Showcase of Combining the Meeting Portal with JADE-based Multiagent System

Aki Vainio, Joseph Awali
University of Vaasa
Vaasan yliopisto, PL 700
65101 Vaasa, Finland
+35850 544 2669
aki.vainio@uwasa.fi, jawali2@hotmail.com

Kimmo Salmenjoki
Seinäjoki University of Applied Sciences
Keskuskatu 34
60100 Seinäjoki, Finland
kimmo.salmenjoki@seamk.fi

Keywords

Multi Agent Systems, Artificial Intelligence, Applications of Ambient Intelligence, Case Studies, System Design and Architecture for Ambient Intelligence

Abstract

Bringing ambient intelligence calmly in to daily routines is complicated. Doing so in contexts where several people are involved, is even more complicated. This paper takes a look at one way of implementing an ambient intelligence application which needs to understand the contexts for several people and locations simultaneously. The proposed system is based on the use of agents as a method of distributed problem solving and the pilot application will be an intelligent meeting room with capabilities for videoconferencing used through a portal.

Introduction

This paper describes design principles for a system designed to help organize meetings through understanding the context and being able to work with personal agents representing the users of the meeting environment. Portals are becoming more and more important to organizations, which have an ever-increasing need to provide employees, partners, and customers with an integrated view of applications, information, and business processes. Our Meetingroom Portal meets these needs, allowing organizations to build portals that combine functionality and resources into a single interface while enforcing organizational policies, processes, and security requirements, and providing personalized views of information to end users [15].

Russell and Norvig [13] use the concept of agent as a tool for analyzing entities. They define agent as something with an environment, perceptions of that environment, actions which it can perform on that environment and goals which it wishes to achieve. In operational and legal sense, an agent is someone or – in this case – something who or which is authorized to work on behalf of someone or something else.

Multiagent Systems

"[Distributed artificial intelligence] is the study, construction, and application of multiagent systems, that is, systems in which several interacting, intelligent agents pursue some set of goals or perform some set of tasks" (Weiss, [17]). Although multiagent systems originated from the field artificial intelligence, they are now becoming the next step in software engineering. This is based on the notion that agents are the next evolutionary step from objects. Instead of working with objects, which are static, the systems are based on active agents. The agents can solve problems by distributing the problem among other agents, which are programmed to deal with certain types of subproblems. Because the architecture defines the way the agents interact, the person programming the individual parts of the system need not know how the other parts of the system work. Therefore programming becomes a simpler task, because the designer of an agent which works within the system can fully concentrate on the design of that one agent.

This functionality is achieved through the use of a communications between the agents. The communication requires a standardized method of forming the messages and an ontology, which is required as a form of giving semantics to the messages, see [16].

Standards are an important tool in using multiagent systems. The JADE framework used in this paper follows the FIPA standard for the architecture and the ontologies used are described using OWL. By using standards, interoperability between systems is easier to achieve.

In order to understand what a multiagent system is, one must understand what an agent is.

Software Agent

Singh and Huhns [11] define software agent as "an active computational entity that

- has persistent identity;
- can perceive, reason about, and initiate activities in its environment;
- can communicate with other agents, including humans."

Russell and Norvig [13] analyze agents by looking at their performance measures, environments, actuators and sensors (collectively known as PEAS), which gives a nice conceptual framework for any ambient and intelligent embedded system applications.

For example, an earthworm has the higher goal of seeing to it that its genes are passed on to the next generation. Its performance measure would therefore be the number of offspring it has successfully brought into the world (many species would also try to protect those offspring). The environment would be the soil its living in. Its actuators would be its muscles, through the use of which it would try to find sustenance, mate and occasionally relieve itself. The earthworm's sensors are its sense of touch and its sight, both of which are quite limited.

A Texas Hold 'Em agent would measure its performance through winnings, its environment would include the cards in play and the other players (including their histories for learning purposes), its actuators would consist of messages through which it would either check, fold or raise, and its sensors would consist of the mechanism it uses to receive messages.

An agent's basic structure and its relation to the environment are presented in figure 1. For most software agents, the sensors are limited to messages received from the environment, sensors and from other agents. Usually the actuators of these agents are also limited to sending messages. They usually have also some other (non-agent related) functionality, but usually that functionality happens outside of the environment. For example, a wrapper agent used to integrate a legacy system into the multi-agent environment would receive requests for information through messages and would return the results through messages. The legacy system itself would be outside of the environment.

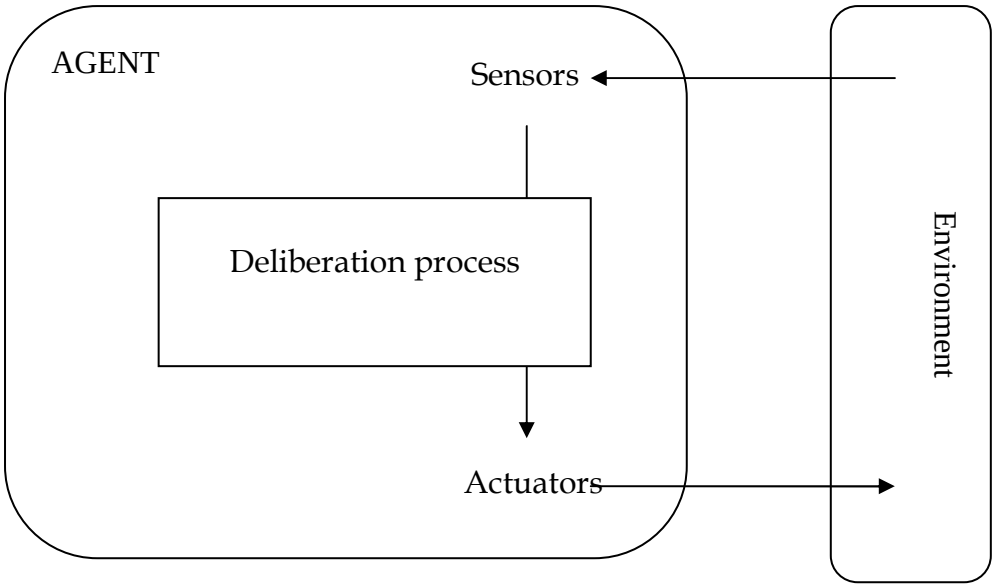

Fig. 1: Schema of the task environment of an agent (based on [12])

FIPA

FIPA stands for The Foundation for Intelligent Physical Agents. However, the physical presence of the agent FIPA specifications deal with is restricted to the memory of a computer. FIPA was an independent organization until 2005 when it was replaced by an IEEE committee by the same name.

Agent systems are highly dependent on standardization, because otherwise producing compatible systems of agents would be extremely difficult. FIPA has taken the role of the standardization, although in many cases, other standards, such as those specified by W3C are used, especially when dealing with web environments or semantics. Standardization is obviously problematic, as standards are always the result of a slow process, which leads to a compromise based on the needs of participants of the process. Not standardizing would on the other hand lead to the loss of many of the key benefits of using agents, primarily the ability to work with countless other agents.

FIPA is also working on finding a solution for the interoperability of web services and agents. The combination of these two technologies could be very powerful and some estimate that soon most user agents on the web are actually representing software agents.

Architecture

The purpose of the FIPA architecture is to ensure interoperability and reusability of code. If two systems need to interact, but are based on different technologies, the elements of the architectures, which must have common functionality, must be identified and codified.

FIPA abstract architecture encompasses communications between the agents and how agents find the other agents they need otherwise known as directory services. In this context agent communications include both message transport and the encoding schemes used to form and interpret messages.

Central to the FIPA architecture is obviously the agent. Agents communicate through speech acts, which are represented by messages. The messages are encoded in an agent communications language. The platform includes a number of agents, which act as services, on which the other agents are dependent on.

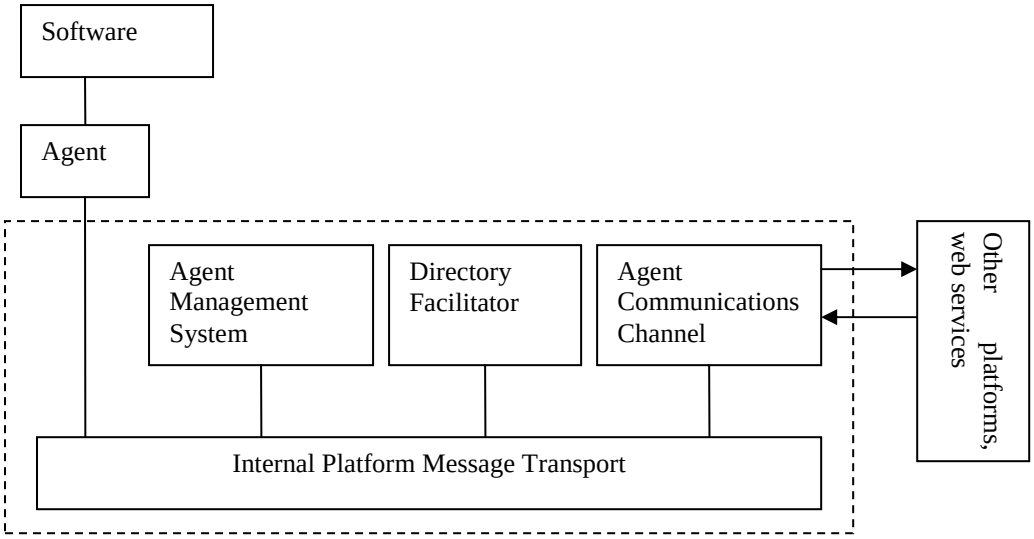

Fig. 2: FIPA Agent Platform (based on [1])

All agents are listed on at least one service-directory-service or agent-directory-service. These are services through which agents can find other agents they wish to interact with. These directories include the agent's unique name (generally dependent on the platform in order to avoid namespace issues in multiplatform environment) and locator, which is a description of how to contact the agent. Also, the entries include any other information the agent wishes to register, such as the services it provides, possible costs, restrictions and so forth.

Another mandatory service is the message-transport-service, which handles transporting messages between agents. It has access to a number of transport methods (stream, datagram, etc), from which it chooses the most appropriate one for the current task. Often the system includes another agent for communications with other agent platforms.

JADE

JADE (Java Agent DEvelopment) framework is a FIPA-compliant framework for multiagent systems. As with any software framework, the object is to simplify the development a software system. JADE has been written in Java and JADE agents are Java programs, which inherit agent functionality from the JADE API.

The platforms are based on Java Virtual Machines (Java VM) and communications by Remote Method Invocation (RMI) between them. Within a single VM, event signaling is used. Each VM is a container for agents and acts as a concurrent runtime environment for agents using a thread for each agent, as well as separate threads for system services. One of the containers represents the platform to other systems outside the platform, such as other agent platforms. This container also includes the agent management system and other critical services.

The JADE framework loosens the FIPA agent communication requirements somewhat by using a lighter transport of objects instead of strings within the platform. However, between platforms, communication is still handled by using FIPA compliant string format. This conversion is done by the agent communications channel, so that the agent implementers only need to deal with one Java class of messages.

JADE also includes a Remote Monitoring Agent (RMA), through which a GUI is provided. RMA acts as an agent in the sense that it communicates with the agent management system just as an agent would.

Coordination in a Multiagent System

Agents are built for a purpose and agents can communicate. The ability to communicate is a property of the agent which makes it more capable of filling its purpose. The idea of communication is to let the agents within a system to coordinate their behavior in such a way that it's beneficial for the system or the individual agents.

Generally coordination is a method of using the resources more efficiently and avoiding situations which would endanger the workings of the system. In coordinating behavior, it's important to make a distinction between cooperation and competition. Cooperating agents follow a common plan, which is devised either centrally or in a distributed manner; competing agents negotiate with other agents to gain a better position for themselves.

For example, personal agents which are trying to set a meeting, coordinate their efforts by distributed planning, in other words, cooperation. Each of the agents will find suitable times from their schedules, possibly taking into account locations and other such matters of context. An agent responsible for calling the meeting will than make a decision based on that information, which hopefully suits as many of the participants as possible.

On the other hand, if the meeting rooms are in heavy use, an agent responsible for setting the meeting might have to negotiate for a room. Perhaps the organization has given each department or project a number of tokens which can be used to "buy" meeting time in an auction or the agent might have to give up a reservation on another time to be able to use a certain room at a certain time.

The Meeting Portal

Meeting room/MyHome portal is an interface to test home-based multimedia. Besides the everyday applications, new approaches of semantic web, wireless technologies and context based inference will be tested in the laboratory setting. The portal can be used by several persons so that it gives the personalized and adapted view for each person differently. Also issues around home devices, location, and embedded programming could be built in to this portal. The main emphasis is to develop software solutions for modern service based ICT applications, one type of which is the previously described Multiagent Systems.

Although the idea of a meeting is not very complex, the situation can turn very complicated once there are several locations involved. Keeping such a meeting under control can require a lot of coordination. In this case, much of the responsibility for the coordination has been placed on a software agents rather than people.

For accessing the meeting room with the portal we have to coordinate activities in space and time. Several users will have access to the portal, and their requests and responses will be displayed and stored via the Meeting room portal. When some actions are taken by the users, the role of the agents in the MAS is to see to the user's personal needs (via the personal agents) or to coordinate and assist the activities in the space via the shared context data and the MAS system activities.

We have implemented only a few cases, where demonstrative examples of use cases in the meeting-room. These examples show how the portal and MAS can coexist to provide the user some user centric, context aware services.

Context and Ontologies

Understanding the context in the meeting room is an important function of the system, as it enables the agent to make assumptions on which to base decisions. Important, but hard to handle, contextual problems include those of space and time. Other contextual information includes participants and so forth.

The role of context also relaxes the user interfacing of the portal and connects the activities of agents and humans together.

The two aspects needed for describing the domain of agent activities are covered by the ontologies in our case describing the organization and operation of agents in the laboratory space by describing the knowledge structures and their relations with rules of operation. For this purpose we will use the COBRA system structure [2] directly. COBRA uses ontologies both for Semantic Web languages both to express context ontologies and to reason about contexts. For this purpose OWL (Web Ontology Language) is used.

Space information can be divided into places and agents. Places include such classes as Place, AtomicPlace, CompoundPlace, Room, Building, Campus and so on. Their properties include coordinates, spatial relations to other places (ie. which place subsumes which other places and vice versa), restrictions on access and so forth. Agent classes include Agent, Person, SoftwareAgent, Role, SpeakerRole, other roles as needed and action related classes. Agent properties include personal information and the agent's role at the meeting. Other classes used involve the agent's relation to places (in other words, where the agent is currently located) and agent's activity (in this portal we are only interested in whether the agent is participating or will participate in an event or meeting and what his role in that event or meeting will be).

Understanding contexts is an important aspect of pervasive computing as it enables the agents to apply calmness to the technologies used. By being able to understand context, the user interfaces can be changed to better fit the situation. For example, if there is a videoconference with multiple locations participating, the agents responsible for the views can emphasize the videostream from the source where the current presentation is going on.

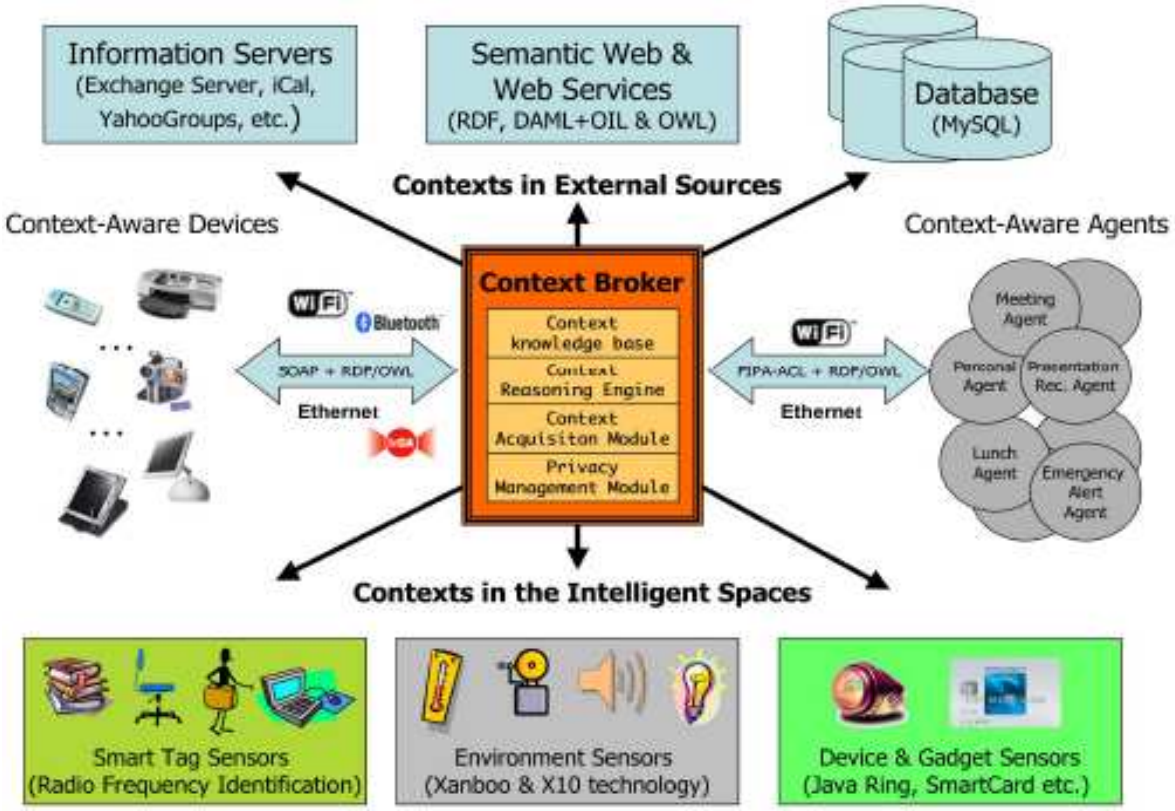

Fig. 3: Context Broker architecture [2]

In the meeting room example, understanding context means also understanding the surroundings. As shown in figure 3, the context broker can communicate with the devices in the meeting room area or areas. Depending on the capabilities of the devices, this can mean setting lighting, switching microphones on and off and so forth.

Users

When the users register to the portal, their identity and preferences will be recorded. Based on this user profile, the various activities of the meeting room will be activated in the portal as requested by the user. The core user services will be provided by the user's personal agent in the shared Multi Agent System MAS context of COBRA on JADE. Also, if there are shared contexts between the users, the MAS can assist the users in sharing the meeting-room space and also possibly some other areas of interest shared by the different users.

Privacy

Users are profiled according to their role and activities in the meeting-room. Together with this basic information of the user, the actions of the user on the portal get recorded in the portal database.

To support the user via the agents, the MAS will deal with the user profile and context data. This assumes of course that the user has confidence in the system and is willing to share his/her context with the other users and the MAS. Users can provide privacy rules for their personal information and context.

As shown in figure 3, the context broker uses – if available – information from devices carried by the users. This can present problems if the users do not trust the system and are unwilling to disclose the information. In these cases the context broker can still use the information from other sources and use assumptions. For example, if there are three unidentified persons in a meeting room, where there are supposed to be three participants for a meeting, the system can assume that those three are the persons are the participants.

On the other hand, if the user feels that the context broker is trustworthy and can protect his or her privacy adequately, the user can even take an extra step and allow the context broker a limited control over his or her devices. For example, the context broker could switch cellular phones to vibrate when the meeting starts and back when the meeting ends. This would obviously require an interface through which the context broker can access the appropriate controls.

Conclusions

In this paper we have presented the basic ideas behind multiagent systems and the principles through which the concept can ease the development of systems which require integration of many different parts. Their usability in an ambient system is demonstrated by a real-world application of a Meetingroom Portal, which is largely dependent on the context broker agent (COBRA) and its interaction with other agents, devices, users and services available to it. This meeting-room portal will be used as test bed for providing wireless services with digital media and physical location in the Technobothnia laboratory in Vaasa.

References

[1] F. Bellifemine, A. Poggi, G. Rimassa. JADE – A FIPA-compliant agent framework. (1999)
[2] H. Chen. An Intelligent Broker Architecture for Pervasive Context-Aware Systems. (2004)
[3] E. Cortese, F. Quarta, G. Vitaglione. Scalability and Performance of JADE Message Transport System. (2002)
[4] FIPA 2000 Specifications. http://fipa.org/repository/standardspecs.html (2000)
[5] The Foundation for Intelligent Physical Agents. http://www.fipa.org. (2007)
[6] S. Frankin and A. Graesser. Is it an Agent, or just a Program?: A Taxonomy for Autonomous Agents [online]. Available: http://www.msci.memphis.edu/~franklin/AgentProg.html. (1996)
[7] JADE Framework. jade.tilab.com (2007)
[8] C. Kindle. Agent Systems' Influence on Information Retrieval. (2003)
[9] Y. Labrou, T. Finin and Y. Peng. Agent Communication Languages: The Current Landscape. IEEE Intelligent Systems, vol. 14, no. 2, p. 45–52. (1999)
[10] M. Luck, R. Ashri and M. D'Inverno. Agent-Based Software Development. (2004)
[11] L. Padgham and M. Winikoff. Developing Intelligent Agent Systems – A Practical Guide. (2004)
[12] J. Pitt, F. Bellifemine. A Protocol-Based Semantics for FIPA'97 ACL and its implementation in JADE. (1999)
[13] S. Russell and P. Norvig. Artificial Intelligence – A Modern Approach, 2nd edition. (2003)
[14] M.P. Singh and M.N. Huhns. Service-Oriented Computing – Semantics, Processes, Agents. (2005)
[15] UWasa Meetingroom Portal. http://ttwin.techno.uwasa.fi/Meetingroom. (2007)
[16] C. van Aart, G. Caire, R. Pels, F. Bergenti. Creating and Using Ontologies in Agent Communications. (2002)
[17] G. Weiss (ed.). Multiagent Systems – A Modern Approach to Distributed Artificial Intelligence. (1998)