This document is published in:

"Andre Ponce de Leon F. de Carvalho, et al. (eds.) (2010) *Distributed Computing and Artificial Intelligence: 7th International Symposium*. (Advances in Intelligent and Soft Computing, 164) Springer, pp. 275- 282.
Doi: http://dx.doi.org/10.1007/978-3-642-14883-5_36"

An Architecture to Provide Context-Aware Services by Means of Conversational Agents

David Griol, Nayat Sánchez-Pi, Javier Carbó, and José M. Molina

Abstract. In human-human interaction, a great deal of information is conveyed without explicit communication. This context information characterizes the situation of the different entities involved in the communication process (users, place, environment and computational objects). In this paper, we present an agent-based architecture that incorporates this valuable information to provide the most adapted service to the user. One of the main characteristics of our proposal is the incorporation of conversational agents handling different domains and adapted taking into account the different users requirements and preferences by means of a context manager. This way, we ensure a natural communication between the user and the system to provide a personalized service. The implementation of our proposed architecture to develop and evaluate a context-aware railway information system is also described.

1 Introduction

Ambient Intelligence (AmI) emphasizes on greater user-friendliness, more efficient services support, user-empowerment, and support for human interactions.To achieve this goal it is necessary to provide an effective, easy, safe and transparent interaction between the user and the system. To do so, in the last years there has been an increasing interest in simulating human-to-human communication, employing the so-called conversational agents [5]. Conversational agents, which enhance agents with computational linguistics, have became a strong alternative to provide computers with intelligent communicative capabilities.

In the literature, there are several approaches developing platforms, frameworks and agents applications for offering context-aware services [4, 1, 6]. The processing of context is essential in conversational agents to achieve an adapted behaviour and also cope with the ambiguities derived from the use of natural language. For

David Griol, Nayat Sánchez-Pi, Javier Carbó, and José M. Molina
Group of Applied Artificial Intelligence (GIAA), Computer Science Department,
Carlos III University of Madrid
e-mail: {david.griol,nayat.sanchez,javier.carbo}@uc3m.es
josemanuel.molina@uc3m.es

instance, context information can be used to resolve anaphoric references, to take into account the current user position as a data to be used by the system or to decide the strategy to be used by the dialog management module by taking into account the specific user preferences [8]. For this reason, the result of the interaction can be completely different depending on the environment conditions (e.g. people speaking near the system, noise generated by other devices) and user skills. This information usually describes the user state (e.g. communication context and preferences) and the environment state (e.g. location and temporal context). One of the most important goals of our work is to combine both sources of information to carry out the system adaptation.

For this reason, in this paper we describe an agent-based architecture that provides context awareness and situation sensitivity to discover, process and provide context aware services adapted to users' location, preferences and needs. In order to ensure a natural and intelligent interaction between the user and the system, advanced conversational agents have been developed including a context manager module to facilitate the provisioning of personalized services similar to human-human communication. We also provide a complete implementation of our architecture for the provisioning of personalized services to users in a railway information system and evaluate the influence of context information in the operation of the dialog model.

2 Our Proposed Multi-agent architecture

The proposed agent-based architecture manages context information to provide personalized services by means of users interactions with conversational agents. As it can be observed in Figure 1, it consists of five different types of agents that cooperate to provide an adapted service. *User agents* are configured into mobile devices or PDAs. *Provider Agents* are implemented by means of *Conversational Agents* that provide the specific services. A *Facilitator Agent* links the different positions to the providers and services defined in the system. A *Positioning Agent* communicates with the ARUBA positioning system [7] to extract and transmit positioning information to other agents in the system. Finally, a *Log Analyzer Agent* generates user profiles that are used by Conversational Agents to adapt their behaviour taking into account the preferences detected in the users' previous dialogs.

Eight concepts have been defined for the ontology of the system. The definition is: *Location* (*XCoordinate* int, *YCoordinate* int), *Place* (*Building* int, *Floor* int), *Service* (*Name* String), *Product* (*Name* String, *Characteristics* List of Features), *Feature* (*Name* String, *Value* String), *Context* (*Name* String, *Characteristics* List of Features), *Profile* (*Name* String, *Characteristics* List of Features), *DialogLog* (*Log* List of Strings).

Our ontology also includes six predicates with the following arguments: *HasLocation* (*Place*, *Position*, and *AgentID*), *HasServices* (*Place*, *Position*, and *List of Services*), *isProvider* (Place, Position, AgentID, Service), *HasContext* (*What*, *Who*),

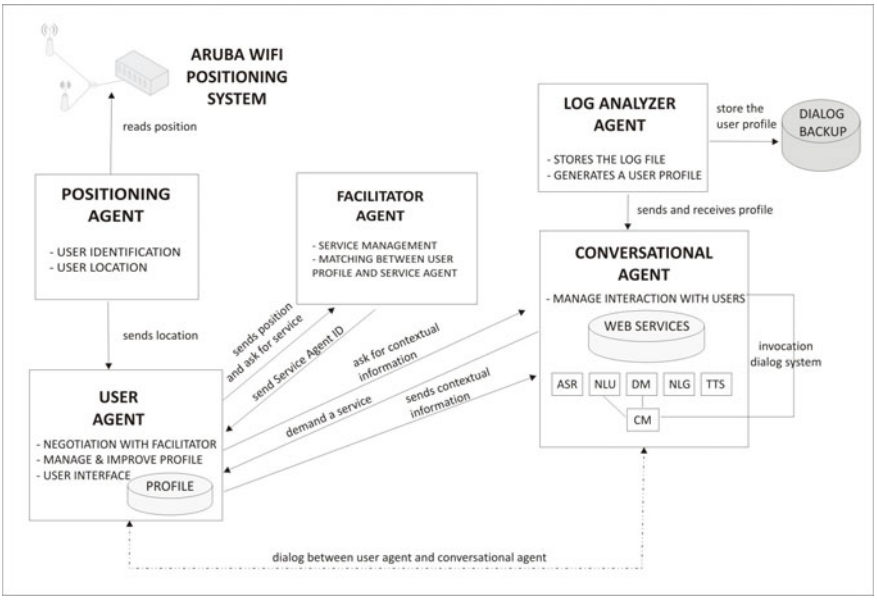

Fig. 1 Schema of the proposed multi-agent architecture

HasDialog (*DialogLog* and *AgentID*), *HasProfile* (*Profile* and *AgentID*), and *Provide* (*Product* and *AgentID*).

The interaction with the different agents follows a process which consists of the following phases:

1. The ARUBA positioning system is used to extract information about the positions of the different agents in the system. This way, it is possible to know the positions of the different User Agents and thus extract information about the Conversational Agents that are available in the current location.
2. The Positioning Agent reads the information about position (coordinates x and y) and place (*Building* and *Floor*) provided by the ARUBA Positioning Agent by reading it from a file, or by processing manually introduced data.
3. The Positioning Agent communicates the position and place information to the User Agent.
4. Once a User Agent is aware of its own location, it communicates this information to the Facilitator Agent in order to find out the different services available in that location.
5. The Facilitator Agent informs the User Agent about the services available in this position .
6. The User Agent decides the services in which it is interested.
7. Once the User Agent has selected a specific service, it communicates its decision to the Facilitator Agent and queries it about the service providers that are available.

8. The Facilitator Agent informs the User Agent about the identifier of the Conversational Agent that supplies the required service in the current location.
9. The User Agent asks the Conversational Agent for the required service.
10. Given that the different services are provided by context-aware Conversational Agents, they ask the User Agent about the context information that would be useful for the dialog. The User Agent is never forced to transmit its personal information and preferences. This is only a suggestion to customize the service provided by means of the Conversational Agent.
11. The User Agent provides the context information that has been required.
12. The conversational agent manages the dialog providing an adapted service by means of the context information that it has received.
13. Once the interaction with the Conversational Agent has finished, the Conversational Agent reads the contents of the log file for the dialog and send this information to the Log Analyzer Agent.
14. The Log Analyzer Agent stores this log file and generates a user profile to personalize future services. This profile is sent to the Conversational Agent.

As stated in the introduction, a conversational agent is a software that accepts natural language as input and generates natural language as output, engaging in a conversation with the user. To successfully manage the interaction with the users, conversational agents usually carry out five main tasks: automatic speech recognition (ASR), natural language understanding (NLU), dialog management (DM), natural language generation (NLG) and text-to-speech synthesis (TTS). These tasks are usually implemented in different modules.

In our architecture, we incorporate a Context Manager in the architecture of the designed conversational agents, This module deals with loading the context information provided by the User and Positioning Agents, and communicates it to the different modules of the Conversational Agent during the interaction.

To manage context information we have defined a data structure called *user profile*. Context information in our user profile can be classified into three different groups. *General user information* stores user's name and machine identifier, gender, prefered language, pathologies or speech disorders, age, *Users Skill level* is estimated by taking into account variables like the number of previous sessions, dialogs and dialog turns, their durations, time that was necessary to access a specific web service, the date of the last interaction with the system, etc. Using these measures a low, medium, normal, high or expert level is assigned. *Usage statistics and preferences* are automatically evaluated taking into account the set of services most required by the user during the previous dialogs, date and hour of the previous interactions and preferred output modality.

The free software JADE (Java Agent DEvelopment Framework)[1] has been used for the implementation of our architecture. It was the most convenient option as it simplifies the implementation of multi-agent systems through a middle-ware that complies with the FIPA specifications and through a set of graphical tools that

[1] http://jade.tilab.com/

supports the debugging and deployment phases. The agent platform can be distributed across machines and the configuration can be controlled via a remote GUI.

3 Case Study: A Context-Aware Railway Information System

We have applied our context aware methodology to design an adaptive system that provides information in natural language about train services, types, schedules, and fares. The compiled code of this application can be executed and is freely available through our website. The requirements for the task have been specified by taking into account the ontologies defined for this task in the DIHANA project [3]. Users can ask for information about *Hour*, *Price*, *Train-Type*, *Trip-Time*, and *Services*. They also can provide task-independent information like *Affirmation*, *Negation*, and *Not-Understood*. The attributes needed by the system to answer to the different user queries are *Origin*, *Destination*, *Departure-Date*, *Arrival-Date*, *Ticket-Class*, *Departure-Hour*, *Arrival-Hour*, *Train-Type*, *Order-Number*, and *Services*. The system responses can be classified into the following categories: *Opening*, *Closing*, *Not-Understood*, *Waiting*, *New-Query*, *Acceptance*, *Confirmation*, *Rejection*, *Question*, and *Answer*. A total of 51 different system actions were defined taking into account the information that the system provides, asks or confirms to the user.

A set of scenarios were manually defined to cover the different queries to perform to the system including different user requirements and defining one or several objectives (e.g., to obtain timetables and prices given a specific origin, destination and date). An example of these scenarios is as follows:

```
User name: John Smith
Location: Atocha Station
Date and Time: 2009-05-01, 9:00am
Device: PDAQ 00-18-41-32-0B-59
Objective: To know timetables and prices to Valencia
```

The first step is to execute the different agents using the JADE platform. Then the information about the position is sent from the Positioning Agent to the User Agent as shown in Figure 2. In the figure, the Positioning Agent is called Sensor Network and the name of the User Agent is John Smith. This figure shows the message that is sent by the Positioning Agent and received by the User Agent and how it sets the new position values.

In the next phase, the User Agent John Smith looks for the Facilitator Agent and ask it about the services that are available in the user's location. The following phase consists of selecting the service. From the list of two services that are available in the current location (TrainTicket and TrainInfo), one of them has to be selected. The service TrainTicket is selected and the Facilitator Agent answers the user's query by informing about the AgentID of the Provider Agent which provides the required service. Once the User Agent knows how to contact with the Conversational Agent that supplies the TrainTicket service, it sends a query to be provided with this service. Figure 3 shows this query to the Conversational Agent, called in the figure "Dialog System" Agent.

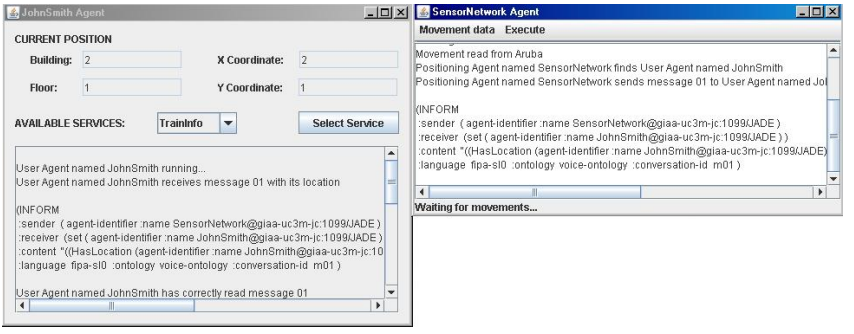

Fig. 2 Positioning information sent from the Positioning Agent to the User Agent

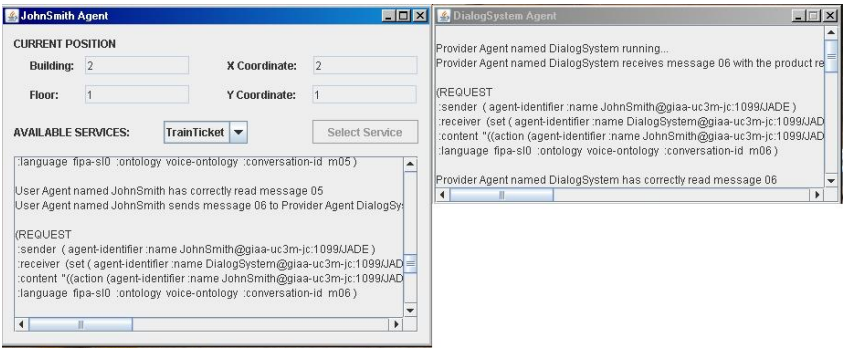

Fig. 3 Query to the Conversational Agent that provides the service

Then, the Provider Agent asks the User Agent about the context information that can be useful for the Conversational Agent to provide an adapted service. The User Agent decides which context information to send as a result of the previous query and transmits this information to the Provider Agent. Once the Conversational Agent has received the context information, it transmits this information to the Context Manager that is included in its architecture. The rest of modules of the Conversational Agent requires the Context Manager for the specific information that they require to adapt its behaviour using the user profile.

Once the interaction with the Conversational Agent has finished, this agent communicates the result of the dialog to the LogAnalyzer Agent. Once the results of this dialog have been interpreted, the LogAnalyzer Agent adds this information to the set of previous dialogs that have been stored regarding the interactions of the User Agent John Smith with the Conversational Agent DialogSystem. This way, this Provider Agent can provide the most adapted service taking into account the specific user's preferences detected in the previous dialogs. This process is automatically carried out, reducing the participation of the user in this process.

To evaluate our architecture, we acquired ten different dialogs for each one of the scenarios considering or not the inclusion of the Context Manager in our

Table 1 Results of the high-level dialog features for the comparison of the two kinds of dialogs

	Without Context Inform.	**Using Context Inform.**
Percentage of successful dialogs	61.6%	78.4%
Average number of turns per dialog	12.6	6.2
Percentage of different dialogs	92.9%	71.9%
Number of repetitions of the most seen dialog	5	12
Number of turns of the most seen dialog	9	5
Number of turns of the shortest dialog	5	5
Number of turns of the longest dialog	25	17

architecture. A dialog simulation technique has been used to acquire the total of 1000 successful dialogs whether introducing the modules that manage context information in our architecture or not [2]. In this technique we automatically acquire a dialog corpus by means of the introduction in our architecture of a dialog manager simulator and a user simulator. A user request for closing the dialog is selected once the system has provided the information defined in the objective(s) of the dialog. The dialogs that fulfill this condition before a maximum number of turns are considered successful. Using this technique it is possible to carry out a detailed exploration of the dialog space and reduce the effort that would be necessary with real users.

We defined seven measures for the comparison of the dialogs acquired using or not context information: the percentage of successful dialogs, the average number of turns per dialog, the percentage of different dialogs, the number of repetitions of the most seen dialog, the number of turns of the most seen dialog, the number of turns of the shortest dialog, and the number of turns of the longest dialog. Using these measures, we tried to evaluate the success of our approach as well as its efficiency with regard to the different objectives specified in the scenarios. Table 1 shows the comparison of these measures. As it can be observed, the first advantage of our approach is regarding the number of dialogs that was necessary to simulate in order to obtain the 1000 successful dialogs of each kind. While, only a 61.6% of successful dialogs is obtained using context information, this percentage increases to 78.4% when the context manager is introduced. The second improvement is the reduction in the number of turns. Using the context manager it is possible to obtain a reduction greater than 50% in the average number of turns per dialog. This is due to the context-aware system requires less information from the user, therefore the possibility of the ASR module introducing errors and the number of data confirmations are reduced. This reduction can also be observed in the number of turns of the longest, shortest and most seen dialogs. Finally, the number of different dialogs is lower using the context information due to the reduction in the number of turns, as can be observed in the number of repetitions of the most seen dialog.

4 Conclusions

In this paper, we have presented a multi-agent architecture for developing context aware adaptable services. This allow us to deal with the increasing complexity that the design of this kind of systems requires, adapting the services that are provided by taking into account the user requirements and preferences by means of a context manager. We have described the different agents that are included and the modules that make it up. One of the main characteristics of our architecture is to ensure a natural and intelligent interaction by means of a conversational agent, whose modules are adapted by means of the context information contained in specific user profiles. The results of the application of our methodology to design a railway information system shows how the main characteristics of the dialogs can be improved by taking into account context information. As a future work, we want to evaluate our methodology with real users and also carry out a detailed study of the user rejections of system-hypothesized actions. Finally, we also want to apply our technique to carry out more complex tasks in which conversational agents no only provide information but also carry out additional functionalities.

Acknowledgements. Funded by projects CICYT TIN2008-06742-C02-02/TSI, CICYT TEC2008-06732-C02-02/TEC, SINPROB, CAM MADRINET S-0505/TIC/0255, and DPS2008-07029-C02-02.

References

1. Bajo, J., Julian, V., Corchado, J.M., Carrascosa, C., De Paz, Y., Botti, V., De Paz, J.F.: An execution time planner for the artis agent architecture. Journal of Engineering Applications of Artificial Intelligence 21(8), 769–784 (2008)
2. Griol, D., Hurtado, L., Sanchis, E., Segarra, E.: Acquiring and Evaluating a Dialog Corpus through a Dialog Simulation Technique. In: Proc. of the 8th SIGdial, pp. 39–42 (2007)
3. Griol, D., Hurtado, L.F., Segarra, E., Sanchis, E.: A Statistical Approach to Spoken Dialog Systems Design and Evaluation. Speech Communication 50(8-9), 666–682 (2008)
4. Jong-yi, H., Eui-ho, S., Sung-Jin, K.: Context-Aware Systems: A Literature Review and Classification. Expert Systems with Applications 36(4), 8509–8522 (2009)
5. McTear, M.F.: Spoken dialogue technology. Springer, Heidelberg (2004)
6. Nieto-Carvajal, I., Botía, J., Ruiz, P., G'omez-Skarmeta, A.: Implementation and evaluation of a location-aware wireless multi-agent system. In: Proc. of EUC 2004, pp. 528–537 (2004)
7. Sánchez-Pi, N., Fuentes, V., Carbó, J., Molina, J.: Knowledge-based system to define context in commercial applications. In: Proc. of SNPD 2007, pp. 694–699 (2007)
8. Seneff, S., Adler, M., Glass, J., Sherry, B., Hazen, T., Wang, C., Wu, T.: Exploiting Context Information in Spoken Dialogue Interaction with Mobile Devices. In: Proc. of Int. Workshop on Improved Mobile User Experience, pp. 1–11 (2007)