Software architecture of SETA, an adaptive Web store shell

L. Ardissono, A. Goy, G. Petrone and M. Segnan
Dipartimento di Informatica - University of Torino
Corso Svizzera 185 - Torino, Italy
{liliana, goy, giovanna, marino}@di.unito.it

1. INTRODUCTION

Personalization has become a key point in several applications. In particular, electronic commerce imposes strong adaptation requirements to personalize business to business, as well as business to customer interactions. This paper describes the software architecture of SETA, a prototype toolkit for the development of adaptive Web stores which tailor the suggestion of items and their presentation to the customer's characteristics. See [2] and [3] for a detailed description of the personalization techniques we exploited. Our system is based on the exploitation of knowledge representation and MAS technologies, which are essential to enhance the system's configurability on differentiated domains. An on-line demo of a prototype store created using SETA is available at the URL: http://www.di.unito.it/~ seta; this store presents telecommunication products, like phones and switchboards.

2. OVERVIEW OF SETA

Our system is based on a multi-agent architecture where specialized agents fill the main roles for the management of personalized interactions with customers [8]. For instance, a User Modeling Component (UMC) manages the user models and a Personalization Agent handles the generation of the Web catalog pages [1]. Knowledge representation techniques are exploited to improve the configurability of the toolkit: each SETA agent retrieves the domain-specific knowledge from its own knowledge base (or database) and can be instantiated on different domains. Moreover, agent-based technologies are used to enhance the extensibility and scalability of the system, which can extended to offer new functionalities in a rather seamless way (by adding agents, or by extending the functionalities of the existing ones).

The SETA agents are created by the Session Manager (a Servlet) at the Web store initialization time and persist during the system's life-span. No middle agents are used to support agent communication because, as the distinct roles to be filled in the architecture are fixed, each agent knows which other agents have to be contacted for requesting services and only needs their references, provided by the Session Manager immediately after the initialization of the store. The multi-user access to the Web store is managed by performing the session tracking within the Session Manager and forwarding the session-specific messages to the agents, who process such messages and perform the related activities.

Each SETA agent is composed of an interface, the "Dispatcher", handling the delivery and reception of messages, and of a set of user-session agents which maintain the session-dependent environments and carry on the activities to be performed within each user session. Each dispatcher acts as a wrapper and supports a uniform communication with the other SETA agents, by separating the communication flows related to the active sessions and forwarding any incoming message to the appropriate user-session agent. Moreover, the dispatcher creates and removes user-session agents, depending on the users currently connected to the store. Dispatchers and user-session agents support a simple management of parallel user sessions and session-dependent activities: in fact, the user-session agents have separate internal states and perform services and activities independently.

3. THE SETA AGENTS

Our approach is integrated in the environment of the ObjectSpace Voyager agent-building tool [9], which we used to wrap the SETA agents, enabling them to run in parallel and to communicate by means of synchronous and asynchronous messages [4]. Each Dispatcher is a Voyager Object and handles the incoming messages in parallel threads of execution, immediately invoking the appropriate user-session agent to perform the requested task. Figure 1 shows the class hierarchy defining the SETA agents: the "Dispatcher" class specifies that a SETA agent has a list of references of the Dispatchers to which it may send messages ("agentReferences"); moreover, it has a set of user-session agents ("userSessions"), devoted to the management of the session-specific tasks. The class also offers the methods for initializing a SETA agent ("init()"), setting the agent references ("addAgentReference()"), creating and removing a user-session agent ("addUserSession()", "removeUserSession()") and forwarding messages to a user-session agent, in order to process a session-dependent request ("forwardMessage()").

Each SETA agent extends the "Dispatcher" class and may override its methods and variables. This is the basis for the exploitation of specialized user-session agents, based on

AGENTS'01, May 28-June 1, 2001, Montréal, Quebec, Canada.

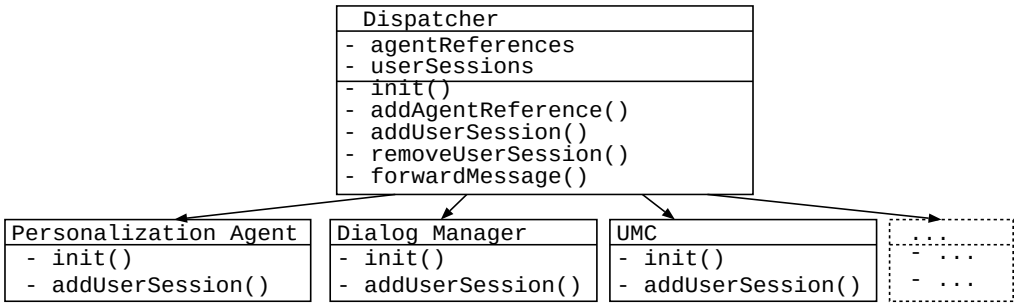

Figure 1: Class hierarchy defining the SETA agents.

different technologies, within the SETA agents. In our system, heterogeneous modules cooperate to the management of the Web store. For instance, some SETA agents, such as the Personalization Agent, are designed in a pure Object-Oriented style, as they have to perform tasks specified via explicit requests (production of pages). These agents are "stateless", as they can get the information about the overall state of the interaction by invoking suitable services on the other agents. Moreover, they handle services by responding to deterministic, though possibly complex, method invocations.

In contrast, other SETA agents are "stateful" and autonomously carry on various types of activities, in addition to the normal service provision ones. For instance, the User Modeling Component revises the user model during the interaction with the customer. This is necessary to have an up-to-date picture of the user's preferences available at each stage of the interaction. However, to support this dynamic revision, the UMC has to carry on the autonomous management of internal revision tasks, in addition to the provision of information about the user's data and preferences. To satisfy this requirement, we have designed the user-session agents exploited by the UMC as action-based agents, based on an explicit representation of the agent state and of the tasks to be performed. Each task is represented as an action with preconditions determining the state conditions where the task can be performed. The presence of an interpreter selecting the actions to be executed enables the autonomous management of the revision of the user models [4, 5].

The modularity of our architecture enabled us to integrate external software tools into the SETA agents. Such tools execute very specific tasks, by cooperating with the other components of the agents. For instance, the UMC exploits a rule-based system to maintain a structured representation of the interaction context and of the user's behavior [5].

4. CONCLUSIONS

We have described the implementation SETA, a prototype toolkit for the creation of adaptive Web stores [1, 4]. In the development of this system, we decided to exploit a basic and light agent-building tool, such as Voyager, to manage a seamless communication among agents, but we preferred to design our own infrastructure for developing the system agents because, as SETA is a specialized architecture for the creation of Web stores, it does not need the full capabilities offered by general-purpose agent-building tools, which typically offer facilities for agent communication, coordination, self-diagnosis, mobility, coordination of MAS to reach non-local goals and real time flexibility [6, 7, 10]. The described work is the evolution of a system developed at the CS Department of the University of Torino, within the project "Servizi Telematici Adattativi" of the "Cantieri Multimediali" initiative, funded by Telecom Italia. We are grateful to L. Console, L. Lesmo, C. Simone and P. Torasso for their comments to this work and fruitful discussions.

5. REFERENCES

[1] L. Ardissono, C. Barbero, A. Goy, and G. Petrone. An agent architecture for personalized web stores. In *Proc. 3rd Int. Conf. on Autonomous Agents (Agents '99)*, pages 182–189, Seattle, WA, 1999.

[2] L. Ardissono and A. Goy. Dynamic generation of adaptive Web catalogs. In *Lecture Notes in Computer Science n. 1892: Adaptive Hypermedia and Adaptive Web-Based Systems*, pages 5–16. Springer Verlag, Berlin, 2000.

[3] L. Ardissono and A. Goy. Tailoring the interaction with users in web stores. *User Modeling and User-Adapted Interaction*, 10(4):251–303, 2000.

[4] L. Ardissono, A. Goy, G. Petrone, and M. Segnan. Configurability within a multi-agent web store shell. In *Proc. 4th Int. Conf. on Autonomous Agents (Agents '00)*, pages 146–147, Barcelona, 2000.

[5] L. Ardissono and P. Torasso. Dynamic user modeling in a web store shell. In *Proc. 14th Conf. ECAI*, pages 621–625, Berlin, 2000.

[6] M. Barbuceanu and R. Teigen. Higher level integration by multi-agent architectures. In P. Bernus, editor, *Handbook of Information System Architectures*. Springer Verlag.

[7] J. Graham and K. Decker. Tools for developing and monitoring agents in distributed multi agent systems. In *Proc. of the Agents'2000 workshop on Infrastructure for scalable multi-agent systems*, Barcelona, 2000.

[8] N. Jennings, K. Sycara, and M. Wooldridge. A roadmap of agent research and development. In *Autonomous Agents and Multi-agent Systems*, pages 275–306. Kluwer Academic Publishers, Boston, 1998.

[9] ObjectSpace. Voyager. http://www.objectspace.com/index.asp, 2000.

[10] D. Steiner. An overview of FIPA 97. http://drogo.cselt.stet.it/fipa/#Papers, 1997.