A Distributed Architectural Strategy Towards Ambient Intelligence *

Maria J. Santofimia, Francisco Moya, Felix J. Villanueva,
David Villa, Juan C. Lopez

Computer Architecture and Networks Group. School of Computing Science
University of Castilla-La Mancha
{MariaJose.Santofimia,Francisco.Moya,Felix.Villanueva,David.Villa,
JuanCarlos.Lopez}@uclm.es

Abstract. This work reveals the benefits obtained from combining common-sense reasoning and multi-agent systems on top of a fully equipped middleware platform. The architecture here proposed is founded on the service composition paradigm, as the comprehensive solution to relieve users from being involved in system decision making. In this regard, the environment and domain understanding is emulated by the common-sense reasoning engine that supports the multi-agent system on the task of effectively accomplishing the actions that fullil the new arisen requirements.

1 Introduction

The vast majority of the literature on the field of systems for ambient intelligence concentrate their efforts on releasing mechanisms to gather information about users, match behavioral patterns, or predict user actions, requirements and needs. Nevertheless, not only users should be considered but also the environment itself, so as to obtain a comprehensive solution that covers the domain context objectives. This issue is not addressed in most solutions presented to date. In this regard, extending the user-centered view, in order to encompass the system services and intentions, arises as a key requirement to the ambient intelligence systems. An appropriate design of a middleware architecture suffices to support the achievement of this requirement. However, this is not enough for assuring the autonomous and intelligent behavior of ambient systems demand. This shortcoming motivates the need to endow the middleware with the capability to understand and reason about its context, as well as making decisions in reaction to events.

This paper main intention is to address some the emerging challenges in the seeking to develop self-managed systems for ambient intelligence. In these endeavors, this approach advocates for mechanisms that support the dynamic generation of behaviors on the basis of basic actions, which are the smallest

* This work has been funded by the Spanish Ministry of Industry under project CENIT Hesperia

units compounding services. An appropriate design of the middleware services reveals the great importance of having a framework supporting this task. This proposal counts on a distributed object-oriented framework for service design and modeling.

2 A combined strategy

The architecture here proposed, and depicted in Fig. 1 rests on a powerful middleware framework, that provides the upper layers with the structure, tools and services required to successfully accomplish their tasks. In [1] a distributed object-oriented framework (DOBS, Distributed Object Based Services) is proposed, so as to overcome the problems appearing when the use of certain services involve managing different protocols. Nonetheless, a detailed description of the middleware framework is out of the scope of this article.

The motivation behind choosing a multi-agent approach is twofold. Firstly, given the service oriented character of the middleware framework, the agent-based approach can be easily fit in the framework, adopting the shape of yet another middleware service. Secondly, autonomy and proactive features are inherent to agents. In addition to this, the BDI model of agency provides the goal-oriented character, required by the architecture here proposed.

Drawing these points together, it can be concluded that adopting a BDI agent infrastructure, such as Jadex [3] is the most promising solution to tackle the automation of the service composition task. Although Jadex provides its own middleware platform, it is modular enough to run on top of the DOBS framework, making the most of the aforementioned features.

The proposed multi-agent system solution consists of four agents. The manager agent supervises the events taking place in the application domain, by monitoring the available services, the event channels and the state of the services deployed on it. The DOBS framework provides a set of channels where services might be bound to publish or subscribe events. When gathering information about services, the manager agent simply sends a request to the appropriate channel and waits to collect the responses generated by the services listening to this channel. Services are univocally identified by means of the proxy concept [4], inherited from the middleware framework.

The manager agent is committed to assure a minimum level of functionality, overcoming service failures or disappearance. Furthermore, it is also committed to ensure the completeness of the generated composite services. Whenever some of these goals are dispatched, the other three agents are started by the manager, which also supplies them with information about the available services and their states.

Once started, the selector agent is basically intended to identify the services involved in the composition, performing this selection according to the set of properties that the manager agent has provided it with. It has to be remarked that these properties have been extracted from the model information system that came along with the middleware framework. Based on the UPnP

templates, services are characterized with property dictionaries. Therefore, the selector agent, borne on the common-sense knowledge provided by the OpenCyc [2] system, infers the set of services, out of the available ones, capable of fulfilling the compendium of stated properties. Once these services are selected, the composer agent receives from the selector agent the list of basic services involved in the composition. This agent is in charge of linking those services, so that the composite service behaves as an unique service. By means of a workflow, it is specified the order in which services are executed, along with the information flow. It might also be required to adapt data input or output in order to meet the required format. At the moment, this agent is constrained to basic workflows, although current efforts are aimed at implementing some artificial intelligent planning technique to support the workflow generation process. In any case, the generated workflow is used by the provider agent to instantiate those services and providing them with the required input, and forwarding outputs to services, as specified on the workflow. Finally, the provider agent deploys the service on the system in a transparent way. Apparently, the ambient environment remains unaware of the composition process that has been accomplished on the background.

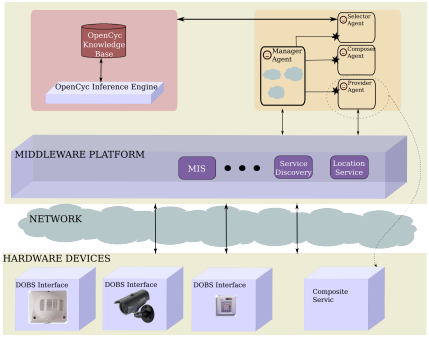

Fig. 1. System Overview

When describing the selector agent, we slightly pointed out the role played by common-sense knowledge provided by the OpenCyc system, on supporting the selection of services involved on the composition task. However, there are a number of issues, regarding this choice, that have to be exposed in arguing for a common-sense knowledge and reasoning system.

OpenCyc is the open source version of the Cyc Knowledge Base, which underlying philosophy advocates for applications capable of flexibly reacting to a variety of challenges. Modeling the domain specific knowledge might be sufficient for static systems, but it definitely fails to address the flexibility required on systems for ambient intelligence environments.

However, the domain specific knowledge plays an important role on the architecture here described. In this sense, the ontology and the model information

system inject semantic meaning to the messages exchange among agents, and set the vocabulary used. Nonetheless, the reasoning and inference capabilities of multi-agent system are scarce and mainly constrained to reason about their plans and goals, having to resort to an external reasoner tool to achieve broader reasoning capabilities. It can be concluded that domain specific knowledge yields poor and deficient when applied to dynamic contexts.

Although the need for counting on common sense knowledge to support the reasoning and inference appear to be evident, the arduous task of mapping this knowledge into a knowledge base reveals the suitability of disposing of more that twenty years of gathered knowledge. Furthermore, OpenCyc provides a wide range of tools that dramatically eases the process of integration and combination with multi-agent systems. The well documented Java API supports the FIPA-OS agent integration, by providing a set of methods that make effective the communication between the OpenCyc server and the multi-agent system.

3 Conclusions

The main drawback encountered when trying to develop systems for ambient intelligent lies on the vast amount of knowledge required when supporting systems with intelligent behaviors. Despite the availability of reasoning tools capable of dealing with domain knowledge, they reveal futile without the common sense knowledge support.

This article has sought to justify the importance of automatic service composition on supporting systems for ambient intelligence. In seeking to accomplish the automation of the service composition task, this approach draws on a combined multidisciplinary approach of multi-agent systems and common sense knowledge. As constituent components of a broader architecture, these are integrated in a distributed middleware architecture that provides them with the groundings to support their endeavors towards intelligent environments. Nevertheless, this does not represent a silver bullet to achieving more intelligent ambient.

References

1. F.J. Villanueva, D. Villa, F. Moya, M.J. Santofimia, J.C. Lpez, *A framework for advanced home service design and management* `http://arco.esi.uclm.es/es/node/418`, IEEE International Conference on Consumer Electronics , Las vegas, EEUU, January 26, 2009.
2. Inc Cycorp. The opencyc project home page, 2008. Available online at `http://www.opencyc.org`. Retrieved on December 10th, 2008.
3. Alexander Pokahr, Lars Braubach, and Winfried Lamersdorf. Jadex: A bdi reasoning engine. In J. Dix R. Bordini, M. Dastani and A. El Fallah Seghrouchni, editors, Multi-Agent Programming, pages 149174. Springer Science+Business Media Inc., USA, 9 2005. Book chapter.
4. Inc. ZeroC. Ice home page, 2008. Available online at: `http://www.zeroc.com/`. Retrieved December 20th, 2008.