Development of CBR-BDI Agents: A Tourist Guide Application

Juan M. Corchado[1], Juan Pavón[2], Emilio S. Corchado[3] and Luis F. Castillo[1]

[1]Dep. Informática y Automática
Universidad de Salamanca
Plaza de la Merced s/n
37008, Salamanca, Spain
Email: corchado@usal.es
http://gsii.usal.es/
[2]Dep. Sistemas Informáticos y Programación
Universidad Complutense Madrid
Ciudad Universitaria s/n
28040, Madrid, Spain
[3]Department of Civil Engineering,
University of Burgos, Spain.
C/ Francisco de Vitoria s/n
09006, Burgos, Spain

Abstract. In this paper we present an agent-based application of a wireless tourist guide that combines the Beliefs-Desires-Intentions approach with learning capabilities of Case Base Reasoning techniques. This application shows how to develop adaptive agents with a goal driven design and a decision process built on a CBR architecture. The resulting agent architecture has been validated by real users who have used the tourist guide application, on a mobile device, and can be generalized for the development of other personalized services.

1 Introduction

Over the last few years, multi-agent systems (MAS) have emerged as an interesting paradigm for constructing distributed and dynamic open systems. MAS have been successfully applied in fields such as electronic commerce, medicine, oceanography, trading market, electronic auctions, production intelligent control, robotics, information retrieval, etc. The telecommunication industry expects a new expansion with the development of UMTS and third generation phone systems. The new challenges of this field require new technology that facilitate the construction of more dynamic, *intelligent*, flexible and open applications, capable of working in a real time environment. MAS solutions intend to cope with the requirements of this kind of systems. Although commercial agent technology today is not yet prepared for such

demand, it is improving continuously and substantially. The proposal presented in this paper is an example of its possibilities and how it has been adopted for the development of a real application.

Agents are usually classified depending on the set of capabilities that they support, such as autonomy, reactivity, proactivity, social ability, reasoning, learning, and mobility, among others [23]. In this work we are mainly interested in the development of deliberative agents using case-based reasoning (CBR) systems, as a way to implement adaptive systems in open and dynamic environments. Agents in this context must be able to reply to events, take the initiative according to their goals, communicate with other agents, interact with users, and make use of past experiences to find the best plans to achieve goals.

Deliberative agents are usually based on a BDI model [20], which considers agents as having certain *mental attitudes*: Beliefs, Desires, and Intentions (BDI). Under this model, agents have a *mental state* that consists of informational, motivational, and deliberative states respectively. Beliefs represent the information about the environment, the internal state the agent may hold, and the actions it may perform. The agent will try to achieve a set of goals, and will respond to certain events.

A BDI architecture has the advantage that it is intuitive and relatively simple to identify the process of decision-making and how to perform it. Furthermore, the notions of belief, desire and intention are easy to understand. On the other hand, its main drawback lies in finding a mechanism that permits its efficient implementation. Most approaches use multi-modal logic for the formalisation and construction of such agents, but they are not always completely axiomatised or they are not computationally efficient (see, for instance, dMARS [10], PRS [17], JACK [5], JAM [14], and AgentSpeak(L) [19]). Rao and Georgeff [20] state that the problem lies in the great distance between the powerful logic for BDI systems and practical systems. Another problem is that this type of agents have difficulties to implement learning capabilities, as these would require constantly adding, modifying or eliminating beliefs, desires and intentions. As most agent applications have highly dynamic environments, we consider that it would be convenient to have a reasoning mechanism that would enable the agent to learn and adapt in real time, while the computer program is executing, avoiding the need to recompile such an agent whenever the environment changes.

Taking into account previous works [12, 8], we propose the use of a case-based reasoning (CBR) system for the development of deliberative agents. The proposed method starts by identifying agent roles and goals, in a similar way as in AAII/BDI methodology [20], but the design and implementation of the agent architecture follows the form of CBR systems, which facilitates learning and adaptation, and therefore a greater degree of autonomy than with a pure BDI architecture. This is made by mapping the three mental attitudes of BDI agents into the information manipulated by a CBR system. This direct mapping between the agent conceptualisation and its implementation is the main difference with respect to other proposals that have also tried to combine BDI and CBR [16, 4, 22, 18].

The proposed agent architecture has been validated with the implementation of a multi-agent system (MAS) that provides tourist guide services through mobile

devices. This application shows how to develop adaptive agents with a goal driven design and a decision process built on a CBR architecture. The system has been validated by users who have used it when visiting the city of Salamanca. The system is able to program a tourist route, and modify it according to the conditions of the places to visit and the available time for the tourist. Because of its design, the services of the tourist guide agent can be easily extended (e.g. to recommend restaurants in the area of the tourist route), and support a high degree of scalability in the number of users.

The rest of the paper is organized as follows. Section 2 describes the wireless tourist guide application and its main components, basically three types of agents, one of them of deliberative nature, which will be used later to show the application of the CBR-BDI agent architecture. This agent architecture is described in Section 3. Implementation details are provided in Section 4. Finally, in the conclusions, we present some of the evaluation results when using this application.

2 The Wireless Tourist Guide System

The tourism industry is one of the major resources of income in Spain and the services offer in this sector is continuously updated and improved. This strategic sector has attracted the attention of the telecommunication operators, who are investing in new tools, services and market research. In this framework, and with the support of a telecommunications partner, a Tourist Guide application, called *TOURIST GUIDE-USAL*, has been developed as a MAS. With this system we wanted to show the feasibility and reliability of this technology, and that fully-functional systems may be constructed within the time restrictions imposed by the industry.

TOURIST GUIDE-USAL agents assist potential tourists in the organization of their tourist routes and enable them to modify their schedules on the move using wireless communication systems. This system has been constructed using an engineering framework developed to design and implement an agent-based tool, as well as integrating existing state of the art in order to create an open, flexible, global anticipatory system with mobile access for the promotion and management of inland and cultural tourism, which will be user-friendly, cost-effective and secure. The system has been standardized to run in any mobile device and is interlingua.

The integrated, multi-platform computer system is composed of a guide agent (*Planner Agent*) that assesses the tourists and help them to identify tourist routes in a city with a given visiting period of time and under a number of restrictions related to cost, tourist interest, etc. There is one assistant agent for each user of the system, the *Performer Agents*. Each user willing to use the system has to register and solicit one of these agents. Finally, there is a third type of agent, the *Tracker agent*, which maintains updated information about the monuments, the restaurants, public transport conditions, etc. This agent maintains horizontally and vertically compiled information on hotel accommodation, restaurants, the commercial sector and transport, in order to meet the needs of the potential visitor on an individually customized basis, and

responds to requests for information, reservations and purchases in the precise moment that they are expressed.

The user may decide whether to install the corresponding *Performer Agent* on a mobile phone or PDA, or run it on the server and interact with it via its mobile device. The first choice supposes a reduction of the cost, since the tourist can interact with his agent as much as needed at no cost because it is installed in the wireless device. Nevertheless, the agent will have to contact regularly with the *Planner Agent*.

Fig. 1 describes the system architecture from a very high abstraction level. Users may interact either with their *performer agents* installed in their wireless devices or in an internet server. The *performer agents* interact with the planner agent looking for plans, and the *tracker agent* interacts with the *planner agent* to exchange information. The *planner agent* is the only CBR-BDI agent in this architecture. The *performer agents* can be considered assistant agents and the *tracker agent* is a reactive agent. The focus in this paper is on the *planner agent*.

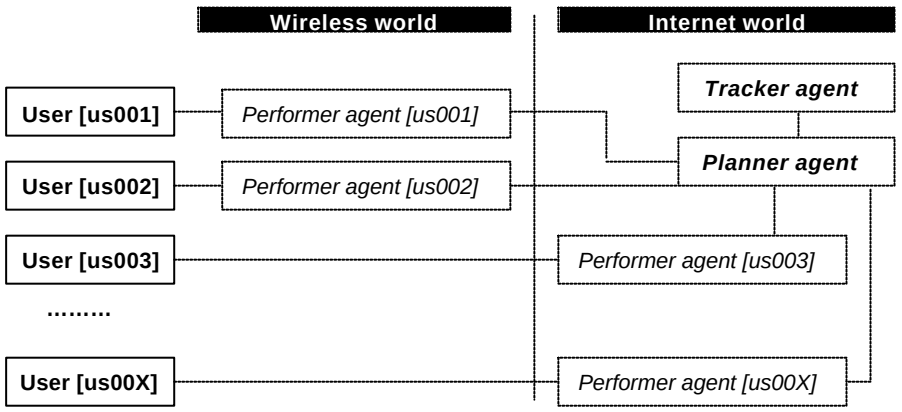

Fig. 1. CBR/Agent integration diagram.

3 Case-based Reasoning Systems and Deliberative Agents

Case-based reasoning (CBR) systems solve new problems by adapting solutions that have been used in the past. Fig. 2 shows a classical CBR reasoning cycle that consists of four sequential phases: retrieve, reuse, revise, and retain [1]. Very often, an additional activity, revision of the expert's knowledge, is required because the memory can change as new cases may appear during this process. Each of these activities can be automated, which implies that the whole reasoning process can be automated to a certain extent [9]. According to this, agents implemented using CBR systems could reason autonomously and therefore adapt themselves to environmental changes.

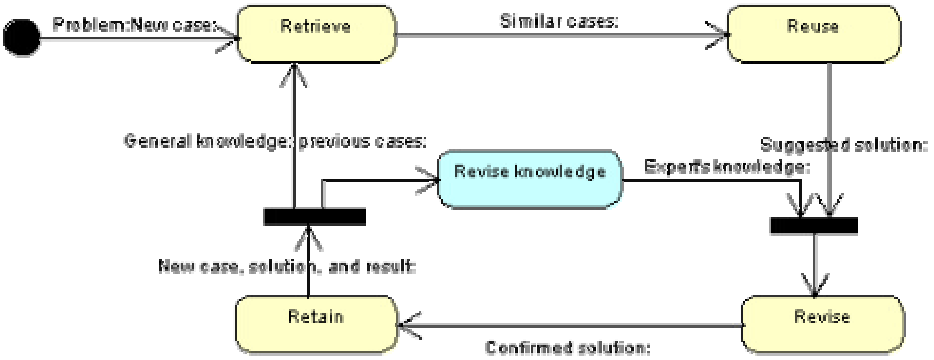

Fig. 2. UML activity diagram describing a CBR life-cycle

On the other hand, as most agent architectures are based on the BDI model, if we are able to establish a relationship between cases, the CBR life-cycle, and the mental attitudes of BDI agents, we can provide a model that facilitates the implementation of the BDI agents using the reasoning cycle of a CBR system, with all its advantages.

Our proposal defines a direct mapping from the concept of an agent to the reasoning model, paying special attention to two elements. First, how the mapping should allow a direct and straightforward implementation of the agent. And second, how the agent is able to learn and evolve with the environmental changes. In this model, the CBR system is completely integrated into the agents' architecture, which differs with the above-mentioned works, in which the agents see the CBR system as just a reasoning tool. Our proposal is also concerned with the agent's implementation and presents a "formalism" which is easy to implement, in which the reasoning process is based on the concept of *intention*. In this model, intentions are cases, which have to be retrieved, reused, revised and retained. To achieve both goals, the structure of the CBR system has been designed around the concept of a *case*. A case is made of three components: the problem, the solution, and the result obtained when the proposed solution is applied. The problem defines the situation of the environment at a given moment. The solution is the set of states that are undergone by the environment as a consequence of the actions that have been carried out inside it. And the result shows the situation of the environment once the problem has been solved. This can be expressed as follows [8]:

Case: <Problem, Solution, Result>
  Problem: initial_state
  Solution: sequence of <action, [intermediate_state]>
  Result: final_state

BDI agent (beliefs, desires , intentions)
  Belief: state
  Desire: set of <final_state>
  Intention: sequence of <action>

In a BDI agent, each state is considered as a belief; the objective to be reached may also be a belief. The intentions are plans of actions that the agent has to carry out in

order to achieve its objectives, so an intention is an ordered set of actions; each change from state to state is made after carrying out an action (the agent remembers the action carried out in the past when it was in a specified state, and the subsequent result). A desire will be any of the final states reached in the past (if the agent has to deal with a situation, which is similar to a past one, it will try to achieve a similar result to the previously obtained result).

The relationship between CBR systems and BDI agents can be established implementing cases as beliefs, intentions and desires which led to the resolution of the problem. This relationship is shown in Fig. 3. When the *agent starts to solve a new problem*, with the intention of *achieving a goal*, it begins a new **CBR reasoning cycle**, which will help to **obtain the solution**. The **retrieval, reuse and revise stages of the CBR system** facilitate the construction of the *agent plan*. The *agent's knowledge-base* is the **case-base of the CBR system** that stores the **cases of** *past believes, desires and intentions*. The *agents* work in dynamic environments and their *knowledge-base* has to be adapted and updated continuously by the **retain stage of the CBR system**.

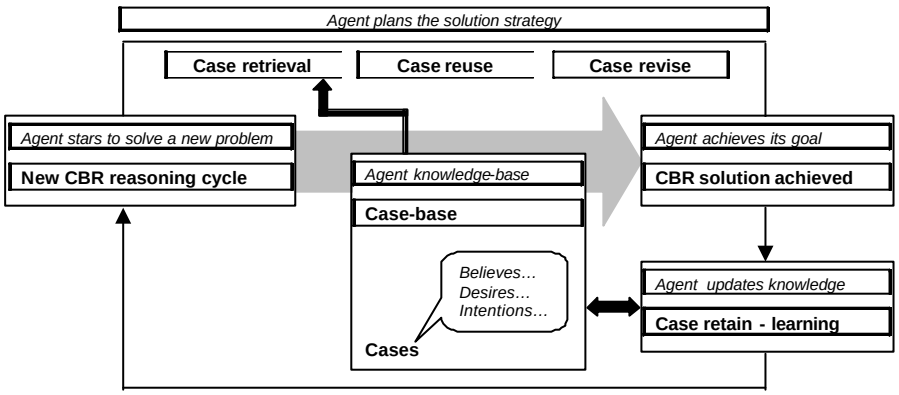

Fig. 3. CBR/Agent integration diagram

Based on this relationship, agents (conceptual level) can be implemented using CBR systems (implementation level). This means, a mapping of agents into CBR systems. The advantage of this approach is that a problem can be easily conceptualised in terms of agents and then implemented in the form of a CBR system. So once the beliefs, desires and intentions of an agent are identified, they can be mapped into a CBR system.

4 CBR-BDI Agent Development: the Planner Agent

In this paper we are concerned with the design and implementation of deliberative agents following the principles described in the previous section. Here we start by

considering that deliberative agents have already being identified in the system, with their roles, responsibilities and services (the basic organization of agents in this system was already depicted in section 2). We focus, therefore, on the design of the *Planner Agent*, the only deliberative agent in this system, by using the principles described in the previous section.

To set up an agent using the CBR-BDI agent architecture we need to identify an initial set of beliefs, desires and intentions and include them in the case-base of the agent in the form of cases. Then, a number of metrics for the retrieval, reuse, revise and retain steps has to be defined. Besides, rules that describe the Expert's knowledge must be established, if available. Once the agent has been initialised it starts the reasoning process and the four steps of the CBR system are run sequentially and continuously until its goal is achieved (or there is enough evidence for a failure situation).

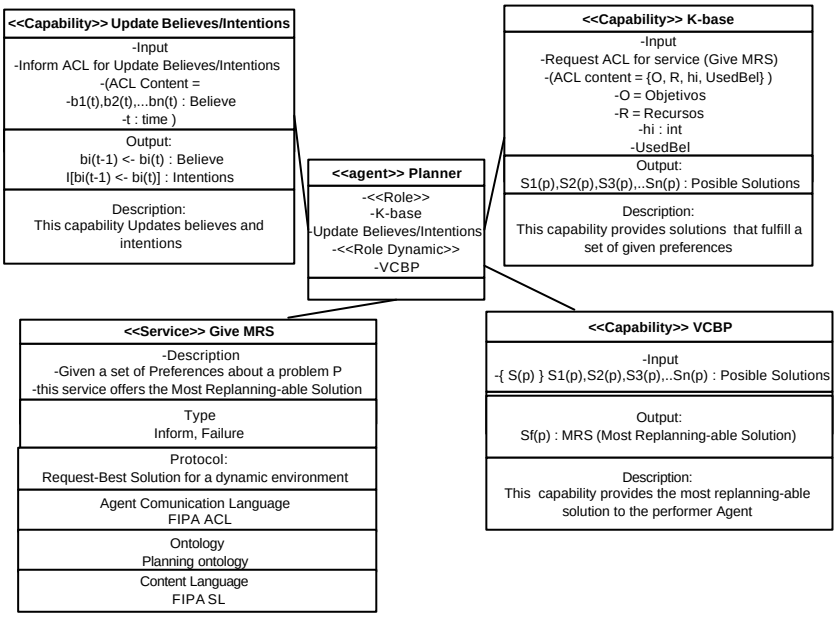

Fig. 4. Planner Agent class diagram in AUML

Fig. 4 shows the AUML (Agent Unified Modeling Language) class diagram of the *Planner Agent* (Agent UML is a modelling language that extends UML; more information at www.auml.org). In these types of diagrams, the roles and goals of the agents are represented as *Capabilities* that may change with the time. In particular, the roles of the *Planner Agent* are (i) to update the believes and intentions, which are stored in the form of cases, (ii) to identify those believes and intentions that can be used to generate a plan n, and (iii) to provide adequate plans to the *Performer Agent* given a number of conditions. These roles allow the agent to generate the closest to

the optimum plan, which in this case has also to be the most replan-able solution. In this context, when the *Performer Agent* asks for a tourist route, given a number of constraints such as the money the tourist is willing to spend, the number of monuments to visit, the type of restaurants to eat, the time availability for the holiday, etc. the Planner Agent generates a plan that fulfils such conditions. This plan is easy to modify at execution time if the user changes of mind. The Planner Agent is a CBR-BDI agent, where the role (i) is carried out during the Retain stage of the CBR life cycle, role (ii) is the Retrieval step, and role (iii) is the Reuse stage.

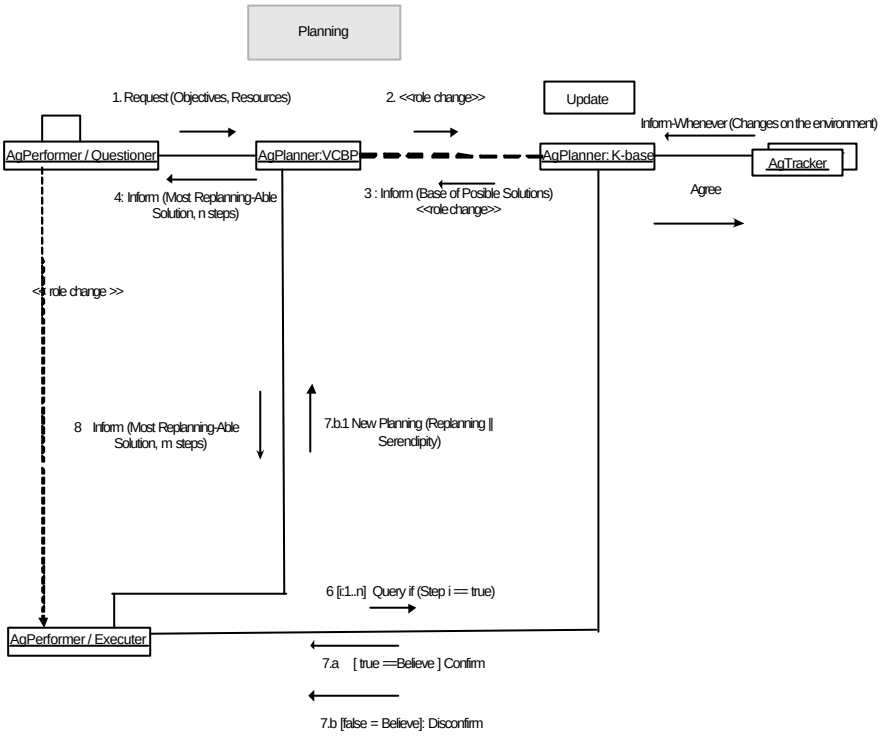

Fig. 5. Collaboration among agents in the tourist guide application

The *Performer agents*, are assistant agents. Each of them is associated to one user and contact the *Planner Agent* to request a plan. These agents may be in waiting mode, waiting for a request from the user, may ask to the *Planner Agent* for a plan, or request a modification in a plan (replanning) to the *Planner Agent*. The *Tracker Agent* is always looking for changes in the visiting conditions of the different sites, and keeps a record of them. The *Planner Agent* regularly contacts the *Tracker Agent* looking for changes in the environment. Fig. 5 shows the collaboration of these agents with a sequence diagram.

4.1 Implementation of the CBR system for the Planner Agent

The *Planner Agent* uses a CBR system for reasoning and generating its plans. This agent has three roles:

- To identify those believes and intentions that can be used to generate a plan.
- To provide adequate plans to the *Performer Agent* given a number of conditions.
- To update the believes and intentions, which are stored in the form of cases.

These roles are carried out sequentially and correspond with the retrieval, reuse, and retain stages of a CBR system. The reasoning cycle has been constructed using a variational calculus based strategy [12].

The retrieval stage must be carried out using a method that guarantees the retrieval of a reasonably small number of cases that are related to the current problem case. We have experimented with a number of different retrieval methods such as Sparse Kernel Principal Component Analysis [8] or a K-nearest neighbour algorithm based strategy [12]. The best results have been obtained with a variational calculus based strategy, as shown below.

Planning can be defined as the construction of a course of actions to achieve a specified set of goals in response to a given situation. The classical generative planning process consists mainly of a search through the space of possible operators to solve a given problem, but for most practical problems this search is intractable. Given that typical planning may require a great deal of effort without achieving very good results, several researchers have pursued a more synergistic approach through generative and case-based planning [4]. In this context, case indexation strategy facilitates and speeds up the planning process substantially.

A case in case-based planning consists of a problem (initial situation and set of goals) and its plan. Given a new problem, the objective of the retrieval and reuse phase is to select a case or a number of cases from the case-base whose problem description is most similar to the description of the new problem and to adapt it/them to the new situation. In case-based reasoning, two different approaches to reuse can be distinguished: transformational and derivational adaptation. Transformational adaptation methods usually consist of a set of domain dependent concepts which modifies the solution directly obtained in the retrieved case. For derivational adaptation, the retrieved solution is not modified directly, but is used to guide the planner to find the solution.

There are different ways to integrate generative and case-based planning: PRODIGY [7, 21], PARIS [2, 13], and Variational Calculus Based Planner (VCBP) [11], which is the method proposed for the resolution of the case-study. These planners may be used in the development of deliberative agent-based systems. In PRODIGY and PARIS the workload imposed on the generative planner depends on the amount of modification that is required to adapt the retrieved cases. Looking at the structure, we can say that PARIS is a *domain-independent* case-based planner while PRODIGY is *domain semi-dependent*. On the other hand, although VCBP is domain dependent, it introduces a new interesting strategy to efficiently deal with the adaptation stage.

Variational Calculus-based Planner (VCBP) guarantees the planning and re-planning of the intentions in execution time. This planning strategy is divided into two steps:

1. identify cases that are similar to the problem case (retrieval stage), and
2. adapt them to the problem case (reuse stage), which correspond to the two roles of the Planner Agent.

Variational calculus automates the reasoning cycle of the BDI agents, and guarantees the identification of an efficient plan, closed to the optimum. Although different types of planning mechanisms can be found in the literature, none of them allows the replanning in execution time, and agents inhabit changing environments in which replanning in execution time is required if goals are to be achieved successfully in real-time.

Some of the planning techniques developed for case-based reasoning systems to select the appropriate solution to a given problem do not have mechanisms to deal with the changes in the environment. For instance, Corchado and Laza [8] and Knobolock *et al.* [15] introduce a kind of plan schema that needs to be reprogrammed over time, when the planning domain changes. Bergmann and Wilke [3], and Camacho *et al.* [6] propose an architecture that tries to be more flexible, in which, if new information has to be introduced from the environment to the system, it is only necessary to change the planning domain instead of reprogramming the plan schema by hand. This architecture allows building plans that contain steps with no detailed information. This is useful because if no specific information is supplied, the solution can handle planning generic operators, plans that are not influenced by unexpected changes.

Now, to find out if the abstract proposed plan is adequate it is necessary to put it into practice in a real domain. This operation requires a great amount of computational time and resources which may be a disadvantage, in for example, web related problems. The flexibility of this approach increases the time spent in applying the abstract solution to the real problem, which is a handicap for real time systems. The proposed solution, a variational calculus based planner, deals adequately with environmental real-time problem changes without applying a reprogramming strategy and without the disadvantages shown in the works mentioned before, because the technique used can replan in execution time.

5 Results and Conclusions

Development times in the telecommunication industry have been drastically reduced. In the last decade a standard project used to have a development period of 8 to 15 months, and now this period has been reduced to 3 to 5 months. This requires an experimented development team, the use of a reliable technology, and knowledge of the problem domain (or at least the capacity of learning fast). The CBR-BDI agents that are proposed in this paper facilitate the construction of distributed wireless system for mobile devices and may be adapted for different problem domains, within the constrains imposed by the industry. The developed infrastructure includes tools for generating CBR-BDI autonomous agents that can reason, learn and communicate with the users and with other agents, a simple communication protocol based on the FIPA

ACL standards, and a number of established processes that facilitate the analysis and design of a MAS using AUML.

The proposed system has been used to improve an agent based system developed for guiding tourists around the city of Salamanca (Spain). As mentioned before, the tourists may use a mobile device to contact their agents and to indicate their preferences (monuments to visit, visits duration, dinner time, amount of money to spend, etc.). There are different types of **cases**. The cases store information about the environment, for example the opening and closing times of monument. This type of information can be seen as an agent believe, for example, The Museum of Contemporary Art opens from 9:00 to 14:00 and from to 16:30 to 20:00. Cases may also be previous successful routes (plans), as shown in Fig. 6(a), that includes the monuments to visit, the time to spend visiting each monument, information about the cost of the visit, the time required for going to one place to another, the characteristics of the route (museum route, family route, university route, roman route, gothic route, etc.), etc. Once a tourist contacts the system he has to describe his profile, to select the type of visit in which he is interested in, to determine how much money he wants to spend and for how long, and the type of restaurants he, she or a family like more. This information is used to construct the **problem case**. Then the reasoning mechanism of the planning agent generates the plan. This reasoning mechanism is the previously mentioned CBR system using VCBP [11, 12]. Fig. 6(b) shows a graphical view of a generated plan.

(a) (b)

Fig. 6. System overview.

The initial system was tested from the 1st of May to the 15th of September 2003. The case base was initially filled with information collected from the 1st of February to the 25th of June 2003. Local tourist guides provided the agent with a number of standard routes. Three hotels of the City offered the option to their 6217 guests to use the help of the agent or a professional tourist guide, 14% of them decided to use the agent based system and 23% of them used the help of a tourist guide. The rest of the tourists visited the city by themselves. During this period the *Planner agent* stored in

its memory 1630 instances of tourist circuits, which covered a wide range of all the most common options that offers the City of Salamanca. The system was tested during 135 days and the results obtained were very encouraging. In this experiment the agent intentions were related to a one-day route (a maximum of 12 hours). On the arrival to the hotel the tourists were asked to evaluate their visit and the route. Table 1 shows the responses given by the tourists after their visit. The tourists that used the help of the agent-based tourist guide provided the answer directly to the agent.

Table 1 shows the degree of satisfaction of the tourists. As it can be seen, the degree of satisfaction of the tourist that used the help of a professional tourist guide is higher that in the other two cases. Nevertheless, the percentage of the tourists whose degree of satisfaction was very high (between 8 and 10) is very similar in the case of the tourists that use the help of the agent and in the case of the tourists that use the tourist guide. 38% of the tourists that used the agent based system let us know that the system did not work successfully due to technical reasons (possibly the server was down, there was a lack of coverage, the tourist did not use the wireless system adequately, etc.) If we take this into consideration, we can say that most of the tourist (92%) that used the help of the agent and did not have technical problems had a high or very high degree of satisfaction (6-10). This degree of satisfaction is higher that the one of the tourist (82,3%) that used the help of a tourist guides.

Table 1. Tourists evaluation.

	%	**Evaluation - degree of satisfaction**				
Tourists that…		8-10	6-8	4-6	0-4	No answer
Used the help of the agent	14%	(55,9%)	(4,7%)	(2,4%)	(0,7%)	(36,3%)
Used the help of a tourist guide	23%	(62,7%)	(19,6%)	(8,9%)	(1%)	(7,8%)
Did not use any of the previous	63%	(16,7%)	(8,3%)	(1,2%)	(0,2%)	(78,8%)

The CBR-BDI architecture solves one of the problems of the BDI (deliberative) agent architectures, which is the lack of learning capability. The reasoning cycle of the CBR systems helps the agents to solve problems, facilitate its adaptation to changes in the environment and to identify new possible solutions. New cases are continuously introduced and older ones are eliminated. The CBR component of the architecture provides a straight and efficient way for the manipulation of the agents knowledge and past experiences. The proposal presented in this paper reduces the gap that exists between the formalization and the implementation of BDI agents.

Acknowledgements

This work has been supported by the MCyT projects TEL99-0335-C04-03, SEC2000-0249 and TIC2003-07369-C02-02.

References

1. Aamodt A. and Plaza E. (1994). Case-Based Reasoning: foundational Issues, Methodological Variations, and System Approaches, AICOM. Vol. 7. No 1, March.
2. Bergmann, R. and W. Wilke (1995). Learning abstract planning cases. In N. Lavrac and S. Wrobel (Eds.), Machine Learning: ECML-95, 8th European Conference on Machine Learning, Heraclion, Greece, April 1995. Number 912 in Lecture Notes in Artificial Intelligence, pp. 55-76. Berlin, Springer
3. Bergmann, R. and W. Wilke (1996). On the role of abstraction in case-based reasoning. Lecture Notes in Artificial Intelligence, 1186, pp. 28-43. Springer Verlag.
4. Bergmann, R., Muñoz-Ávila, H., Veloso, M. and Melis, E. (1998). CBR Applied to Planning. In Lenz, M. Bartsch-Sporl, B., Burkhard, H. and Wess, S. (Eds.) Case-Based Reasoning Technology: From Foundations to Applications. Lecture Notes in Computer Science 1400, pp. 169-200. Springer 1998, ISBN 3-540-64572-1.
5. Busetta, P., Ronnquist, R., Hodgson, A., Lucas A. (1999). JACK Intelligent Agents Components for Intelligent Agents in Java. Technical report, Agent Oriented Software Pty. Ltd, Melbourne, Australia, 1998.
6. Camacho D., Borrajo D. And Molina J. M. (2001) Intelligence Travell Planning: a multiagent planing system to solve web problems in the e-turism domain. International Journal on Autonomous agens and Multiagent systems. 4(4) pp 385-390. December.
7. Carbonell J.G., Knoblock C. A., Minton S. (1991). Prodigy: An integrated architecture for planning and learning. In K. VanLenh (Ed.), Architectures for Intelligence, pp.241-278. Lawrence Erlbaum Associates, Publishers.
8. Corchado J. M. And Laza R. (2003). Constructing Deliberative Agents with Case-based Reasoning Technology, International Journal of Intelligent Systems. Vol 18, No. 12, December.
9. Corchado J. M. and Lees B. (2001). A Hybrid Case-based Model for Forecasting. Applied Artificial Intelligence. Vol 15, no. 2, pp.105-127.
10. D'Iverno, M., Kinny, D., Luck, M., and Wooldridge, M. (1997). A Formal Specification of dMARS. In: Intelligent Agents IV, Agent Theories, Architectures, and Languages, 4th International Workshop, ATAL '97, Providence, Rhode Island, USA, July 24-26, 1997, Proceedings. Lecture Notes in Computer Science 1365, Springer Verlag, pp. 155-176.

11. Glez-Bedia M. and Corchado J. M. (2002) A planning strategy based on variational calculus for deliberative agents. Computing and Information Systems Journal. Vol 10, No 1, 2002. ISBN: 1352-9404, pp: 2-14.
12. Glez-Bedia M., Corchado J. M., Corchado E. S. and Fyfe C. (2002) Analytical Model for Constructing Deliberative Agents, Engineering Intelligent Systems, Vol 3: pp. 173-185.
13. Holte, R. C., T. Mkadmi, R. M. Zimmer, and A. J. MacDonald (1995). Speeding up problem solving by abstraction: A graph-oriented approach. Technical report, University of Ottawa, Ontario, Canada.
14. Huber, M. (1999). A BDI-Theoretic Mobile Agent Architecture. AGENTS '99. Proceedings of the Third Annual Conference on Autonomous Agents, May 1-5, 1999, Seattle, WA, USA. ACM, pp. 236-243.
15. Knobolock C. A., Minton S., Ambite J. L., Muslea M., Oh J. and Frank M. (2001). Mixed- initiative, multisource information assistants. 10th International world wide web conference (WWW10). ACM Press. May 1-5,pp.145-163.
16. Martín F. J., Plaza E., Arcos J.L. (1999). Knowledge and experience reuse through communications among competent (peer) agents. International Journal of Software Engineering and Knowledge Engineering, Vol. 9, No. 3, 319-341.
17. Myers, K. (1996). A Procedural Knowledge Approach to Task-Level Control. Proceedings of the Third International Conference on Artificial Intelligence Planning Systems,, pp. 158-165.
18. Olivia C., Chang C. F., Enguix C.F. and Ghose A.K. (1999). Case-Based BDI Agents: An Effective Approach for Intelligent Search on the World Wide Web, AAAI Spring Symposium on Intelligent Agents, 22-24 March 1999, Stanford University, USA.
19. Rao, A. S. (1996). AgentSpeak(L): BDI Agents speak out in a logical computable language. Agents Breaking Away, 7th European Workshop on Modelling Autonomous Agents in a Multi-Agent World, Eindhoven, The Netherlands, January 22-25, 1996, Proceedings. Lecture Notes in Computer Science 1038, Springer Verlag, pp. 42-55.
20. Rao, A. S. and Georgeff, M. P. (1995). BDI Agents: From Theory to Practice. First International Conference on Multi-Agent Systems (ICMAS-95). San Franciso, USA.
21. Veloso, M. M. (1994). Planning and Learning by Analogical Reasoning. Number 886 in Lectures Notes in Computer Science. Berlin, Springer.
22. Wendler J. and Lenz M. (1998). CBR for Dynamic Situation Assessment in an Agent-Oriented
Setting. Proc. AAAI-98 Workshop on CBR Integrations. Madison (USA) 1998.
23. Wooldridge, M. and Jennings, N. R. (1995) Agent Theories, Architectures, and Languages: a Survey. In: Wooldridge and Jennings, editors, Intelligent Agents, Springer-Verlag, pp. 1-22.