services", so these kind of connections should be fostered. Adopting random connections, preferring intra-area links, proved that our approach can work, improving the results after few steps. However, since an automatic optimised connection highly depends on the context and the offered services, the initial topology should be designed from an expert of the considered domain.

Interesting results came from the last experiments with real users. Although they perceived that they could control the whole house from each device, accomplishing our goal, they found unnatural talking with a specific device to manage everything - e.g. the fridge to switch the light on, etc.-. This aspect arose questions about which strategies are better to make the users feel the interaction as "natural". A possible strategy they advised is to elect a unit as manager for all the devices in each room. Alternatively, one can coordinate all the devices with the same interface, hiding a shared intelligence. Some studies [19] followed this way: this issue will be argument of future investigations for PHASER as well.

7 Conclusions

In this work we presented PHASER, a framework that manages HCI and Machine Machine interaction in Intelligent Environments. Against other approaches in this area, PHASER is based on a strongly distributed model, where each device carries on a dialogue with the user and the global intelligence is built upon

the collection of those capacities. Particular attention has been posed on the presented *Navigation Problem*, seen in Section 4; each node of the network forwards the received request to its connected entities if the local Dialogue Manager is not able to produce the expected output. The navigation of the request relies on a *toll*, a context-aware solution adopted to improve the path finding.

The actual use of the network trains the values of each toll: the system learns from real data to foster paths that gave good results in previous interactions. Moreover, the system adds unforeseen connections; they create bridges among different nodes in the network but they are activated just if their use is confirmed during the time. The system has been tested in three cases: we provided a simulated environment in order to test the model in extreme situations; a more realistic scenario has been considered: a case study in a smart house has been proposed with a reasonable number of devices and connections. In each cases, paths found by PHASER were close to the shortest path, without pre-processing the network.

PHASER relies on an external configuration that declares which inputs each node supports, the provided output, the dialogue manager and internal parameters that regulates the behaviour of each node. Currently a configuration is in XML format but a graphical tool is in work in progress to easily design the interaction in an comfortable way.

References

1. Abowd, G.D., Dey, A.K., Brown, P.J., Davies, N., Smith, M., Steggles, P.: Towards a better understanding of context and context-awareness. In: International Symposium on Handheld and Ubiquitous Computing. pp. 304–307. Springer (1999)
2. Alegre, U., Augusto, J.C., Clark, T.: Engineering context-aware systems and applications: a survey. Journal of Systems and Software 117, 55–83 (2016)
3. Augusto, J.C., Callaghan, V., Cook, D., Kameas, A., Satoh, I.: Intelligent environments: a manifesto. Human-Centric Computing and Information Sciences 3(1), 1–18 (2013)
4. Bello, O., Zeadally, S.: Intelligent device-to-device communication in the internet of things. IEEE Systems Journal 10(3), 1172–1182 (Sept 2016)
5. Bratman, M.: Intention, plans, and practical reason (1987)
6. Busse, B., Fuchs, M.: Prevalence of cell phone sharing. Survey Methods: Insights from the Field (SMIF) (2013)
7. Chen, H., Finin, T., Joshi, A.: The soupa ontology for pervasive computing. In: Ontologies for agents: Theory and experiences, pp. 233–258. Springer (2005)
8. Cook, D., Das, S.: Smart environments: Technology, protocols and applications, vol. 43. John Wiley & Sons (2004)
9. Di Mauro, D., Cutugno, F.: A framework for interaction design in intelligent environments. In: Intelligent Environments (IE), 2016 12th International Conference on. pp. 246–249. IEEE (2016)
10. Dooley, J., Henson, M., Callaghan, V., Hagras, H., Al-Ghazzawi, D., Malibari, A., Al-Haddad, M., Al-Ghamdi, A.: A formal model for space based ubiquitous computing. In: 7th International Conference on Intelligent Environments. pp. 74–79 (2011)

11. Dressler, F., Akan, O.B.: A survey on bio-inspired networking. Computer Networks 54(6), 881 – 900 (2010), new Network Paradigms
12. Ferrando, S.P., Onaindia, E.: Context-aware multi-agent planning in intelligent environments. Information Sciences 227, 22 – 42 (2013), http://www.sciencedirect.com/science/article/pii/S0020025512007748
13. Karlson, A.K., Brush, A., Schechter, S.: Can i borrow your phone?: understanding concerns when sharing mobile phones. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. pp. 1647–1650. ACM (2009)
14. Li, W., Logenthiran, T., Woo, W.L., Phan, V.T., Srinivasan, D.: Implementation of demand side management of a smart home using multi-agent system. In: 2016 IEEE Congress on Evolutionary Computation (CEC). pp. 2028–2035 (July 2016)
15. Lison, P.: A hybrid approach to dialogue management based on probabilistic rules. Computer Speech & Language 34(1), 232–255 (2015)
16. Lo, K.W., Tang, W.W., Leong, H.V., Chan, A., Chan, S., Ngai, G.: i*chameleon: A unified web service framework for integrating multimodal interaction devices. In: Pervasive Computing and Communications Workshops (PERCOM Workshops). pp. 106–111. IEEE (2012)
17. Loseto, G., Scioscia, F., Ruta, M., Di Sciascio, E.: Semantic-based smart homes: a multi-agent approach. In: 13th Workshop on objects and agents (WOA 2012). vol. 892, pp. 49–55 (2012)
18. Matthews, T., Liao, K., Turner, A., Berkovich, M., Reeder, R., Consolvo, S.: "she'll just grab any device that's closer": A study of everyday device & account sharing in households. In: Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems. pp. 5921–5932. CHI '16, ACM (2016), http://doi.acm.org/10.1145/2858036.2858051
19. Mennicken, S., Zihler, O., Juldaschewa, F., Molnar, V., Aggeler, D., Huang, E.M.: "it's like living with a friendly stranger": Perceptions of personality traits in a smart home. In: Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing. pp. 120–131. UbiComp '16, ACM (2016), http://doi.acm.org/10.1145/2971648.2971757
20. Musolesi, M., Mascolo, C.: Evaluating context information predictability for autonomic communication. In: Proceedings of the 2006 International Symposium on on World of Wireless, Mobile and Multimedia Networks. pp. 495–499. IEEE Computer Society (2006)
21. O'hara, K., Harper, R., Mentis, H., Sellen, A., Taylor, A.: On the naturalness of touchless: Putting the "interaction" back into nui. ACM Transactions on Computer-Human Interaction (TOCHI) 20, 5 (2013)
22. Peña-Ríos, A., Callaghan, V., Gardner, M., Alhaddad, M.J.: Using mixed-reality to develop smart environments. In: Intelligent Environments (IE). pp. 182–189. IEEE (2014)
23. Sun, Q., Yu, W., Kochurov, N., Hao, Q., Hu, F.: A multi-agent-based intelligent sensor and actuator network design for smart house and home automation. Journal of Sensor and Actuator Networks 2(3), 557–588 (2013)
24. Toch, E., Wang, Y., Cranor, L.F.: Personalization and privacy: a survey of privacy risks and remedies in personalization-based systems. User Modeling and User-Adapted Interaction 22(1-2), 203–220 (2012)
25. del Val, E., Rebollo, M., Botti, V.: Combination of self-organization mechanisms to enhance service discovery in open systems. Information Sciences pp. 138–162 (2014)
26. Valero, S., del Val, E., Alemany, J., Botti, V.: Enhancing smart-home environments using magentix2. Journal of Applied Logic (2016)