2009 Ninth International Conference on Intelligent Systems Design and Applications

Combining agents and ontologies to support task-centred interoperability in Ambient Intelligent Environments

Gaëtan Pruvost
LIMSI
The National Center for Scientific Research
Orsay, France
gaetan.pruvost@limsi.fr

Achilles Kameas, Lambrini Seremeti
DAISy Research Unit
Computer Technology Institute and The Hellenic Open University
Patras, Greece
kameas@cti.gr, seremeti@cti.gr

Tobias Heinroth, Wolfgang Minker
The Institute of Information Technology
Ulm University
Ulm, Germany
tobias.heinroth@uni-ulm.de, wolfgang.minker@uni-ulm.de

***Abstract*—This article describes our approach towards the specification and realization of interoperability within Next Generation Ambient Intelligent Environments (NGAIE). These are populated with numerous devices and multiple occupants or users exhibit increasingly intelligent behaviour, provide optimized resource usage and support consistent functionality and human-centric operation. In NGAIEs users will interact with their environments using the devices therein complemented with adaptive multimodal dialogue. This requires the definition of the local and global information which is relevant to the interaction and mechanisms to share this knowledge among entities. In our approach, knowledge is represented as a set of heterogeneous ontologies which have to be aligned in order to provide a uniform and consistent knowledge representation. The combination of heterogeneous ontologies and ontology matching algorithms allows for semantically rich information exchange. Based on a combination of agent-based and service-oriented architectures, the proposed approach adopts a task-based model to maximize the use of available heterogeneous resources.**

***Index Terms*—Ontology alignment, Model-based Interaction, Agent-based architecture, Task-centred model**

I. INTRODUCTION

AMBIENT Intelligent Environments (AIEs) are (usually closed) spaces equipped with a variety of sensors, devices and services. The high degree of distribution, heterogeneity and autonomy of these components makes interoperability within AIEs a non-trivial task.

The Next Generation of AIEs (NGAIEs) will be populated with numerous objects and have multiple occupants. These objects interact with each other and the environment, thus comprising an ambient ecology. All these objects are considered to be the basic building blocks of pervasive applications; the latter are regarded as orchestrations of services offered by the objects. Despite the distribution of computing components and services within NGAIEs, users' interaction with a ubiquitous computing system will not cease to be task-centric: in order to implement an activity, users are still interested to carry out specific tasks, using the skills, tools and information available in their heads (cognitive tools), or in the environment (interactive tools). Goals and tasks are independent from any ecology, but always must be realized within some ecology. The realization of a goal requires the binding of ecology resources to its concrete tasks.

We have defined the concept of an activity sphere (AS), to be both the model and the realization of the set of information, knowledge, services and other resources required to achieve an individual goal within a NGAIE. The formation of AS is supported by an ATRACO system using 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 [1]. This paper describes how the ATRACO architecture uses agents and ontologies to support interoperability and user interaction within NGAIEs.

Today's Human-Computer Interaction (HCI) technologies are usually characterized by a fixed variety of modes, modalities, and media for each task or service. A mode refers to the human sensory system used to produce or perceive given information (visual, gestural, auditory, oral, tactile, etc.). A modality is defined by the information structure that is perceived by the user (text, ring, vibration, etc.) and not the structure used by the system. Finally, a medium is a physical device which supports the expression of a modality (screen, loudspeakers, etc.). In NGAIEs, available modes, modalities and medias will no longer be static and user interaction has to opportunistically adapt to dynamic variations of the context of interaction.

User interface adaptation has been explored notably with the concepts of plasticity [2] that tackle the issue of modifying the user interface depending on the inputs, the outputs and the

978-0-7695-3872-3/09 $26.00 © 2009 IEEE
DOI 10.1109/ISDA.2009.195

running platform. In NGAIEs, two main issues complicate the interaction adaptation. Firstly peripherals are heterogeneous and distributed over a network and secondly the context of interaction, as defined in [3] is very dynamic and may even change during the interaction.

Previous works, such as CAMELEON-RT [4] or WWHT [5] emphasised the need for an architecture that combines multiple modalities and devices in order to provide rich and adapted interaction. We describe in section II.C our approach of an agent that relies on ontological description of context to realize task-centred user interaction.

Several approaches to agent-based AIEs such as [6] and [7] have been proposed but, to the best of our knowledge, this is the first attempt that uses ontology alignment as an integrated interoperability mechanism to achieve context-based adaptation within agent-based environments. A similar approach is proposed in [8], where an architecture that achieves adaptation based on context information is described, but is applied to a traditional static pervasive systems architecture. They provided little or no support for adaptation based on context information. . Other research projects [9], [10] provided support for adaptation based on context information. In these attempts, ontology techniques, such as merging and mapping have been adopted, but they all use ontologies as static objects.

In Section II, we present the agents of the ATRACO architecture. Section III illustrates task-centred interaction with an exemplary scenario showing how the architecture works. The last section explains the role of ontologies in the inter-agents communication and dynamic sharing of knowledge.

II. SYSTEM ARCHITECTURE

In this section we present the different agents that take part in the realization of the Activity Sphere. Each AS is realized as a ubiquitous computing application based on an agent-based, service oriented approach, which consists of active and passive entities, intelligent agents, and ontologies. Active entities manage the binding or resources, allocate and realize the tasks and manage knowledge and information exchange; two main software modules, the Sphere Manager (SM) and the Ontology Manager (OM) provide foundational functionalities within the AS such as service discovery, data access, and event handling. Passive entities provide the services that are orchestrated by the SM in order to realize the AS goal. Intelligent agents realize adaptation; the agents (Interaction Agent (IA), Planning Agent (PA), and Task Agent (TA)) utilize the AS and offer user-centred services, while, as like the other entities they also provide their own local ontology. Finally ontologies encode global and local knowledge and state-related information. Passive entities encode their state, properties, capabilities, and services in local ontologies. Furthermore agents, such as the IA maintain their own knowledge base. In this paper, we exemplary present one of those knowledge bases: the Interaction Ontology (IO). The Sphere Ontology (SO) is the main information and knowledge pool formed by aligning, merging and mapping of the passive entities' and the agents' local ontologies, which is constantly evolving and being updated.

Among those agents, we focus on the SM, OM and IA in order to highlight the role of ontologies and knowledge sharing between agents for the realization of interactive tasks.

A. Sphere Manager

The Sphere Manager is responsible for creating, managing and dissolving spheres. It instantiates the various agents which are responsible for supporting adaptation, interacting with the user and monitoring the realization of the concrete tasks. Each goal is decomposed in a hierarchy of abstract tasks. After initialization, the PA is responsible for resolving the abstract tasks into concrete tasks, based on the resources of the ambient ecology. Based on the task description, the SM discovers the necessary ambient ecology components to be included in the AS and orchestrates their services in order to realize the tasks in the task model. The TA is responsible for realizing one or more concrete tasks in an adaptive way. The IA serves as direct connection between the user and the system and provides adaptive multimodal dialogue.

B. Ontology Manager

The Ontology Manager offers two important services: Ontology alignment and Ontology querying. As a process, an alignment follows specific methods (local, global) and uses tools (in our case we are using Alignment API [11] and Falcon-AO [12]), in order to provide the desirable result. There are two matchers integrated in Falcon-AO: a linguistic and a graph one. Its linguistic matcher relies on lexical comparison and statistical analysis, while its graph matcher uses directed bipartite graphs to represent ontologies and measures the structural similarity between graphs.

In ATRACO we adopt the alignment process given in [13], whereby two or more ontologies that are compared pair-wise. The output of the process is a list of correspondences, which indicate the relations between the entities of the ontologies with an additional confidence value. The degree to which an alignment process can be automated and the performance of the process depends on the complexity and the size of the ontologies. The complexity of alignment algorithms currently proposed in the literature varies from $O(n \log(n))$ to $O(n^2 \log^2(n))$. After aligning the local ontologies, the OM can answer queries by following the alignment points in order to query the local ontologies and apply inference mechanisms to compose the query results. In the case of ATRACO we use the Jena inference engine.

C. Interaction Agent

The IA is responsible for interacting with the user. Based on the WWHT approach [5], we designed a process that selects the most suitable interaction component at runtime thus offering interaction adaptation. Fig. 1 gives a general view of this agent's architecture. The IO encodes local knowledge about interaction and context. It also describes a set of interaction components called Off-the-shelf Interaction Objects (OIO) that are available for interacting with the user. Based on this information and on a set of generic rules provided by the designer, a module called Multimodal Manager selects at runtime the most suitable combination of OIOs through a library. This selection is made in two steps called allocation and instantiation. Allocation refers to the

selection of a combination of OIOs and devices to run them on whereas instantiation refers to the selection of values for the properties of those objects (style, size, etc.). The process of selection follows Rousseau's approach of an election system in which each rule grants or removes points to possible solutions [5]. This approach is hybrid between a fully top-down refinement of task model such as proposed in TERESA [14] and a bottom-up generation of user interface based on existing user interface elements. Since OIOs can be of varying complexity, it affords both the flexibility of low-level components and expressivity and richness of high-level interaction components. The possibilities of interaction and adaptation of the IA thus relies on the richness of available OIOs.

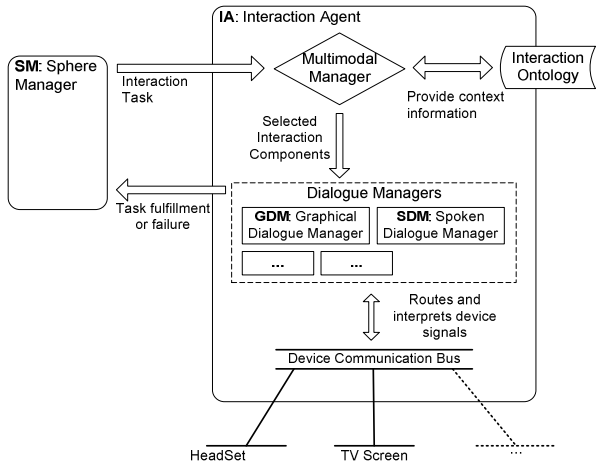

Fig. 1 Interaction Agent Architectural overview

Once the selection is done, Dialogue Managers are responsible for controlling the evolution of the interaction tasks during runtime. They maintain the current state of the interaction, make it evolve, adapt it to user's inputs and inform the SM of task fulfilment or failure. Different dialogue managers are envisioned depending on the interaction style they offer. For instance, a Spoken Dialogue Manager (SDM) transforming task description into standardized VoiceXML representation is proposed for natural speech.

D. Other agents

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. Its aim is to resolve all goals into concrete tasks given a specific ecology. Each Task Agent is responsible for the realization of a single concrete task. TAs are associated with specific input (e.g., sensors) and output entities (e.g., actuators) in order to realize the current task. If a specific task has been realized in the past, the TA can retrieve the respective fuzzy rule set from the user profile. If the task is considered novel, the TA 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). 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. The linguistic labels for each of these functions are retrieved from the SO when needed.

III. REALIZING TASK-CENTRED INTERACTION

In this section we shall present a scenario to illustrate the realization of user tasks within NGAIEs. The user (Suki) is used to such environments and regularly uses his ATRACO system, which he calls *Julia*. In our scenario Julia assists Suki in attaining the goal *"Cook Pasta for dinner"*. To reference the different episodes of the scenario we use text marks B1 to B5 for the episodes of the task description.

Suki starts cooking the pasta. Without him being aware of it, Julia starts a new activity sphere, which includes all the necessary "resources": the recipe, the kitchen, the cooking pot and the ingredients, pasta and salt, which are placed on the kitchen top [B1]. It also includes interaction resources such as microphones, speakers and screens. Suki fills the cooking pot with water and places it on the kitchen stove, which automatically switches on to the correct temperature, according to the recipe [B2]. Suki occupies himself with other cooking tasks in the kitchen. When the water in the cooking pot has almost reached the boiling temperature, Julia locates Suki (he is nearby, in the kitchen), informs him to add some salt and warns him that in a couple of minutes, he'll have to add the pasta [B3]. A couple of minutes later, Suki adds the pasta; Julia detects that Suki has lifted the pasta from the kitchen top and infers that he is going to add them [B4]. Julia confirms with Suki that pasta has been added and informs him that, given the current stove temperature, the pasta will boil within twelve minutes [B5].

A. Task Analysis

This section shows how different knowledge representations are used to implement the above scenario in a structured way. Julia starts the "Cook Pasta" AS. It initializes the SM and passes to it the pasta cooking recipe (in a suitable format, but this is of no concern here). Using it, the SM locates in the NGAIE, all the resources offering the necessary services; these are candidate members of the "Cook Pasta" AS. The SM then passes the references to these resources to the OM, which aligns their local ontologies and creates the SO. Then, the PA is instantiated, which uses the recipe (i.e., the abstract task description) and the SO in order to produce a concrete task description. The PA queries the SO for the properties of specific resources. Example queries are "get the ID of pasta package", which returns a list with the IDs of all available pasta packages or "get the ID of salt container", which returns a list with the IDs of all available salt containers.

In our example, the following concrete tasks are produced:

1. Boil water (B2)
   a. Get cooking pot (locate a specific cooking pot)
   b. Add water (quantity depends on the number of persons to be served)
   c. Place cooking pot on stove (locate stove; place the cooking pot on it; stove automatically switches to maximum cooking temperature)
   d. When water temperature reaches 100 C, signal via the Interaction Agent that water-is-boiling

2. Add salt (B3)
   a. Get salt (locate a specific salt container)
   b. Inform Suki to add-salt
   c. Add salt (quantity depends on the water quantity)
3. Add pasta (B4)
   a. Get pasta (locate a specific pasta package)
   b. Inform Suki to add-pastas
   c. Add pasta (quantity depends on the number of persons to be served)
4. Boil pasta(B5)
   a. Based on the recipe, stove temp is reduced to 80% of maximum temp
   b. Cooking-time is set to 12 min, according to the info on the pasta package
   c. When cooking-time reaches limit, then stove produces signal cooking-over and switches off
   d. Stove produces signal remove cooking pot

The SM allocates three TAs to monitor the realization of these tasks: one for boiling the water (B2), one for adding the ingredients (B3 – B4) and one for boiling the pasta (B5).

Each TA is responsible for monitoring the realization of a task and adapting to any changes that may occur. Based on the SO, each of these agents will create a set of rules in order to describe the requirements of the task it monitors. For example, regarding the "Boil pasta" task, the agent will create, among others, rules to set the cooking-time, to switch the stove off when cooking should be over, to produce an alarm two minutes before cooking is over, to produce an alarm to remove the cooking pot, etc.

Within B2 the user indirectly triggers the stove to switch on to maximum temperature by putting the cooking pot filled with water on the stove. The TA monitors the temperature of the water (via the cooking pot) and when this reaches a boiling state, it asks the SM to trigger the IA to inform and warn the user within B2. The "boiling" water state is a linguistic label attached to the boiling temperature and is extracted from the SO. The SM uses the IA twice to interact with the user: inform the user about adding salt and warn the user that the water will boil in 2 minutes (the latter is encoded in the cooking policy which is also aligned in the SO). The IA has access to available interaction devices via alignment of its local ontology with the SO. By querying the OM, it selects how to best interact with the user. In this scenario, as there is no screen available, the spoken dialogue modality is used. The same process occurs for tasks B3 and B4, B5.

IV. ROLE OF ONTOLOGIES

Ontologies and the OM have a great role in enabling the realization of the presented scenario by enabling knowledge sharing and interoperability between agents. In our scenario, two ontologies are important to the system: the SO and the IO.

A. Sphere Ontology

The Sphere Ontology is a virtual entity, as it is formed by aligning the local ontologies of the sphere resources (Fig. 2). This means that no concrete sphere ontology exists, which would require to merge the local ontologies. By choosing alignment instead of merging, we ensure that, each time the sphere ontology is queried; it produces an answer based on the latest state of the constituent ontologies.

In the example we follow in this paper, the OM aligns the local ontologies of the stove, the cooking pots and the pasta packages it can discover. Moreover, it aligns the user profile, the IO and media device ontologies. The OM receives pointers to these resources from the SM and then accesses their local ontologies. Some of the local ontologies are shown in Fig. 2, where we show the ontology of the pasta package (it is a package of lasagne), which encodes information about the package contents, weight, expiration date, etc, as well as information on how it can be cooked, the ontology of one of the cooking pots, which encodes information about its properties (size, material, colour etc.), services (it embeds sensors that read the volume and temperature of its contents) and usage (i.e., it can be used as a cooking pot or as a container) and the ontology of the stove, which, apart from its components and properties, indicates that cooking is one of the services it offers.

Then the OM applies alignment algorithms to produce alignment points (shown as connecting lines in Fig. 2). Now the OM can answer various queries posed by the agents or the SM. For example, it can locate a specific cooking pot, based on its ID, its location, its capacity or its appropriateness for use with an electric stove (all these values are contained in the cooking pot ontology). Then it can use the volume sensor of the pot to check whether the correct quantity of water has been added (the correct value has been calculated by the PA from the recipe and the number of pasta servings to be cooked). The pasta ontology, the cooking pot ontology and the stove ontology are aligned based on the "cooking" term, which is included in each of them. Then, the OM can answer queries about the cooking temperature, the water temperature (using linguistic labels it can determine whether the water is cold, hot, boiling, etc.), or the remaining time (in the pasta package ontology, a cooking time of 12 min is mentioned and the stove offers a timing service).

The IO is aligned with the resource ontologies based on the interaction properties of the devices. They are classified in interactive and non-interactive devices; clearly the stove and the pot are non-interactive devices. Other media devices (i.e., speakers, TV, photo frame, etc.) would be classified as interactive devices. Then, in order to realize interaction adaptation, the IA is able to ask the OM queries such as "is the media device mobile?", "If mobile, what is the level of battery?", or "Which modalities does it support?". Then, the modalities must be matched with the instances of the modality concept of the IO. Moreover, the IA can query about non interaction specific contextual information. Indeed, using the SO, the IA can use concepts of the IO as a vocabulary to get information about environmental conditions. Example queries it can pose are: "What is the noise level?" or "What is light level?".

B. Interaction Ontology

The IO is the knowledge base connected to the Multimodal Manager. It is part of the IA and provides concepts in order to dynamically share contextual knowledge with the rest of the system. By using the IO as a vocabulary description, the IA is able to formulate queries to the OM, and find answers in the

aligned sphere ontology. The main concepts of the IO are grouped in four categories: *User*, *EnvironmentalConditions*, *AllocationDimensions* and *ElectionLogic*.

The first two important concepts – *User* and *Environmental-Conditions* – do not refer to the description of the interaction in itself but rather to the context influencing the interaction decision-making process. This external knowledge is already represented and maintained by the Ontology Manager in the Sphere Ontology via alignment of user and local resource ontologies. Because some of these resources will be used for interaction purposes, we shall use those two concepts to facilitate the alignment between the interaction ontology and the sphere ontology.

The last two main concepts refer to the description of adaptation capabilities of the system. They allows for describing what an interactive task is, what an OIO is and how they relate to each other. They also provide a structure for the designer to describe contextual rules for the process of allocating and instantiating the most adequate communication channel.

Inferring systems and navigation through the instances of the ontology thus facilitates the factoring of the rule base. Of course, in order to propagate such properties from one instance to another and compute a weight for each solution, the Multimodal Manager must rely on the shared knowledge represented by the concepts and relationships defined in the ontology. But whenever this standardized structure is respected, new instances and concepts can be added and used with no time-consuming recoding of the Multimodal Manager.

V. CONCLUSION AND FUTURE PLANS

In this paper we have presented an architecture that combines agents and ontologies in order to support interoperability within NGAIEs; we have used a scenario to illustrate how users can collaborate with such systems. The proposed approach builds a virtual knowledge base, which is common for the resources that participate in the realization of a specific user task. This knowledge base is formed by aligning the heterogeneous ontologies of NGAIE resources, user profile, task description, and for example, all the information that forms the context of task-based HCI within a specific NGAIE. Then the aligned ontology can be used to answer queries that combine state, service or property related information of the constituent resources.

The Multimodal Manager of the IA provides capabilities to realize a model-based approach of generation of user interface that allows for the cohabitation of top-down and bottom-up design methods. The resulting manager relies on a predefined set of interaction objects and uses the IO to decide which one to instantiate and how to combine them. The rules that describe which components should be composed are still experimental. We plan to evaluate with real users such rules to determine precisely which aspects of the environment might influence the interaction with the user.

In the near future, we plan to evaluate more complex alignment algorithms that go beyond lexical or syntactic matching, so as to achieve closer semantic interoperability. However, it is possible that the complexity of such algorithms will not allow their application in (near) real-time, in which case, we shall research the development of less generic, but more efficient algorithms tailored to the specific ATRACO conceptual model. This will be possible with the help of an ATRACO Upper Level Ontology (ULO) that we are developing, which will serve as a pivot ontology, in order to align all constituent ontologies; as a further step, the merging of the simple constituent ontologies into the ATRACO ULO will be researched as an alternative, so as to decrease the number of component ontologies that need to be queried each time a query is posed to the SO. We are already developing a module that constructs simple ontologies from UPnP device headers, which it will subsequently merge in the ATRACO ULO.

ACKNOWLEDGMENT

The research leading to these results has received funding from the European Community's 7th Framework Programme (FP7/2007-2013) under grant agreement No. 216837 (project ATRACO) and from the Transregional Collaborative Research Centre SFB/TRR 62 “Companion-Technology for Cognitive Technical Systems” funded by the German Research Foundation (DFG).

REFERENCES

[1] Christos Goumopoulos, Ioannis Calemis, Kostas Togias, Achilles Kameas, Yacine Bellik, Hani Hagras, Tobias Heinroth, Gaëtan Pruvost, Christian Wagner, and Julien Bidot, “D06 - system specification and architectural design”, Public deliverable, The ATRACO Project (FP7/2007-2013 grant agreement n° 216837), 2008.

[2] D. Thevenin and J. Coutaz, “Plasticity of user interfaces: Framework and research agenda”, in *Human-computer Interaction, INTERACT'99: IFIP TC. 13 International Conference on Human-Computer Interaction, 30th August-3rd September 1999, Edinburgh, UK*. IOS Press, 1999, p. 110.

[3] A. K. Dey, *Providing Architectural Support for Building Context-Aware Applications*, PhD thesis, Georgia Institute of Technology, 2000.

[4] L. Balme, A. Demeure, N. Barralon, J. Coutaz, G. Calvary, et al., “Cameleon-rt: A software architecture reference model for distributed, migratable, and plastic user interfaces”, *Lecture Notes in Computer Science*, pp. 291–302, 2004.

[5] C. Rousseau, Y. Bellik, F. Vernier, and D. Bazalgette, “A framework for the intelligent multimodal presentation of information”, *Signal Processing*, vol. 86, no. 12, pp. 3696–3713, 2006.

[6] N. Hanssens, A. Kulkarni, R. Tuchinda, and T. Horton, “Building agent-based intelligent workspaces”, in *ABA Conference Proceedings*, 2002, pp. 675–681.

[7] A. Omicini, A. Ricci, M. Viroli, C. Castelfranchi, and L. Tummolini, “Coordination artifacts: Environment-based coordination for intelligent agents”, in *Proc. Third International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 2004, pp. 286–293.

[8] J. Euzenat, J. Pierson, and F. Ramparany, “Dynamic context management for pervasive applications”, *The Knowledge Engineering Review*, vol. 23, no. 1, pp. 21–49, 2008.

[9] M. Roman, C.K. Hess, R. Cerqueira, A. Ranganathan, R.H. Campbell, and K. Nahrstedt, “Gaia: A middleware infrastructure to enable active spaces”, *IEEE Pervasive Computing*, vol. 1, no. 4, pp. 74–83, 2002.

[10] H. Chen, T. Finin, and A. Joshi, “A context broker for building smart meeting rooms”, in *Proc. AAAI 2004*, pp. 53–60.

[11] Jérôme Euzenat, “An api for ontology alignment”, in *International Semantic Web Conference*, 2004, pp. 698–712.

[12] Ningsheng Jian, Wei Hu, Gong Cheng, and Yuzhong Qu, “Falconao: Aligning ontologies with falcon”, in *Integrating Ontologies*, 2005.

[13] Marc Ehrig, *Ontology Alignment: Bridging the Semantic Gap*, vol. 4 of *Semantic Web And Beyond Computing for Human Experience*, Springer, 2007.

[14] G. Mori, F. Paterno, and C. Santoro, “Design and development of multidevice user interfaces through multiple logical descriptions”, *IEEE Transactions on Software Engineering*, vol. 30, no. 8, pp. 507–520, 2004.

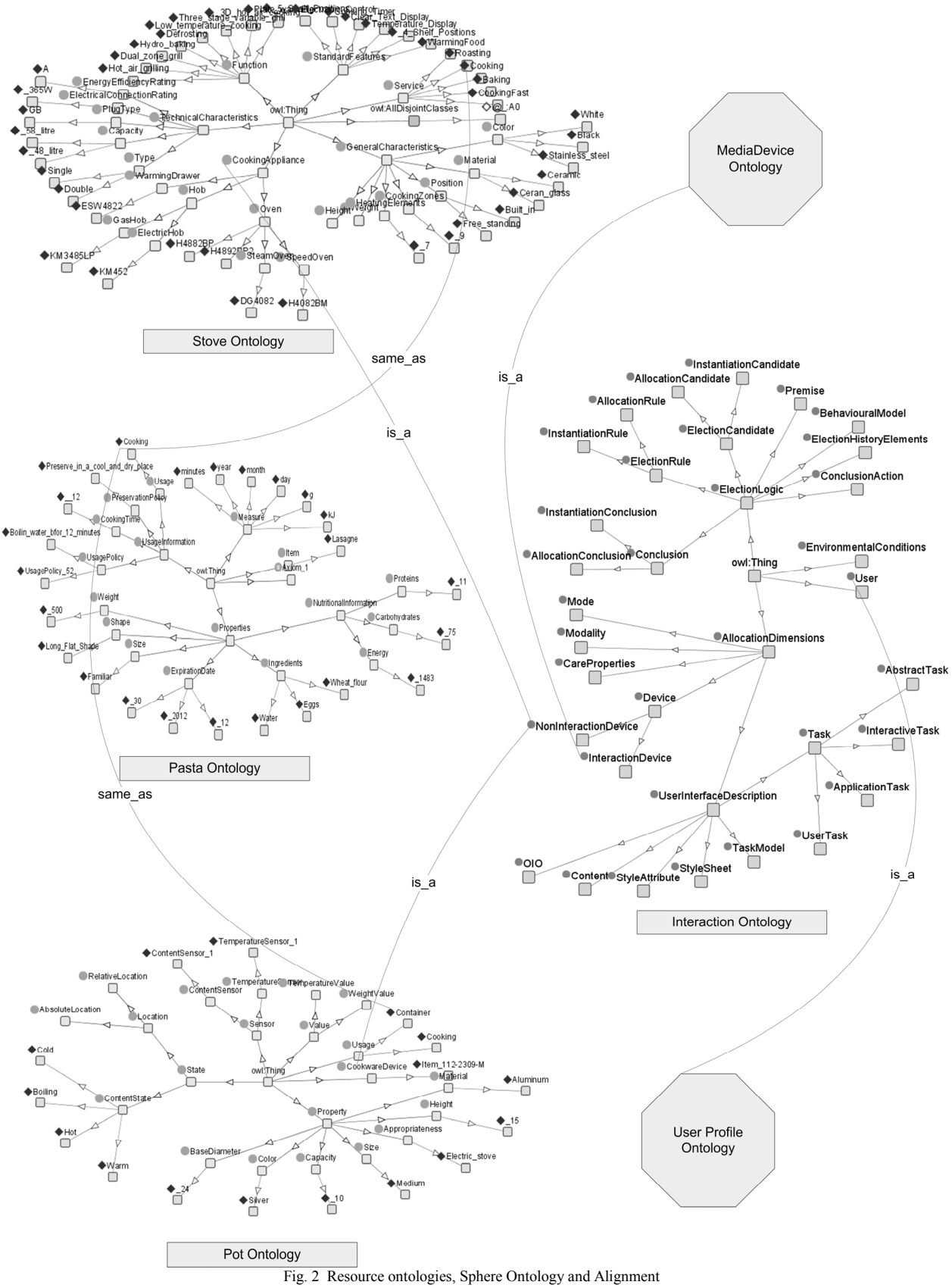

Fig. 2 Resource ontologies, Sphere Ontology and Alignment