A Proposal for Realizing Adaptive Ambient Ecologies through an
ontology-based Multi-Agent System

Tobias Heinroth, Achilles Kameas, Christian Wagner, and Yacine Bellik

Abstract—This paper presents an agent-based approach for realizing a new generation of intelligent environments referred to as adaptive ambient ecologies. These are highly distributed systems, which require new ways of communication and collaboration in order to support the realization of people’s tasks. We use three types of agents for the main functions of planning, adaptation and interaction. The knowledge repository of an ambient ecology is encoded in an ontology that is assembled on demand and is made accessible to the agents. In our approach, we introduce methodologies for extending the concept of ontologies beyond the storage of structured data to further include mechanisms for the exchange and manipulation of activity related information within ecologies.

I. INTRODUCTION

INTELLIGENT environments (IEs) are (usually closed) spaces equipped with a variety of technical components ranging from flat-screen monitors, centrally controlled lights, radio receivers to HVAC systems and automated window blinds. The strong heterogeneity of these components makes their configuration and maintenance as part of an integrated system a non-trivial task. Within the European Commission funded project ATRACO (Adaptive and Trusted Ambient eCOlogies) we try to lay the foundation for the development of next generation IEs, so-called Ambient Ecologies (AEs) [1]. An ambient ecology is a set of devices located in close proximity (i.e. in the same space) and services that can communicate, collaborate, and interact with each other, the environment and most importantly, the people inhabiting the IE.

At the basis of our approach is the assumption that the end-user(s) want(s) to fulfil aims or goals such as feel comfortable, go shopping or prepare dinner within their AEs. For this reason we use the notion of activity spheres describing the aim-dedicated conglomeration of entities and the collective knowledge surrounding the user living in an ecology. The formation of such activity spheres is supported by a service-oriented ambient ecology architecture, which includes APIs to interface with existing hardware modules and communication protocols, ontologies and ontology management modules, decision making mechanisms, planning modules, negotiation and learning mechanisms, intelligent agents, trust policies and privacy enforcement mechanisms and compose-able interaction components, all necessary for the formation of task-based transient ubiquitous computing applications.

Compared to previous research, we adopt a unique standpoint in that we assume that (a) various IEs are available, each of them hosting an AE and (b) devices and services of an AE (1) are inherently heterogeneous and (2) as such contain heterogeneous descriptions of their capabilities and services that can only be accessed from but not modified by other components. Thus, in order to achieve task-based collaboration amongst them, one has to deal with this heterogeneity, while at the same time achieving independence between a task description and its respective realization within a specific AE. To this end, we employ ontology alignment mechanisms, which are described in more detail in Section IV. Moreover, the system aims to establish only a transient collaboration between members of an AE in order to support a specific user activity in changing context. In order to achieve this, we use a set of intelligent agents to support adaptive planning, task realisation and enhanced human-machine interaction.

In this paper, we shall focus on the realization of the ontology and the agents that support adaptive realization of user tasks within changing heterogeneous AEs.

The remainder of this paper is organised as follows. In Section III, the related work is presented. The conceptual architectural design is introduced in Section III and the core software modules Sphere Manager and Ontology Manager are presented. Section IV describes the agents which are necessary for the formation of the ecology. Section V focuses on the realization of ambient ecologies concerning information storage, exchange and manipulation through the use of ontologies. An example realization of a task within an AE is used throughout the paper to better illustrate our concepts. Finally we draw conclusions and present some aspects of our future work in Section VI.

II. RELATED WORK

Ontology based pervasive systems have already been presented in the literature. Among these:
- the CADO (Context-aware Applications with Distributed Ontologies) framework [2] relies on distributed ontologies that are shared and managed in a peer-to-peer fashion, so as to ensure semantic interoperability via the process of ontology merging;

- CoBrA (Context Broker Architecture), is a pervasive context-aware computing infrastructure, that provides knowledge sharing, context reasoning, and privacy protection support for ubiquitous context-aware systems, using a collection of ontologies, called COBRA-ONT, for modelling the context in an intelligent meeting room environment. These ontologies expressed in the Web Ontology Language (OWL), define typical concepts associated with places, agents, and events and are mapped to the emerging consensus ontologies that are relevant to the development of smart spaces [3];

- GAIA is an infrastructure for Smart Spaces, which are ubiquitous computing environments that encompass physical spaces [4]. Ontologies are introduced in this system, as an efficient way to manage the diversity and complexity of describing resources, that is, devices and services. Therefore, these ontologies are beneficial for semantic discovery, matchmaking, interoperability between entities, and interaction between human users and computers.

All these ontology-based systems use static heavyweight domain ontologies to support ubiquitous computing applications. These ontologies are used to represent, manipulate, program and reason with context data and they aim to solve particular ubiquitous computing problems, such as policy management, context representation, service modelling and composition, people description, location modelling, etc. However, in the ubiquitous computing
domain, it is difficult for applications to share changing context information, as they will have to constantly adapt to the changes.

There have also been proposed agent-oriented architectures for pervasive systems. For example, project ASK-IT supports agent applications on a service-oriented architecture, using agents as brokers between the needs of the users and the server-side services [http://www.ask-it.org]. Its successor, project OASIS, has adopted a full fledged agent system, which consists of user agents, service prioritisation agents, service provider agents and content connector agents; a hyperontology is also used to deal with heterogeneity [5]. This architecture is close to ATRACO, but it approaches the problem of adaptation from a different perspective – that of global co-ordination and optimization of matching user requests with systems services based on user preferences.

On the contrary, in ATRACO we propose an ontology-based adaptation approach, which can be tailored to the task description, applied by intelligent agents that realize task execution, and maintained dynamically by re-calculating on-demand the interfaces between the constituent local ontologies.

In modelling ATRACO functionality, we adopt an approach similar to the one described in project DynAMITE, but we are proposing a distributed task-oriented resource management and interaction scheme [6].

III. ARCHITECTURE

In its traditional interpretation, the concept of IEs refers mainly to the resources of a space and the services made available to users or devices within it. We extend this notion to define Ambient Ecologies, whereby we assume that all IE resources (i.e. devices, services, data, etc.) are self-described, accessible and have the capability for communication. With the support of an AE, user tasks can be realized. We introduce the notion of Activity Sphere (AS) to model the set of resources and the software modules necessary to support the realization of specific user goals that can be expressed as inter-dependent tasks. Aim-related activity spheres can represent the adaptive realization of a user’s task in a changing environment concerning states, events and all involved entities.

An AS is both the virtual description of the resources (e.g., devices, services, agents, and user) required to achieve a user’s single aim and its fulfilment in the context of a specific ecology. Moreover, one AS may be owned by multiple users, multiple AS may exist within the same AE, while the same resource may be used in different AS. Thus, the realization of AS based on AEs requires a new architectural approach discussed in this section. In our approach, an AS usually resides in an AE and in general consists of ontologies as primary knowledge and information repository, the user itself as the task owner (and possibly the environment’s occupant) and further entities, which can be categorised as passive and active.

Throughout this paper, we shall use the example of an AS that supports the user’s goal “Feel comfortable after work”. Fig. 1 shows a snapshot of the AS that supports it. The AS is everything inside the main circle cloud. The sphere consists of:

1) Ontologies:
   - The aim ontology acts as a description of the user’s aim defined as a set of goals each modelled within a task model (i.e., the domain).
   - Local ontologies are provided by both active and passive entities and encode their state, properties, capabilities and services.
   - The sphere ontology is an information pool which is constantly evolving and being updated. It serves as the core of the sphere. Different ways of forming this ontology are possible: Aligning, merging and mapping (see Section V) of entities’ local ontologies. The two latter possibilities result in a new ontology. Aligning forms a transient quasi-ontology which connects its constituting ontologies thorough a network of reconfigurable pointers.

2) Passive entities:
   - Devices such as interaction devices (touch screens, speakers, microphones, etc.), actuators and sensors including televisions, radio receivers and HVACs, etc.
   - Services such as remote or external web-services (e.g., online banking) and local or internal services (e.g., personal calendar).

3) Active entities:
   - Agents (Fuzzy Task Agent, Interaction Agent and Planning Agent) are responsible for automated adaptation, resolving conflicts, interacting with the user, establishing plans and in general realizing the concrete tasks in the task model.
   - The ontology manager is responsible for keeping the sphere ontology up-to-date and for handling events within the sphere.
   - The sphere manager is responsible for the formation, the integration into and releasing from activity spheres of passive entities as well as the spheres’ dissolving.

4) The user who as the occupant of the IE is at the centre of each activity sphere.

The high-level view of the AS in Fig. 1 shows how the realization of the aim “Feel comfortable after work” can be supported. The sphere consists of a defined set of passive and active entities, which participate in the different tasks that realize the sphere goals. Each goal is decomposed into a hierarchy of abstract tasks, where the description of an
abstract task does not depend on the AE at hand. Based on this abstract description and the descriptions of the resources of a specific AE, the Planning Agent derives the descriptions of concrete tasks. Once the concrete tasks are computed, they are mapped to the various entities in the AE. The decomposition to concrete tasks and the mapping of entities may initially be computed offline. As soon as the mapping is complete, each active entity independently tries to realize its function within the task by using passive entities or interacting with other active entities.

The user and both passive and active entities except the sphere and the ontology managers are maintaining their own local ontology which stores their state, properties, capabilities and services. Thus, all data concerning an agent, a device, a service or a user is stored in their respective local ontologies. Fig. 1 shows the local ontologies as small ellipses. The lights, for example, may maintain the following values within their local ontology:
- Watts (W)
- Amperage (A)
- Durability (h)
- Number of bulbs (integer)
- Actual state (Boolean [on | off] or as linguistic variable [bright | dim | dark])

The user’s local ontology is used as his profile. All active entities can edit and add-in values regarding the user to this ontology. Such values can be, for example, private information, the user’s abilities and system-learned user-behaviours or preferences.

The architecture presented in this section is based on the principle of independence between AE and AS; thus an AS may be realized in different AEs (in the general case, this includes an AE changing its configuration because, for example, a device has been switched off, or a new device has arrived). Of course, this will require adaptation of the AS, as two AEs cannot be expected to be identical. In our architecture, we use three types of agents to achieve adaptation: a Fuzzy Task Agent (FTA), a Planning Agent (PA) and an Interaction Agent (IA) (see Section IV).

The PA re-computes the mapping from abstract to concrete tasks when the AE changes. The FTA controls directly the AE resources that have been mapped to a specific task by implementing adaptive Fuzzy Logic based control methods. The IA intervenes when a decision on adaptation or task concretization cannot be reached, or when the user wishes to express an explicit request to the system.

A. Sphere manager

The sphere manager (SM) module is responsible for setting up, managing and dissolving an activity sphere. It does so based on a goal description and on information about the entities of the AE. SM then initializes the PA and passes it the abstract task description. Subsequently, based on the concrete task plan, it initializes the Ontology Manager, which composes the sphere ontology.

The SM monitors the AE and “listens” to changes in its composition. When such a change is detected (i.e., the SM gets notified that a new device has entered, or some device cannot be discovered any longer), it triggers the PA in order to re-compute the concrete task plan, which is again passed to the Ontology Manager.

Finally, the SM dissolves the sphere, once all tasks have been accomplished and the goal attained, by asking the agents to “shutdown” and relinquish the resources they’ve been using, and by asking the Ontology Manager to delete the ontology alignments.

B. Ontology manager

The Ontology Manager (OM) module creates and manages the sphere ontology. It uses ontology alignment algorithms to uncover alignment points between the constituent ontologies, which it maintains until the sphere is dissolved. The OM is able to answer queries in respect to the knowledge stored in the ontology by using the alignments to access the constituent ontologies, which in turn contain the requested information.

IV. THE MULTI-AGENT SYSTEM

Agents are the most important active entities within the activity sphere, as they directly affect and manipulate the ecology surrounding the user. Three different types of agents are necessary for the formation of an ambient ecology: Planning Agents, Fuzzy Task Agents and Interaction Agents.

A. Planning Agent

The Planning Agent is able to solve combinatorial problems that have a relative small number of consistent plans (solutions) with respect to all possible plans. The PA’s aim is to resolve all goals into concrete tasks given a specific ecology [7], [8]. To start the planning process some initial information (such as an abstract task model and a problem description) is already provided regarding the ambient ecology, but some information may also has to be retrieved initially by communicating with the user through the Interaction Agent.

During the reasoning process and during the evolution of the sphere, a situation may occur, where the PA needs further information that is either needed to complete the planning or to direct search towards the solutions that the user prefers. Direct user-machine interaction will then be initiated by involving the Interaction Agent in those situations.

B. Fuzzy Task Agent

The Fuzzy Task Agent (FTA) is responsible for the direct control of any entities within the current ecology. More specifically, an individual FTA is instantiated by the sphere manager for the realization of each concrete task. During instantiation, the FTA is associated with the input entities (e.g. sensors) and output entities (e.g. actuators) which have been identified by the PA in order to realize the current task.
If a specific task has previously occurred, the FTA will incorporate any rule sets and fuzzy set definitions which have previously been defined and stored in local ontologies of the respective involved entities (e.g. the user).

If the task is considered novel, the FTA will start monitoring the entities it has been associated with for a specified amount of time, recording the combination of their individual states as they change within the ecology (e.g. heater setting modified by user, temperature sensor, etc.).

After the monitoring phase is complete, the recorded set of input-output combinations is evaluated and a number of fuzzy membership functions are created which model the individual states of the entities [9]. The number of fuzzy membership functions as well as the linguistic labels for each of these functions is retrieved from the sphere ontology. A heater for example could be represented by three fuzzy logic sets labelled “cold”, “medium” and “warm”.

After the fuzzy sets have been created, a rulebase, which incorporates the different input-output mappings from the gathered data and applies them to the fuzzy sets, is generated according to the principles outlined in [4]. The rules within the rulebase have the following format which is standard in Fuzzy Logic Control:

\[
R_r: \text{IF } x_1 \text{ is } \tilde{F}_1 \text{ AND } \ldots \text{ AND } x_p \text{ is } \tilde{F}_p \\text{ THEN } y_1 \text{ is } \tilde{G}_1, \ldots, y_q \text{ is } \tilde{G}_q\]

\[r \in \{1, \ldots, S\}\]

where \(\tilde{F}_1\) to \(\tilde{F}_p\) are fuzzy input sets, \(\tilde{G}_1\) to \(\tilde{G}_q\) are fuzzy output sets, \(P\) is the number of FLC inputs, \(Q\) the number of FLC outputs and \(S\) is the number of rules in the rule base. After the initial setup phase is complete, the FTA starts controlling the entities within the ecology in order to fulfill its current task (e.g. “Heat the room to a comfortable level”). As the characteristics of the entities are subject to a high level of uncertainty (e.g. sensors give inaccurate readings, actuator performance changes over time, etc.) and as the user preferences (e.g. what temperature is “comfortable”? ) are subject to change - which can be seen as another source of uncertainty, we are applying a lifelong learning/adaptation approach, based on the concepts detailed by Doctor et al. in [10].

The sphere adaptation is with the help of the FTA, realised on two levels. First, the actual rules which can be created, deleted or modified to reflect changes within the ecology or the user preferences. Second, the actual fuzzy sets which represent the linguistic variables are modified to reflect changes in the perception of the user of these variables (e.g. if a user changes his concept of “cold”, the fuzzy set should change to reflect this). We are aiming to use the strong uncertainty modelling capabilities of type-2 fuzzy logic sets [11] in order to model and adapt the meaning of the linguistic variables as accurately as possible while maintaining a small number of rules, which in turn should improve performance and facilitate human interpretation and as such system accountability.

During its operation, the FTA continually monitors the entities which are associated with its inputs and controls the entities associated with its outputs according to its rulebase which is constantly adapted. A schematic view of this process can be seen in Fig. 2. As the FTA adapts, it exposes the modified fuzzy set definitions and the rulebase through its ontology to the sphere ontology. This allows for other components of the system to react to new information encoded within the FTA’s rules as well as maintaining an up-to-date view of the definition of specific linguistic variables.

**C. Interaction Agent**

The Interaction Agent (IA) is responsible for the direct interaction between the user and the system within the current ecology. It is in charge of the interpretation of the messages produced by the user and the presentation of messages produced by the system (through the PA). Different communication modalities may be used, either by the user or the IA, to express messages (speech, touch buttons, text, graphics, avatars, etc.). The IA has to determine the best means of interaction (modalities and devices) [12] according to the current ecology. Similar to the FTA, an individual IA is instantiated by the sphere manager for the realization of each concrete task. During instantiation, the IA can access the following elements to achieve its mission:

- Available input and output interaction devices such as speakers, microphones, touch screens, etc.
Available input and output interaction modalities (e.g., text, graphic, touch, speech): the list of available interaction devices is not enough for the IA to achieve its mission because an interaction device may support different interaction modalities. For instance, a speaker can support different modalities, such as earcon (sound pattern used to represent a specific item), speech synthesis, music, etc. The presence of a speaker in the list of available interaction devices doesn't necessarily mean that the speech synthesis modality is also available. As such, the IA needs to be aware of what interaction modalities are available within the current ecology.

Available input and output modes: modes refer to the human sensory-motor systems such as vision and audition. The IA needs to know which modes are available and/or preferred at a given moment to determine the best modalities to use. This availability/unavailability and preference may be temporary (for instance, the user is currently closing his eyes) or permanent (the user is blind). Such information may be retrieved from the user profile, and it is important to not select an interaction modality which is not preferred by the user because it may not be perceived by him.

Physical environment characteristics: the characteristics of the physical environment (such as ambient light level, ambient sound level, etc.) may influence the interaction process (for instance, a sound modality will be inadvisable if the ambient sound level is loud). The IA needs this information to select the best modalities to use.

Message characteristics: the characteristics of the interaction message (such as its semantic content, its importance, its emergency state, etc.) may also influence the way the message will be expressed. For instance, redundant modalities (speech synthesis and text display) may be used to express an important message.

Based on all these elements, the IA will use a behavioural model, which identifies the best interaction components (modalities and devices) adapted to the current ecology. Its formalization can be made in different ways: rules [13], automats [14], Petri networks [15], etc.

V. REALIZATION OF AMBIENT ECOCOLOGIES

The main concept of our approach is that we use ontologies not only for storing and classifying data but also for the communicative exchange of data and knowledge within a sphere. The sphere ontology is used as an intermediate layer between active and passive entities of a sphere. As will be described in this section, passive entities contribute their state to the ontology, while active entities may also change it.

A. Role of the ontology

An ontology is usually defined as “a formal, explicit specification of a shared conceptualization” [16]. A “conceptualization” refers to an abstract model of some phenomenon in the world, which identifies the relevant concepts of that phenomenon. “Explicit” means that the type of concepts used and the constraints on their use are explicitly defined. “Formal” refers to the fact that the ontology should be machine readable. “Shared” reflects the notion that an ontology captures consensual knowledge, that is, it is not private of some individual, but accepted by a group. Thus, an ontology is a structure of knowledge, used as a means of knowledge sharing within a community of heterogeneous entities.

In our approach, we assume that each AE entity (user, agent, device or service) maintains locally an ontology (could only be a protocol header or a simple description), where its properties, capabilities and current state are described. This local ontology is managed only by the owning entity. In other words, the local ontology of an entity represents the complete set of knowledge associated with this entity.

Based on the concrete task plan, which details the entities that must be used in order to realize an abstract task, the ontology manager forms the sphere ontology, which contains information about:

- the states of the devices and services that participate in the task;
- the knowledge bases of the AS agents;
- the user profile;
- the constraints and policies that apply to the realization of the goal and its tasks.

The sphere ontology is formed by accessing the local ontologies of these entities and attempting to interface them with the help of ontology matching algorithms. Ontology matching is the process of finding relationships or correspondences between entities of two different ontologies. Its output is a set of correspondences between two ontologies, that is, relations holding, or supposed to hold, between entities of different ontologies, according to a particular algorithm, or individual. This set of correspondences is called an alignment. According to [17], the ontology alignment process is described as: given two ontologies, each describing a set of discrete entities (which can be classes, properties, rules, predicates, or even formulas), find the correspondences, e.g. equivalences or subsumptions, holding between these entities.

Based on these alignments, one can apply ontology merging in order to produce the top level, sphere ontology, which realizes an activity sphere [18]. When performing ontology merging, a new ontology is created, which is the union of the source ontologies. The new ontology, in general, replaces the original source ontologies. The challenge in ontology merging is to ensure that all correspondences and differences between the ontologies are reflected in the merged ontology. However, ontology
merging cannot be directly applied in our system, although it would speed up the knowledge management within a sphere, because the constituent (local) ontologies will continue to exist and change, thus the sphere ontology will require continuous re-engineering.

B. Functional description

The main function provided by the sphere ontology is information sharing amongst entities populating the sphere. Apart from the passive entities which simply provide a local ontology and keep it synchronized with their actual state, agents play a more sophisticated role within the AS: agents can read and write data encoded in all the ontologies they are aligned to. To realize, for example, the task “Heat the room to a comfortable level” the PA initially may identify the following agents and corresponding passive entities which are necessary for this task:

- User (passive);
- Radiator (passive);
- Window (passive);
- FTA (active);
- IA (active).

The OM is now able to align these local ontologies and build up a common knowledge base. The FTA might find information about the preferred temperature in the user’s ontology, which can be used as a starting point for learning. It can also directly edit the values in the local ontologies of the radiator and the window to change their state. Due to the alignment of the FTA and the IA, it is also possible for the IA to react to a direct user input about temperature by triggering the FTA to change the radiator’s or the window’s state. Whenever a structural change happens (i.e., a new device is discovered by a device registration service), the OM will do a realignment to update the sphere ontology and triggers the agents to include the new elements they are aligned to in deducing their behaviour. If a device changes its state the OM does not have to realign; it can either throw events to trigger the agents to read a new value or the agents themselves can constantly read values of entities they are aligned to in order to react adequately.

We can use the same paradigm to demonstrate how a complete goal and not only one single task is fulfilled. Since a sphere is directly linked to a specific goal, it is possible to align, map or merge different ontologies in respect to several tasks. Thus, the communication within the sphere depends on the current activity and for this reason only relevant data will be transferred. We divide the members of the ecology into active and passive for the following reason: all passive entities can use their local ontology only to provide information, i.e., they behave passive.

By contrast agents can also express the requirement for additional information not contained in their individual local ontology, i.e., they behave active. If the ecology provides knowledge related to the agent’s requirement, the ontology manager aligns this knowledge with the local ontology of the agent and by doing so enables it to act in a more intelligent way. The managers within the ecology are by definition active, as they throw events, form structures (i.e., ontologies) and align, map or merge data provided in local ontologies.

C. Example

The following short scenario should help to understand how we want to use our concept for adaptation:

Suki has been living in this new apartment for the past 10 months. His is no ordinary house; it is not a commonplace first generation smart house: it is a brand new adaptive house! When he comes home after work he wants to feel comfortable and for this reason the house should adapt the temperature, the level of lighting and sometimes present his favourite TV program. Suki prefers a warm living room and a cold bedroom.

The goal “Feel comfortable after work” can be described, for example, with abstract tasks such as:

1) Set a comfortable temperature (TEMP)
   a. Sense the indoor and outdoor temperatures
   b. Adjust room heating/cooling according to the user preferences and context.
2) Set a comfortable level of lighting (LIGHT)
   a. Sense the indoor light levels
   b. Adjust indoor light levels according to the user preferences and context.
3) Select favourite TV program (FAVTV)
   a. Check Media options
   b. Set Media according to the user preference and context.

At home the abstract task TEMP could be made concrete by the PA with the use of the entities mentioned in the last section. In this scenario the concrete task may look like:

1) Concrete TEMP
   a. FTA senses indoor temperature using Sensor S1 in the living room.
   b. FTA senses indoor temperature using Sensor S2 in the bedroom.
   c. FTA senses outdoor temperature using Sensor S3.
   d. FTA checks Suki’s temperature preferences stored in his local ontology.
   e. FTA deduces Suki’s favourite temperature for the several rooms.
   f. FTA adjusts temperature by using the windows
   g. FTA adjusts temperature by using the radiator
   h. IA provides control dialogues (using different modalities) for allowing Suki to directly adjust the temperature.

The OM aligns the entities referring to this plan and Suki is glad about the setting but on the next day the scenario continues as follows:

Suki has to stay the night at a hotel in the city close to his office due to a public transportation strike. When Suki enters the hotel room the AS detects that the goal “Feel comfortable after work” should be set up in a different environment.
The PA checks the hotel room’s device registry and adapts to the new environment by evolving the abstract task TEMP into a different concrete one:

1) Concrete TEMP
   a. FTA senses indoor temperature using Sensor S1 in the hotel room.
   b. FTA checks a weather forecast service to get the outside temperature.
   c. FTA checks Suki’s temperature preferences stored in his local ontology.
   d. FTA deduces Suki’s favourite temperature for the hotel room.
   e. FTA adjusts temperature by using the HVAC.
   f. IA provides a more sophisticated dialogue (using speech modality) for allowing Suki to directly adjust the temperature in the new environment.

The FTA recognizes the hotel room as a bedroom and sets up the HVAC to cold. Suki does not want to go to bed but wants to finish some work and for this reason he wants the AS to adjust the temperature to a normal level. He starts a set-up dialogue and tells the system always to set the temperature to a normal level if there is only one room available.

The outcome of this set-up dialogue will be stored by the IA in Suki’s local ontology and makes it possible for the FTA to involve such information in future decision to make automatic adaptation more exact.

VI. CONCLUSIONS

We have presented an approach for supporting the realization of adaptive pervasive applications. It is based on (a) the existence of heterogeneous smart components (devices, services) within an Ambient Intelligence environment, (b) the fact that these components maintain and make available local representations of their self and state and (c) their ability to communicate and collaborate. We have termed this set of components an Ambient Ecology and we use it to realize user tasks. We conceptualize an Activity Sphere a set of coherent tasks aiming to the same goal.

Our system supports the instantiation of an Activity Sphere within a specific Ambient Ecology. To achieve this, we propose the engineering of a sphere ontology by matching the local ontologies of the ecology components. The sphere ontology will serve as a common repository of knowledge and information among passive and active sphere entities. The latter are users and agents of three kinds: fuzzy task agent, planning agent, and interaction agent. All these maintain their local knowledge bases, which are also matched in the sphere ontology.

After drawing the conceptual picture of ATRACO our future work aims toward implementing a basic working system that can be used as a starting point for a evaluation sessions with real users.

ACKNOWLEDGMENT

The described work has been supported by the European Commission funded research project ATRACO (www.atraco.org) (Grant agreement No. 216837) within the 7th Framework Programme (FP7) as part of the FET Pervasive Adaptation (PERADA) initiative.

REFERENCES