A Common-Sense Planning Strategy for Ambient Intelligence*

Maria J. Santofimia[1], F. Moya[1], Juan C. Lopez[1], and Scott E. Fahlman[2]

[1] Computer Architecture and Network Group, School of Computing Science, University of Castilla-La Mancha, Spain
[2] Language Technologies Institute, Carnegie Mellon University, Pittsburgh, PA 15213, USA
{MariaJose.Santofimia,Francisco.Moya, JuanCarlos.Lopez}@uclm.es,sef@cs.cmu.edu

Abstract. Systems for Ambient Intelligence contexts are expected to exhibit an autonomous and intelligent behavior, by understanding and reacting to the activities that take place in such contexts. These activities, specially those labelled as trivial or simple tasks, are carried out in an effortless manner by most of the people. In contrast to what it might be expected, computers are struggled to deal with these activities, while easily performing some others, such as high profile calculations, so hard for humans. Imagine a situation where, while holding an object, the holder walks to a contiguous room. We have effortless inferred that the object is changing its location along with its holder. However, these sort of inferences are not well addressed by computers, due to their lack of common-sense knowledge and reasoning capability. Providing systems with these two issues implies collecting a great deal of knowledge about the everyday life, and implementing inference mechanisms to derive new information from it. The work proposed here advocates for a common-sense approach as a solution to the shortages of current systems for Ambient Intelligence.

Key words: Ambient Intelligence, Common Sense, Planning

1 Introduction

One decade after Mark Weiser defined the concept of Ubiquitous Computing, the IST Advisory Group brought up the Ambient Intelligence paradigm [1]. Lying on the Ubiquitous Computing paradigm, Ambient Intelligence refers to those environments where people are surrounded by all kind of intelligent and intuitive devices, capable of recognizing and responding to situations. In these contexts, people perceive the context as a service provider that satisfies their needs or inquiries in a seamless, unobtrusive, and invisible way.

* This work has been funded by the Spanish Ministry of Science and Innovation under the project DAMA (TEC2008-06553/TEC), and by the Regional Government of Castilla-La Mancha under project RGRID (PAI08-0234-8083).

Systems for Ambient Intelligence are expected to supervise the context and understand the activities that take place in it, since presumably, they are context-aware. Moreover, the behavior exhibited by these systems is supposed to be driven by a set of specific goals that are to be achieved, maintained, or satisfied. However, up to date, these systems remain quite far away from the envisioned scenarios in [1]. In the authors' opinion, the existing distance is grounded in the overlook of the common sense importance.

Under the common sense optic, the problem of developing systems for Ambient Intelligence has to be tackled from two different angles: the cognitive and the behavioral . From the cognitive perspective, the problem can be addressed as an **understanding** problem. Comprehending a situation that takes place in a context might involve, for instance, the inference of implicit, nondeterministic or delayed effects. A delayed effect of having a kitchen sink with a stopper, after opening the tap, will be a water overflow. From a behavioral perspective, the problem can be addressed as a **planning** problem of deciding what action to take under given circumstances. Therefore, a common-sense strategy to planning and understanding, as presented in [2] seems to be the most compelling approach to emulate the human-like rationality and reasoning capability.

The need for a planning strategy for Ambient Intelligence has already been stated in [3]. That work pays a special attention to the device heterogeneity existing in Ambient Intelligence contexts, advocating for a distributed-centralized HTN-like [4] approach. In spite of agreeing in the need for addressing device dynamism and heterogeneity, here, it is believed that these aspects have to be addressed from the middleware layer, being transparent for the planner. This work resorts to the framework proposed in [5] to this end. Eventually, authors agree with [3] in the role assigned to the multi-agent system architecture, as the context observer and regulator. The multi-agent system assumes the responsibility of providing the planner with the required information about the context and the mechanisms to respond.

From the cognitive perspective, understanding a situation involves the identification of implicit relations among events taking place. To this end, efforts have to be addressed to effectively represent the knowledge, from where these connections are to be inferred. This implicit knowledge, is normally referred as common sense knowledge. Mueller in [6] presents an extensive analysis about the key issues that should ground common sense systems. Such issues concern about representation aspects of events, time, effects of events, etc. This work advocates for a common-sense knowledge base and reasoning engine, and its lisp-like language to address these postulations.

At the risk of being too pretentious, by combining and supporting an action planner on a common sense system, this work proposes an approach to overcome the lacks and deficiencies of the approaches to systems for Ambient Intelligence available up to date. This work basically joints two long-studied fields, such as common sense and planning and make them work towards Ambient Intelligence.

The remainder of this paper is organized as followed. First, a state-of-the-art on Ambient Intelligence is presented, this section also states the require-

ments and needs that have to be targeted when seeking for autonomy in systems for Ambient Intelligence. The second section goes through the theoretical concepts grounding the planning strategy presented here. Section three describes the foundation of a planning strategy for Ambient Intelligence, and presents the commonsense approach that this work advocates for. Fourth section describes the implementation details of the proposed solution. Finally, the last section provides a summary of the most important ideas presented in this work.

2 A Semantic Model for Actions and Events in Ambient Intelligence

The strong coupling between the action and event theory and planning has given birth to the concept of *action planning*. Commonly, actions and events have been treated univocally, or with the slight difference of considering actions as events intentionally generated [7]. However, this work strongly believes that not only events are not actions, as defended in [8], but adopting this assumption prevents the planning strategy from being accomplished by means of a low-level guidance, as demanded by Ambient Intelligence contexts. The main argument supporting this dissociation lays on considering actions and their agents as inseparable or correlative [9]. The theory of action for multi-agent planning [10] also advocate for this distinction, although hinting that actions are accomplished by agents in their endeavor to achieve a goal. Despite agreeing in considering agents along with actions, this work does not consider actions in terms of the targeting goals, but as justified later on, in terms of requirements for the action to take place and consequences of the action

The planning problem has been traditionally stated in terms of a world description or initial state, the goal to be achieved and the available actions. Actions are specified in terms of prerequisites and the effects. Nevertheless, the planner strategy proposed here differs from this traditional approach in considering actions, in opposite to states of the world, as primary elements.

Planning problems under this perspective are here stated as a set of a **goal action**, which is a **non-feasible action** intended to be performed on an **object**. The planner seeks the set of **doable actions** that provide the means to produce the same effects on the object as the goal action. The following section will take care of the planning strategy details. In the mean time, this section introduces the *semantic model* at the root of this proposal.

However, what does a semantic model stand for? From our perspective, it is considered to be an agreement on how to interpret the knowledge represented in the knowledge base. Furthermore, resorting to a semantic model, to be shared among different instances, is also essential when these instances are expected to extract the same meanings or conclusions out of the represented knowledge.

The proposed semantic model, depicted in figure **??**, spins around the concept of **service**. Under the Ambient Intelligence perspective, services can be decomposed in the set of the **actions** performed on **objects**. These services are offered by **devices**. Adopting this semantic model has a significant impact over

the planning strategy, since not only preconditions and effects of actions have to be considered, but also the objects receiving those actions, and the devices providing the services that implement actions.

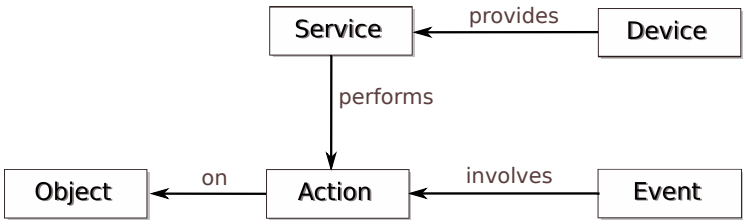

Fig. 1. The semantic model for Ambient Intelligence

Spreading the semantic model to the elements involved in the Ambient Intelligence architecture (Middleware and the Multi-Agent System), requires the semantic model to be formalized by means of some formal language. Ontologies are one of the most widely used approaches when it comes to knowledge representation. OWL[11] and RDF[12] are two standards commonly used to describe ontologies. Due to the nature of the relationships established among the classes of the semantic model, OWL2 language is used for description purposes. Figure 1 depicts the entities and their relationship composing the semantic model proposed to model application domain for Ambient Intelligence.

3 The Planning Strategy

The capability to exhibit a self-driven behavior is the most challenging requirement faced when developing systems for Ambient Intelligence. On this endeavor, devising architectures with the capability to compose services out of general specifications or requirements becomes an essential ingredient for self-sufficiency. Obviously, it is not a trivial matter, and so far it has only been achieved in a semi-automatic manner. Service composition has been largely studied and implemented in a wide spectrum of computer technologies, probably being Web Services[13] the most successful approach to address the problem. Here, the problem of supporting automatic service composition is reformulated as an action planning problem, enhanced with common sense knowledge.

3.1 Leveraging Common Sense

It is apparent that exhibiting an autonomous and self-sufficient behavior needs to be founded on the knowledge about how things work, and eventually, in the capability to make decisions based on that knowledge. In this regard, available

systems for commonsense knowledge are limited to a small group, mainly represented by Cyc[14], and Scone[15].

Cyc implements a Davidsonian[16] interpretation of events and actions. In the Cyc knowledge base, events are asserted as individuals about which facts can be stated, in order to specify the moment in time when the event took place, the location, or the performer agent, for instance. The approach adopted by Scone better meets the demands of the service composition.

Scone models the knowledge about events and actions in terms of the context state just before the action or the event takes place and immediately afterwards[17]. The planner exploits the fact that two actions that have similar **after contexts** and, therefore similar effects on the context are considered to be functionally equivalent. Moreover, the **before context** can be considered as the requirements that need to be met for the action to be performed. These two considerations are the cornerstone of proposed planning algorithm.

The following code provides some examples of action and event descriptions, in the Scone lisp-like syntax. Events and actions might have associated roles. For instance, the `detectingFace` action involves a `detect` event, and for that reason the action inherits from the event the `detectionSource` and the `detectionObject` roles. Roles are used to express the relevant characteristics of an event or an action being described. The `detectionSource` role is used to symbolize the source over which the `detect` event is performed. Generally, actions tend to specify the event roles. As it can be observed for the `detectingFace` action, the `detectionSource` role is concretized from being a general `thing` to be digital `data`.

As mentioned above, the before context states the requirements that need to be satisfied for the action to take place. The requirement for detecting a face is that there exist an image file from where to seek for a face silhouette. Therefore, the before context for the `detectingFace` demands the digital data to be an `imageFile` and not a sound or a text file. it can be can be accomplish, data have to be available, and so states the statement in the before context. The after context of the same action describes the effects of performing the action. The after context for the `detectingFace` action describes that after performing the action, a face silhouette has been observed in the `imageFile`, implicitly stating the existence of a person, although not yet identified.

```
(new-event-type {detect} '({event})
 :roles
 ((:indv {detectionSource} {thing})
  (:indv {detectionObject} {thing}))
 :before
 ((new-not-statement {detectionObject} {is noticed in} {detectionSource}))
 :after
 ((new-statement {detectionObject} {is noticed in} {detectionSource})))
(new-event-type {detectingFace} '({detect} {action})
  :throughout
```

```
((the-x-of-y-is-z {detectionSource} {detectingFace} {data})
 (the-x-of-y-is-z {detectionObject} {detectingFace} {face})
 (the-x-of-y-is-z {object-of} {detectingFace} {imageFile})
 (the-x-of-y-is-z  {agent} {detectingFace} {faceDetectorSystem}))
:before
((new-statement {data} {is recorded in} {imageFile})) ;;an image is required
:after
((new-statement {detectionObject} {not yet identified as} {person identity})
(new-statement {face} {is noticed in} {imageFile})))
```

This way of representing the knowledge about how things works as a network of nodes and link among nodes, along with its particular implementation of the inference and search process, makes automatic service composition a feasible task. The marker-passing algorithms implemented by the Scone engine provides a fast and efficient way of performing common-sense reasoning on the basis of the available knowledge. As stated in [18], although with no proof of logical completeness, Scone is capable of performing "*inheritance of properties, roles and relations in a multiple-inheritance type hierarchy; default reasoning with exceptions; detecting type violations; search based on set intersection; and maintaining multiple, overlapping world-views at once in the same KB.*"

The planner algorithm exploits this capabilities and faces the service composition task as a planning problem of achieving the course of actions that from a given before context (the current situation) lead to a concrete after context.

3.2 The planning algorithm

Taking advantage of service versatility, systems for Ambient Intelligence could respond to whatever the needs on the basis of the available services and devices. In this context, needs are considered to be the desire of performing actions on objects. The innovative contribution of this work is the description of services as actions to be performed on objects. To this end, the idea of Hierarchical Task Networks (HTN) is adapted to work with actions, instead of tasks.

The actions that can be performed by a system are determined by the devices and services available at each moment in time. Those actions that cannot be performed, due to the lack of services providing the functionality, are named here as **non-feasible** actions. Whenever the system demands the execution of a non-feasible action, the planner comes into scene.

As listed underneath, the `Planning` algorithm starts with an empty plan, the Π plan, to be completed with the list of actions, provided by services. This course of actions are intended to emulate the demanded non-feasible action. The course of actions is provided as a set of actions performed on objects, A and O respectively, and the results R of accomplishing such actions.

Let's imagine a system where the biometric identification service was not available, while services such as the face recognition or face detection were. In order to figure out the alternative set of actions that could simulate the same functionality, the planner is instantiated with the following arguments, namely,

Algorithm 1 Planning(Π, A, O, R)

1: $\pi = (A, O, R)$
2: **if** A is non-feasible **then**
3: get all the actions $A = (a_1, a_2, ..., a_n)$ that have the same result A
4: **while** a_i is non-feasible **do**
5: delete a_i from A
6: **end while**
7: **while** only doable actions a_i does not have an equivalent target object **do**
8: list all the objects $Objects = (o_1, o_2, ..., o_n)$ of action a_i
9: check if those o_i are equivalent to or can be O
10: **end while**
11: Recursively call $\pi = Planning(a_i, o_i, resultOf\ a_i)$
12: **end if**
13: Add π to Π
14: Return Π

the plan to be executed (initially empty), the non-feasible action to be simulated, the object on which the action is to be performed, and the result of performing that action, something like:

```
Planning(plan, {biometricIdentification}, {biometricFeature}, {person
identity})
```

Where the {`biometricIdentification`} is the action to be simulated by means of the available actions in the system. The {`biometricFeature`} is the object on which the action is to be performed. Notices that instances of `biometricFeature` could be a face, an iris, or a fingerprint. Finally the {`person identity`} is the outcome of performing a `biometricIdentification` on a `biometricFeature`. For a simulated context with a particular set of devices that provide a set of services, the resulting plan to the previous request is the following list of ternary elements, to be executed bottom-up:

1. {face recognition} {capture result of a recording image device} {person identity} using the available `face recognition system`.
2. {detecting face} {capture result of a capturing biometric feature} {image file} using the available `face detector system`.
3. {capturing face} {thing} {image file} using the available `video camera service`.

4 Implementation Details

The Belief-Desire-Intention model (BDI) has proved to be a powerful framework for building rational agents [19], mainly because it offers the possibility to describe the agent behavior depending on the goals to be met, the knowledge it holds about the context, and a set of plans that assist the agent towards its goals. Indeed, the BDI model is intended to reproduce the process carried out when people make decisions to achieve a certain goal and perfectly couples with the action planning proposed here.

Adopting such an approach allows the Ambient System to describe its goals in terms of general goals, and expects the BDI system to concrete how these goals are to be achieved. Here is where the common-sense planner comes into action by receiving

the general description of the goal to be achieved, the object of that goal, and the expected result from achieving the goal. On the basis of this information, the planner provides the intelligent agent with the list of services that have to be instantiated.

The multi-agent system implementation is supported on Jadex [20], an agent-oriented reasoning engine for rational agents. Instead of using formal logic descriptions, Jadex proposes the use of two commonly known languages, such as Java and XML. The BDI agent is modeled by mapping the concepts of beliefs into Java objects, while desires and intentions are mapped into procedural recipes coded in Java that the agent carries out in order to achieve a goal.

Despite the fact that agents have been a widely adopted solution for supporting service composition in the field of Web Services, its implementation in other fields of distributed systems has not been so prolific. On the contrary to Web Services, that share a communication protocol such as SOAP, services deployed in pervasive environments are rarely capable of inter-working with the rest of services, mainly due to the heterogeneity of their implemented protocols. This drawback is tackled in this work by means of a middleware technology, the ZeroC ICE[21], which makes the communication process transparent.

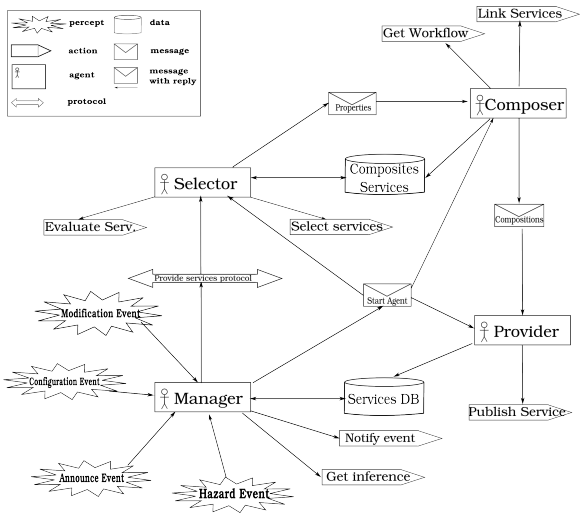

Fig. 2. System overview diagram

Figure 2 depicts an overall view of the Multi-Agent System (MAS), in its role of observing and regulating the Ambient Intelligence context. The MAS is composed of four agents, known as *Manager, Selector, Composer*, and *Provider*. So in order to be context-aware, the *Manager* agent subscribes to those channels, where services publish their status information and accept requests. Hypothetically, a presence sensor service would publish the presence detection to one of these channels whenever noticed. Automatically, the `Manager` agent, which has been implemented to react to this sort of events, notifies the `Selector` agent, who is the one that knows how to manage the situation. The following code, extracted from the description of the `Selector` agent,

states that whenever an `unauthorisedPresence` event has been triggered, one of the goals to be achieved is the `intruder_ identification`.

```
    <!-- O2. Intruder identification -->
    <achievegoal name="intruder_identification">
      <parameter name="unauthorisedPresence" class="Event">
      <bindingoptions>beliefbase.eventTypes</bindingoptions>
      </parameter>
      <unique/>
    </achievegoal>
 <!-- Plan intended to accomplish a biometric ID of the intruders -->
    <plan name="get_biometric_ID">
      <body class="GetBiometricIDPlan"/>
      <trigger>
        <goal ref="intruder_identification"/>
      </trigger>
    </plan>
```

The novelty here is that plans are not hard-coded in the `Selector` plans code, but on the contrary they are stated as requests to the planning module. For instance, whenever the goal `intruder_identification` is dispatched, the `get_biometric_ID` plan is used to accomplish the goal. In the plan Java code, a sentence like the following can be found:

```
getPlan("{identification}", "{biometric feature}", "{person identity}")
```

The main strength of this approach is grounded on the generality of the plans managed by the `Selector` agent. Finally,the `Composer` agent is in charge of instantiating each of the actions composing the plan, and the `Provider` agent is in charge of publishing this new composite service as an available service of the system.

5 Conclusions

The work presented here provides a solution to the self-sufficiency issue demanded by systems for Ambient Intelligence. To this end, this work proposed a comprehensive solution intended to provide automatic service composition, achieved by means of a commonsense planning strategy.

Combining a Belief, Desire, and Intention approach with the Scone system poses the basis for implementing an action planning, capable of solving the problem of automating the service composition task. The use of a middleware layer places an abstraction layer in between the heterogeneous services and the Ambient System supervising the environment, in such a way that service instantiation and supervision is achieved by simply supervising the communication channels where information is published and from where services receive invocations.

References

1. Ducatel, K., Bogdanowicz, M., Scapolo, F., Leijten, J., Burgelman, J.C.: ISTAG: Scenarios for ambient intelligence in 2010. Technical report, ISTAG (2001)

2. Wilensky, R.: Planning and Understanding. Addison-Wesley, Reading, MA (1983)
3. Amigoni, F., Gatti, N., Pinciroli, C., Roveri, M.: What planner for Ambient Intelligence applications? IEEE Transactions on Systems, Man, and Cybernetics, Part A **35**(1) (2005) 7–21
4. Erol, K., Hendler, J., Nau, D.S.: HTN planning: Complexity and expressivity. In: In AAAI-94. (1994)
5. Villanueva, F.J., Villa, D., Santofimia, M.J., Moya, F., Lopez, J.C.: A framework for advanced home service design and management. Computers in Education, International Conference on **0** (2009) 1–2
6. Mueller, E.T.: Commonsense Reasoning. Morgan Kaufmann (2006)
7. Hommel, B., Musseler, J., Aschersleben, G., Prinz, W.: The theory of event coding (TEC): A framework for perception and action planning. Behavioral and Brain Sciences **24** (2001) 849–878
8. Bach, K.: Actions are not events. Mind **89** (1980) 114–120
9. Hyman, J.: Three fallacies about action. In: 29th International Wittgenstein Symposium. (2006)
10. Georgeff, M.P.: A theory of action for multiagent planning. In Bond, A.H., Gasser, L., eds.: Readings in Distributed Artificial Intelligence. Kaufmann, San Mateo, CA (1988) 205–209
11. W3C: OWL web ontology language home page (2009) Available online at: `http://www.w3.org/TR/owl-features/`. Retrieved March 15th, 2009.
12. W3C: Resource description framework (RDF) (2010) Available online at: `http://www.w3.org/RDF/`. Retrieved February 28th, 2010.
13. DiBernardo, M., Pottinger, R., Wilkinson, M.: Semi-automatic web service composition for the life sciences using the biomoby semantic web framework. J. of Biomedical Informatics **41**(5) (2008) 837–847
14. Cycorp, I.: The Cyc project home page (2008) Available online at: `http://www.cyc.com`. Retrieved on December 10th, 2008.
15. Fahlman, S.E.: The Scone knowledge-base project (2010) Available online at: `http://www.cs.cmu.edu/~sef/scone/`. Retrieved on February 28th, 2010.
16. Davidson, D.: The Essential Davidson. Oxford University Press, USA (2006)
17. Chen, W., Fahlman, S.E.: Modeling mental contexts and their interactions. In: AAAI 2008 Fall Symposium on Biologically Inspired Cognitive Architectures, Washington. (2008)
18. Fahlman, S.E.: Marker-passing inference in the scone knowledge-base system. In: First International Conference on Knowledge Science, Engineering and Management (KSEM'06), Springer-Verlag (Lecture Notes in AI) (2006)
19. Wooldridge, M.J.: Reasoning about Rational Agents. The MIT Press, Cambridge, Massachusetts (2000)
20. Pokahr, A., Braubach, L., Lamersdorf, W.: Jadex: A BDI reasoning engine. In: Multi-Agent Programming, Springer Science+Business Media Inc., USA (9 2005) 149–174 Book chapter.
21. ZeroC, I.: Ice home page (2008) Available online at: `http://www.zeroc.com/`. Retrieved December 20th, 2008.