Exploiting planning capabilities of a rational agent in adaptive web-based recommendation systems: a case-study

Matteo Baldoni, Cristina Baroglio, Viviana Patti, Andrea Molia
Dipartimento di Informatica Università degli Studi di Torino
c.so Svizzera 185, 10149, Torino
Italy

ABSTRACT

In this paper we show a tutoring system having a multi-agent architecture. The kernel of the system is a *rational agent* , whose behavior is programmed in the agent logic programming language DyLOG. In the prototype that we implemented the planning capabilities of the language are exploited for dynamically building a web site, with the aim of helping the current student to define a study plan according to his needs, constraints, and competence.

KEY WORDS: multi-agent system, planning-based recommendation system, adaptive web sites.

1. INTRODUCTION AND MOTIVATION

Many of the most advanced solutions on web site adaptation to the user, such as [1,2,3,4,5], start from the assumption that adaptation should focus on the user's characteristics; in different ways, they all try to associate the user with a reference prototype, the “user model”, to which the presentation is adapted. By doing so, such approaches catch important aspects connected to the personality and the general interests of the user. In our opinion, however, especially in the case of recommendation systems, the user's intentions and needs - which may change at every connection- play a very important role in order to achieve a real adaptation. In [6,7] we present an approach to web site adaptation which uses an agent logic programming language, called DyLOG, for building cognitive agents, that dynamically generate a site based on the user's goals and constraints. When a user connects to a site managed by one of our agents, (s)he does not access to a fixed graph of pages and links but (s)he interacts with a rational agent that, starting from a knowledge base specific to the site and from the requests of the user, builds an ad hoc presentation structure. We obtain run-time adaptation by exploting the capability of a rational agent to reason about its actions and to make plans. So the problem to generate a web site adapted to user's needs is interpreted as a planning problem. In this sense, our web sites are structureless. Their structure corresponds to the plan built for pursuing the user's goal. Thus adaptation occurs at the navigation level rather than at the presentation level. Indeed, our focus is on the definition of the navigation possibilities available to the user and on determining which page to display based on the dynamics of the interaction. This approach brings along various innovations w.r.t. what can be found in the literature. From a human-machine interaction perspective, the kind of adaptation that we propose is focused on the actual interests of the user for that specific connection and not, as it often happens in the user model approach, on his/her generic and cultural interests or past choices (past-oriented adaptation). From a web site design perspective, some advantage is expected for the build-modify-update process. In fact, while in order to modify a classical web site one has to change both the contents of the pages and the links among them, in our case the site does not exist as a structure, because the structure is built at the moment. All that we have is a data base containing the domain knowledge, whose maintenance is simple, and an agent program, that is not changed often.
Although our approach could recall some works on Natural Language cooperative dialogue systems, there are some differences. For instance, in [8] Sadek and Bretier proposed a logic of rational interaction for implementing the dialogue management components of a spoken dialogue system. This work is based, like DyLOG, on dynamic logic and reasoning capabilities on actions and intentions are exploited for planning dialogue acts. Our work, instead, is not focused on recognizing or inferencing the user's intentions (which is also a very interesting task): the user's needs are taken as explicit input by the software agent, which uses them to generate the goals that will drive its behavior. The novelty of our approach is that we exploit planning capabilities for building web site presentation plans guided by the user's goals rather than for dialogue act planning. The structure of the site is built by the system as a conditional plan, according to the initial user's needs and constraints. The execution of the plan by the system corresponds to the navigation of the site, where the user and the system cooperate for finding a solution to the user's problem.
We chose DyLOG as an agent programming language due to two of its main characteristics. The first is that, being based on a formal theory of actions, it can deal with

reasoning about action effects in a dynamically changing environment and, as such, it supports planning. The second is that the logical characterization of the language is very close to the procedural one, and this allows to reduce the gap between theory and use. In order to check the validity of the proposed approach, we have implemented an agent system, named WLog. In this application the users connect to the structureless web site for asking for support in some task. As an example, we will show a virtual tutor that helps users to build "studiorum itinera" that depend on the competence they want to acquire, on their current competence, on the student's time constraints, and on the material available in the system's store. Similarly to what happens in an interaction between humans, the solution is obtained thanks to a dialogue, in this case guided by the agent. Requests, information, and proposals are presented to the user in the form of web pages. Technical information about the system can be found at http://www.di.unito.it/~alice.

2. MODELING THE VIRTUAL TUTOR AS A RATIONAL AGENT

In this section we will sketch the kernel of the system, i.e. the rational agent that builds study plans. The language that we used for specifying our virtual tutor is based on a logical theory for reasoning about action and change in a modal logic programming setting. It allows one to specify a rational agent behavior by defining both a set of simple actions, that the agent can perform (some of which are sensing and suggesting actions for interacting with the user) and a set of Prolog-like procedures, which build complex behaviors upon simple actions. The advantage of DyLOG is that its interpreter allows both to execute the procedures, which define the agent behavior, and to reason about their execution, extracting (possibly) conditional plans. The plan extraction process of the interpreter is a straightforward implementation of the proof procedure contained in the theoretical specification of the language. See [9,10] for details on the language and its use for agent programming.
We assume that the behavior of a rational agent is driven by a set of goals. In the particular case of the virtual tutor, at the beginning of the interaction the student expresses his/her desire. The tutor starts a dialogue aimed at finding both a set of goals and a set of constraints (such as the overall time the student can devote to the course). Afterwards, it adopts the user's goals and exploits its reasoning capabilities to build a *conditional plan* that will allow the student to grow knowledge about the chosen topic. Roughly speaking, the conditional plan is the dynamically generated web site, run-time adapted to the user's needs. Branches correspond to alternatives and are generated whenever the procedures which specify the agent behavior contain actions that perform queries for inputs, whose outcome cannot be known at planning time. For instance, in the example reported in Fig. 1 the student expressed his intention to learn about programming languages. The execution consists in showing the web pages which correspond to the different actions in the conditional plan. The process is similar to the one presented in [7].
The agent system is currently used for helping the user to build a study plan by working on *meta-knowledge* about the various courses. Nevertheless, an agent system of this kind could be a valid interface for an on-line library of training modules (in the line of commercial systems, such as Competus Framework). Modules can be tutorials, CBT tools, links to web pages, etc. and have the most various origin. Each module should have an associated *description* about the prerequisites necessary to use it and what it allows to learn. Such a system would be extremely useful for all those persons who cannot attend regular classes: they could download the suggested modules and use them when and where they can. This will be a subject of our future work.

3. THE ARCHITECTURE

WLog, the prototype system, that we developed in order to verify the feasability of the intention-driven approach, has a *multi-agent system* architecture. Agent technology allows complex systems to be easily assembled by means of the creation of distributed artifacts able to accomplish their tasks through cooperation and interaction. Systems of this kind have the advantage of being modular and, therefore, flexible and scalable. So, on one hand, each module can be developed by exploiting the best, specific technology for solving a given issue, on the other, new components can be added for supporting either new functions or a wider number of users. Briefly, the system consists of different kinds of agent: reasoners, executors, data-base managers, and a facilitator.
A DyLOG reasoner generates conditional presentation plans; once built a plan is executed. Actual execution is the task of the executors, which are Java servlets embedded in an Apache web server.
Executing a conditional plan implies following one of the paths; only the part of the site corresponding to this path will actually be built. The execution of the actions in the path consists in showing one or more web pages to the user; in particular, when a page that corresponds to a branching point is shown, a feedback from the user is required. Data-base managers handle the system stores and supply information about the available teaching material to the other agents in the system. Currently we use a data-base implemented in XML [11].
The connection between the agents has the form of message exchange in a distributed system; message exchange is FIPA-like [12]. Each agent is identified by its "location" which can be obtained by other agents from a facilitator ("Agent Locator" in Fig. 2).
The interaction between the user and WLog starts with the declaration of the user's goal. The executor works as an interface between the reasoner and the user. It looks

for a free reasoner; if one is available, from that moment till the end of the connection will be dedicated to serve that specific user.

REFERENCES

[1] L. Ardissono, A. Goy, Tailoring the interaction with users in electronic shops, *Proc. 7th Int. Conf. on User Modeling*, 1999.
[2] L. Ardissono, A. Goy, G. Petrone, M. Segnan, A software architecture for dynamically generated adaptive Web stores, *Proc. 17th Int. Joint Conf. on Artificial Intelligence*, Seattle, Oregon, USA, 2001.
[3] B.N. De Carolis, Introducing reactivity in adaptive hypertext generation, *Proc. 13th European Conf. on Artificial Intelligence*, Brighton, UK, 1998.
[4] B.N. De Carolis, S. Pizzutilo, F. de Rosis, User Manuals as Animated Agents, *Proc. AI*IA Workshop on "Agenti intelligenti e internet: teorie, strumenti e applicazioni*", Milano, Italy, 2000.
[5] L. Marucci, F. Paternò, Designing an Adaptive Virtual Guide for Web Application, *Proc. 6th ERCIM Workshop (UI4All),* Florence, Italy, 2000.
[6] M. Baldoni, C. Baroglio, V. Patti, Strureless Intention-guided Web Sites: Planning-based Adaptation, *Proc. 1st Int. Con. on Universal Access in Human-Computer Interaction, track of HCI International 2001*, 2001.
[7] M. Baldoni, C. Baroglio, A. Chiarotto, V. Patti, Programming Goal-driven Web Sites using an Agent Logic Language, *Proc. 3rd Int. Symp. on Practical Aspects of Declarative Languages*, Las Vegas, Nevada, USA, 2001, 60-75.
[8] P. Bretier, D. Sadek, A rational agent as the kernel of a cooperative spoken dialogue system: implementing a logical theory of interaction, *Proc. of ECAI-96 Workshop on Agent Theories, Architectures, and Languages*, 1997.
[9] M. Baldoni, L. Giordano, A. Martelli, V. Patti, Reasoning about Complex Actions with Incomplete Knowledge: a Modal Approach, *Tech. Rep. 53/00, Dip. Informatica, Università di Torino,* Torino, Italy, *2000.*
[10] M. Baldoni, L. Giordano, A. Martelli, V. Patti, Modeling Agents in a Logic Action Language, *Proc. of the Workshop on Rational Agents, (FAPR'00),* London, UK, *2000*.
[11] A. Molia, Analisi di XML come linguaggio per la rappresentazione di una base di conoscenza e realizzazione di un agente per la sua interrogazione, *Master Thesis*, *Univ. Of Torino*, Torino, Italy, 2001.
[12] Agent Communication Language, *FIPA97 Specification, Foundation for Intelligent Physical Agents*, 1997, http://www.fipa.org.

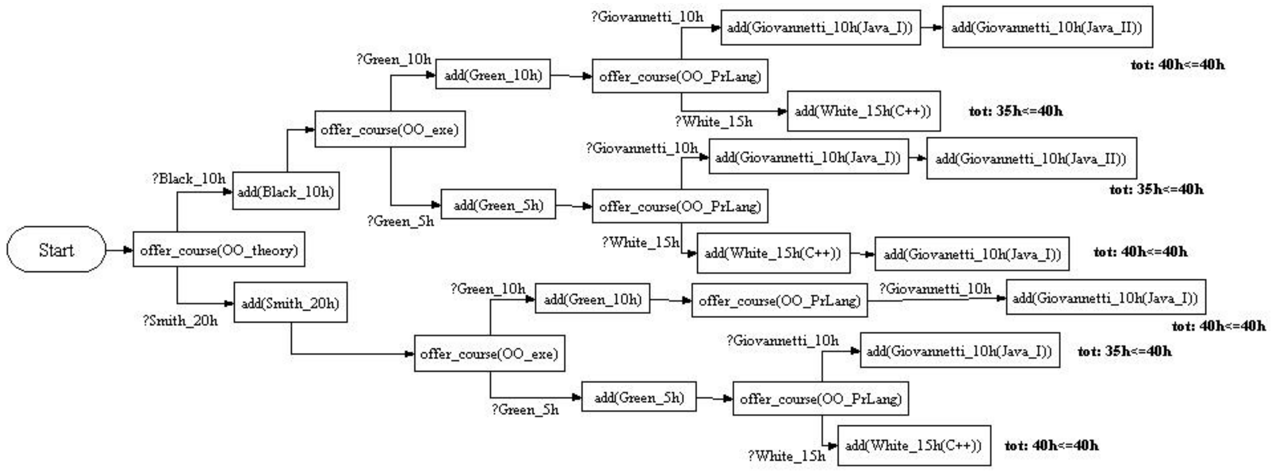

Figure 1 – An example of a generated plan.

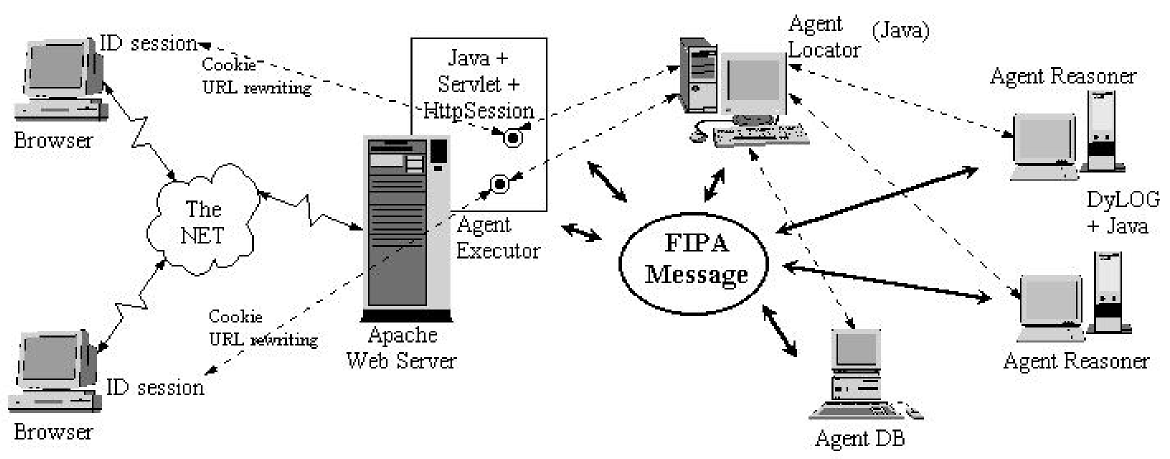

Figure 2 - A sketch of the WLOG architecture.