# A Service Oriented Architecture Combining Agents and Ontologies Towards Pervasive Adaptation

Christos GOUMOPOULOS[a,1] and Achilles Kameas[a, b]
[a]*Research Academic Computer Technology Institute, DAISy group, 26500 Rion, Patras, Hellas*
[b]*Hellenic Open University, 23 Sahtouri Str., 26222, Patras, Hellas*

**Abstract.** Contemporary software technologies complying with the Service-Oriented Architectural (SOA) paradigm, such as OSGi, UPnP, and the Web services do not meet on their own the adaptability and interoperability challenges of the Ambient Intelligence (AmI) environments. In this paper we present a solution based on the combination of the SOA model with Agents and Ontologies. The agent approach complements the SOA infrastructure by providing high level adaptation to user's tasks, as an intelligent control layer above SOA. Ontologies are used to tackle the semantic heterogeneity that arises in AmI spaces and provide to agents a common repository of system knowledge, policies and state.

**Keywords.** ambient ecology, agents, SOA, adaptation, ontology

## 1. Introduction

Ambient Intelligence (AmI) is a paradigm that puts forward the criteria for the design of the next generation of UbiComp environments [11]. In this context we have introduced the *Ambient Ecology* (*AE*) metaphor to conceptualize a space populated by connected devices and services that are interrelated with each other, the environment and the people, supporting the users' everyday activities in a meaningful way [5].

In the context of the EU funded R&D project ATRACO [6] we aim to extend the AE concept by developing a conceptual framework and a system architecture that will support the realization of adaptive and trusted AEs which are assembled to support user goals in the form of *Activity Spheres* (*ASs*). Our approach is based on a number of well established engineering principles, such as the distribution of control and the separation of service interfaces from the service implementation, adopting a SOA model combined with intelligent agents and ontologies. Agents support adaptive planning, task realization and enhanced human-machine interaction while ontologies provide knowledge representation, management of heterogeneity, semantically rich resource discovery and adaptation. ATRACO ASs are dynamic compositions of distributed, loosely-coupled and highly cohesive components that operate in dynamic environments.

Therefore the architecture and the system we propose operate in an AmI environment, which is populated with people and an AE of devices and services. Our

[1] Corresponding Author.

basic assumption is that the AE components are all autonomous, in the sense that (a) they have internal models of their properties, capabilities, goals and functions, and (b) these models are proprietary and "closed", that is, (i) they are not expressed in some standard format and (ii) they can only be changed by the owner components. However, each component can be queried and will respond using a standardized protocol.

Previous research projects have applied the SOA paradigm to support dynamic service composition [1] or have combined SOA with ontologies to form a conceptual framework [9]. Most research efforts that have contributed to adaptation of ubiquitous applications during migration across different pervasive computing environments [10] provided little or no support for adaptation based on context information. Other research provided support for adaptation based on context information [1]. In other research efforts, ontology techniques, such as merging and mapping have been adopted, but they all use ontologies as static objects. ATRACO architecture builds upon previous research by supporting a multi-dimensional pervasive adaptation functionality into AmI spaces. Besides SOA a novel mechanism is proposed to achieve the different kinds of adaptation centered upon the management of knowledge, which is encoded in multi-layered ontologies, which are used by intelligent agents.

In the next section we will present the main concepts underlying our approach. In Section 3 we discuss the research and design challenges that the proposed architecture should tackle. In Section 4 we outline the architecture in the form of basic components and describe their role towards meeting the adaptability and interoperability requirements previously established. Finally our conclusions are given.

## 2. ATRACO World Model

The concepts discussed below constitute a critical subset of the ATRACO conceptual framework defined for building AmI applications.

| **Concept** | **Description** |
|---|---|
| *Ambient Ecology* (*AE*) | The set of heterogeneous artefacts with different capabilities and provided services that reside within an Intelligent Environment (IE). |
| *Activity Sphere* (*AS*) | It is formed to support an actors' specific goal. An AS represents both the model and the realization of the set of information, knowledge, services and other resources required to achieve an individual goal within an IE. The concept of AS is a "digitization" of the concept of "bubble" used by the psychologist Robert Sommer [12] to describe a temporary defined space that can limit the information coming into and leaving it. |
| *Intelligent Environment* (*IE*) | A territory that has both physical properties and offers digital services. It is the container of AE. ASs are instantiated in an IE using the resources provided by its AE. |
| *Artefact* | A tangible object which bears digitally expressed properties; usually it is an object or device augmented with sensors, actuators, processing, networking unit etc. or a computational device that already has embedded some of the required hardware components. |
| *Actor* | Any member of AE capable of setting and attaining goals by realizing activities. Within the AE actors are users or agents. |
| *Goal* | Each actor may have its own set of goals and plans to achieve them. A goal is described as a set of abstract tasks, which is described with a task model. |
| *Task Model* | It may be *abstract* or *concrete*. An abstract task model describes what should |

| Concept | Description |
|---|---|
| | be done, without details of how it should be done or by the use of what kind of modality; these are described in the corresponding concrete model. The abstract task model may also contain several decomposition rules modelled as a set of subtasks. |
| *Local Ontology*(*LO*) | Each member of the AE stores locally descriptions of its properties, services and capabilities. It is a sub-class of the class Ontology |
| *Sphere Ontology* (*SO*) | The SO results from the LO of those AE members that are required to achieve the AS's goal based on the resolution of its task model. Apart from device and service ontologies, it may contain user profiles, agent rule bases and policies. It is another sub-class of the class Ontology. |
| *Agent* | A software module (is a kind of actor) capable of pursuing and realizing plans in order to achieve specific goals based on tasks. It includes three types of agents: *Task Agent* (e.g., *Fuzzy systems based Task Agent* or *FTA*), who manipulates sensors and actuators in order to realize specific tasks; *Planning Agent* (*PA*), who resolves an abstract task hierarchy into concrete tasks using the resources of the AE; and *Interaction Agent* (*IA*), who manages user-system interaction using a mixed-initiative dialogue model. |
| *User* | The actor that uses the available services and devices in order to perform a task. When a user performs a task, this can be subdivided into different activities. Users use devices, which provide them with services. Devices run these services in a physical environment (context). Users use these services according to personal conditions (user profile) and within a physical context. |
| *Aim* | It is attributed to a user; it is decomposed into a set of interrelated goals, which are distributed to the components of the AS. |
| *Policy* | Actors specify high-level rules for granting and revoking the access rights to and from different services. Examples of policy ontologies are privacy policy ontology, interaction ontology and conflict resolution policy ontology |
| *Service* | The entity which describes the service offered by a device. |
| *Device* | The entity that has physical/digital properties and offers a specific service. |
| *Resource* | A resource can be the *space*, an *entity*, or a *component*, such as managers (e.g. Ontology Manager, Sphere Manager) or other basic components. |

The basic terms and concepts of the ATRACO world model are encoded in the ATRACO Upper Level Ontology (ULO). In general, ontology is used as the means to share information among heterogeneous parties in a way that is commonly understood [7]. An ontology is a network of concepts and entities, which can be associated with different types of relations (the most common being the hierarchical association, or is-a relation). More concrete (or domain) ontologies contain also instances of these entities with specific properties and values. More powerful ontologies contain constraints and rules that cause inferences for the entities. Figure 1 illustrates in UML representation the AS domain model which is also encoded as ontology in the ATRACO ULO.

## 3. Research and Design Challenges

Initial requirements, captured with applications scenarios, were used as input for a process of abstraction which allowed the identification of a set of challenges that must be addressed, in order to design adaptive pervasive systems. These challenges are listed in the following, together with the approach we have adopted in ATRACO in order to deal with them.

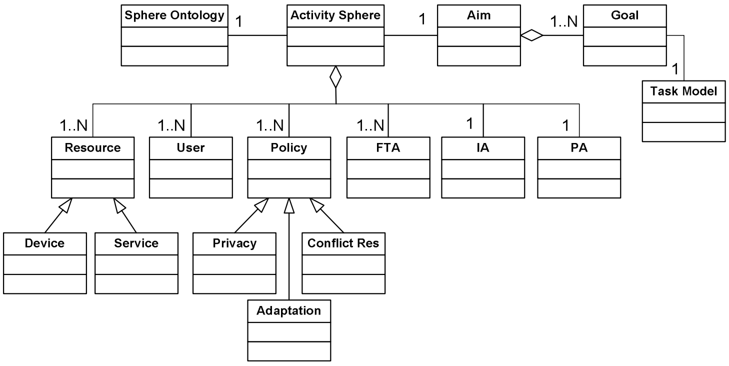

**Figure 1.** Activity Sphere domain model (part of ATRACO ULO)

### *3.1. Assemble/Dissolve Applications*

Actors must be able to assemble and dissolve pervasive computing applications in the form of collections of resources, in order to achieve their goals. Such an action should be possible for both users or and software agents.

In the ATRACO system level, a service agent, called *Sphere Manager*, is defined that generates service compositions based on a detailed task model that is generated for the given goal. Other agents (*Planning Agent* and *Interaction Agent*) are involved when a human actor participates in the composition process guiding the composition to enable the user to perform activities in the way she wishes to do. A *Sphere Ontology* is defined by aligning the local resource ontologies using the ATRACO ULO (ULO will be explained in the next subsection). An *Ontology Manager* is defined as a separate component to provide access to the Sphere Ontology.

### *3.2. Adaptability*

In ATRACO, at the AE level, the system supports the realization of the same AS in different IEs. At the same time, the system adapts to changes in the configuration of the AE (i.e., a new device joining, a device going out of service, etc.). At the task level, the system realizes the tasks that lead to the achievement of user goals using the resources of the AS. The artefacts also adapt to the uncertainties associated with the changes in the artefacts characteristics, context as well as changes in the user(s) preferences regarding these artefacts and their operation. Another dimension of adaptation concerns the interaction between the system and the user, in order to tailor the degree of system transparency to each specific user. A further dimension of adaptation is related to the network adaptation to allow devices and services to be used seamlessly by the ATRACO system and to simplify the discovery, management and access of networks in the home as well as in corporate environments.

We argue that in order to achieve complete pervasive adaptation in IEs, any infrastructure should provide adaptation in several forms: *functional adaptation* (the realization of the same AS in different AmI environments); *structural adaptation* (the persistent achievement of the goal when changes on the type of the available resources occur); *semantic adaptation* (changes in semantic models in order to deal with any disturbance that would affect meaning); *behavioral adaptation*, where the application logic is changed as a result of learning - specialized as artefact and user behavior model adaptation); *user interaction adaptation* and *network adaptation*.

### *3.3. Semantic Heterogeneity*

As mentioned above, the deployment of any pervasive application over an IE requires the orchestration of available services in the IE. In the general case, we expect that these will be heterogeneous and that they would not adhere to a specific protocol. However, the system must ensure the interoperability and the user centric operation of the pervasive application. In order to achieve these we use ontologies to encode local resource information and user preferences. Then, to ensure the user centric operation of the sphere, we compose a SO by matching the LOs of the sphere resources, so as to ensure interoperability between the various services and devices. Moreover, the pertinent policy ontologies are matched to ensure correct sphere operation. Finally, the user profile ontology is matched to ensure that the sphere will serve a specific user goal (and its associated tasks) and take into account the user preferences and experience. The SO encodes the information and knowledge necessary for sphere operation; it also provides context representation for the components of the sphere.

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. Current techniques for ontology matching require access to the internal structure of constituent ontologies, which must be verified for consistency, and result in static solutions (a set of mappings or a new ontology), which in addition have to be stored somewhere. But an AS is a transitory, dynamically evolving entity, composed of heterogeneous, independent, usually third-party components. That is why we choose to apply the ontology alignment technique [4].

### *3.4. Trustworthiness*

Privacy and trust are two important properties of the design space of any pervasive application. Several approaches use centralized components or third party control; these approaches do not scale and imply loss of privacy and autonomy.

In ATRACO, we chose to control our resources through policies rather than fabricate new mechanisms. We follow a distributed scheme where each resource has its own polices encoded as ontologies using a declarative approach. This approach decouples the declarative policy expression from the mechanism that ensures the desired behavior. Identity Management and privacy policies encoded in a privacy policy ontology constitute the main mechanisms to address privacy and trust. Polices expressed as a set of factual and behavioral specifications that are binding on every computing element and resource within an AmI space can be specified independently, leaving dependencies and conflict management to a reasoning framework.

### *3.5. Sharing of Resources*

Sharing of resources may lead to conflicts between pervasive applications. For handling such conflicts, it is necessary to apply specific policies. In ATRACO, an extension of the scheme discussed in the previous section is applied. A policy-based resource management scheme is defined and conflict resolution policies are encoded as ontologies. Such policies are used to describe and restrict the way each one resource is used, and to perform resource allocation when multiple applications have similar requests. Another use is to describe how the usage of one resource is dependent on (or constrained by) another.

## 4. Architecture

The ATRACO approach uses a Service-Oriented Architecture (SOA) that enforces a clear distinction between service interfaces and implementation. SOA has been envisioned as an evolution of the component-based architectures centred on the concept of service [1]. The SOA approach appears to be a convenient architectural style for realizing adaptable and reconfigurable systems.

In ATRACO we propose a combination of the SOA model with Agents and Ontologies. The agent approach complements the SOA infrastructure by providing high level adaptation to user's tasks, as an intelligent control layer above SOA. Agents have a local knowledge base that contains rules about the control of their behaviour and they may communicate and exchange messages which contain a high degree of semantics because of internal processing. In ATRACO, agents support adaptive planning, task realization and enhanced human-system interaction. Ontologies are used to tackle the semantic heterogeneity that arises in AmI spaces and provide to agents a common repository of system knowledge, policies and state.

On a system level ATRACO services will be provided by a set of system components. The role of each component has been specified so that their interaction can provide the adaptability and trust we envision as part of the realization of ASs. The architecture that supports the realization of ASs is shown in Figure 2. General functional blocks were defined, and the illustrated component diagram shows the overall functional blocks that were identified.

The Sphere Manager (SM) forms or dissolves an AS for a specific user goal. The SM is responsible for initializing the other system components and operates as an event service to them. SM implements a semantic-based discovery mechanism based on Sphere Ontology (SO) to resolve the services in the concrete plan provided by the Planning Agent (PA) to actual executable services provided in the AmI space. The SM composes an executable service workflow and implements an Execution Management and Control mechanism which is responsible for the execution of services on top of the SOA layer. An important role of the SM is to support the structural adaptation of ASs providing for the persistent achievement of the goal when changes on the type of the available resources occur. To achieve this, it monitors the state of execution of the task workflow and might change the composition of services in case of any problem.

The Ontology Manager (OM) matches local (i.e. device, agent, policy and user profile) ontologies according to the task model that fulfils the sphere goal. The OM is responsible for creating, dissolving and generally managing the SO and responding to queries regarding the SO. To that end, the OM maintains rules and provides inference services. The OM interacts with all system components.

One or more (depending on the goal complexity) Fuzzy Task Agents (FTAs) oversee the realization of given tasks within a given AmI space. These agents are able to learn the user behavior and model it by monitoring the user actions. The agents then create fuzzy based linguistic models which could be evolved and adapted online in a life learning mode [8]. The FTA maintains its own local knowledge base, which is initially formed by the SM, based on the task model and the SO.

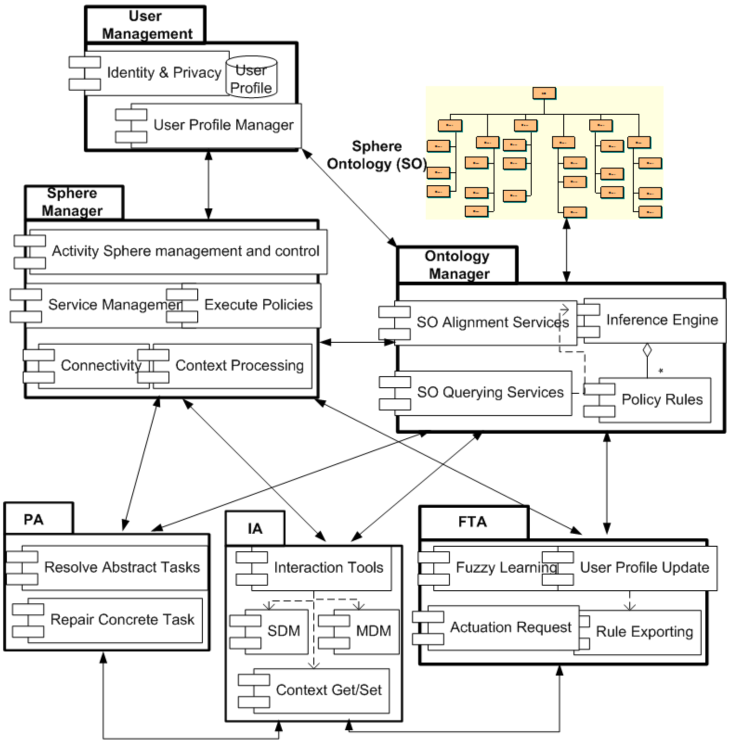

**Figure 2.** ATRACO component architecture

The Interaction Agent (IA) provides a multimodal front end to the user. Depending on the SO it optimizes task-related dialogue for the specific situation and user. The IA may be triggered both by the FTA and the PA to retrieve further context information needed to realize and plan tasks by interacting with the user. The IA uses two managers: Speech Dialogue Manager and Multimodal Dialogue Manager.

User Management (UM) contains the necessary functionality to provide, validate, and manage the information that forms the individual user's identity (user data) within the supported computing environment. More specifically, as a system component UM will include functionality for the provisioning, management and storage of multiple user identities, user specific preferences, and profiles.

The proposed approach has been validated in a real environment (iSpace facility). A prototype system has been developed based on the main components specified in the architecture and several basic components for controlling the environment (e.g., control of lights, HVAC, music player) and a sample AS was realized as outlined in Figure 3.

## 5. Conclusion

In this paper we have outlined an approach towards pervasive adaptation based on the combination of the SOA model with agents and ontologies. The mechanism we propose to achieve the different kinds of adaptation implied by this approach is centred round the management of knowledge, which is encoded in multi-layered ontologies, which are used by intelligent agents. The ongoing and future work will report on the full evaluations of the ATRACO system with multiple users and in various testbeds.

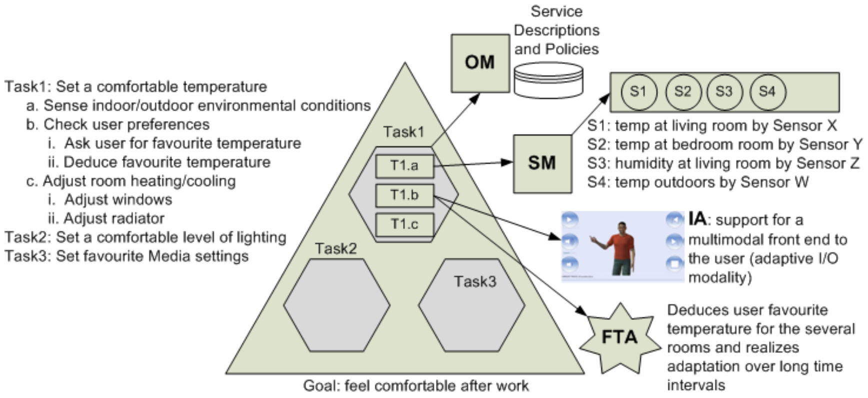
Service Descriptions and Policies
OM
Task1: Set a comfortable temperature
a. Sense indoor/outdoor environmental conditions
b. Check user preferences
i. Ask user for favourite temperature
ii. Deduce favourite temperature
c. Adjust room heating/cooling
i. Adjust windows
ii. Adjust radiator
Task2: Set a comfortable level of lighting
Task3: Set favourite Media settings
Task1
T1.a
T1.b
T1.c
Task2
Task3
SM
S1
S2
S3
S4
S1: temp at living room by Sensor X
S2: temp at bedroom room by Sensor Y
S3: humidity at living room by Sensor Z
S4: temp outdoors by Sensor W
IA: support for a multimodal front end to the user (adaptive I/O modality)
FTA
Deduces user favourite temperature for the several rooms and realizes adaptation over long time intervals
Goal: feel comfortable after work

**Figure 3.** ATRACO AS for the user goal “Feel comfortable after work”

## Acknowledgement

The research described is partly supported by the ATRACO (ICT-216837) project.

## References

[1] Ambient Intelligence for the Networked Home Environment project, website: http://www.hitech-projects.com/euprojects/amigo/

[2] H. Chen, T. Finin, and A. Joshi, A context broker for building smart meeting rooms. In Proc. AAAI 2004, pp. 53–60.

[3] T. Erl, *Service-Oriented Architecture : Concepts, Technology, and Design*, Prentice Hall PTR, 2005.

[4] J. Euzenat, A. Mocan, F. Scharffe, Ontology alignments: an ontology management perspective, in: Ontology Management: Semantic Web, Semantic Web Services, and Business Applications, Springer, New-York (NY US), pp. 177-206, 2008.

[5] C. Goumopoulos and A. Kameas, Ambient ecologies in smart homes, *The Computer Journal*, doi: 10.1093/comjnl/bxn042, 2008.

[6] C. Goumopoulos, A. Kameas, H. Hagras, V. Callagan, M. Gardner, W. Minker, M. Weber, Y. Bellik, and A. Meliones, ATRACO: Adaptive and Trusted Ambient Ecologies, Self-Adaptive and Self-Organizing Systems, Workshop on Pervasive Adaptation (PERADA), IEEE CS, pp. 96-101, 2008.

[7] T.R. Gruber, Towards Principles for the Design of Ontologies Used for Knowledge Sharing, *International Journal of Human-Computer Studies*, 43(5/6), pp. 907–928, 1995.

[8] H. Hagras, F. Doctor, V. Callaghan, and A. Lopez, An incremental adaptive life long learning approach for type-2 fuzzy embedded agents in ambient intelligent environments, IEEE Transactions on Fuzzy Systems, 15(1), pp. 41–55, 2007.

[9] PalCom project, website: www.ist-palcom.org

[10] A. Ranganathan, C. Shankar, and R. Campbell, Application polymorphism for autonomic ubiquitous computing, *Multiagent Grid Systems*, 1(2), pp. 109-129, 2005.

[11] P. Remagnino and G. L. Foresti, Ambient Intelligence: A New Multidisciplinary Paradigm, *IEEE Transactions on Systems, Man, and Cybernetics*, 35(1), pp. 1- 6, 2005.

[12] R. Sommer, Personal Space: The Behavioral Basis of Design, Prentice Hall Trade, Englewood Cliffs, N. J., 1969.