Intelligent Agents supporting user interactions within self regulated learning processes

Carlo Alberto Bentivoglio[1], Diego Bonura[1], Vincenzo Cannella[2], Simone Carletti[1], Arianna Pipitone[2], Roberto Pirrone[2], Pier Giuseppe Rossi[1], Giuseppe Russo[2]

[1]Dipartimento di Scienze dell'educazione e della formazione, Università di Macerata [2] Dipartimento di Ingegneria Informatica, Università di Palermo

Keywords: Multi Agent Systems, Learning Management Systems, Self Regulated Learning, Natural language processing

Abstract

The paper focuses on the main advantages in the definition and utilization of an open and modular e-learning software platform to support highly cognitive tasks performed by the main actors of the learning process. We present in detail the integration inside the platform of two intelligent agents devoted to talking with the student and to retrieving new information sources on the Web. The process is triggered as a reply to the system's perception that the student feels discontented with the presented contents. The architecture is detailed, and some conclusions about the growth of the platform's overall performance are expressed.

1 Introduction

This work aims at integrating inside a unique framework the experiences about the design of advanced systems to support e-learning based on artificial intelligence methodologies.

The online learning environments are not to be seen merely as tools to support learning. They are components of a wider approach that is more "theoretic". A well-structured learning environment has to facilitate the user in the connection of different tools in order to build, to share and to change his/her level of knowledge (Rossi, 2006). The online environment is not a space but rather a place where emotional and relational factors are relevant as cognitive aspects. These considerations lead to the definition of an auto-poietic, flexible and networked environment. The defined environment must be able to help students in the definition of a learning path and has to give orientation in the learning processes with the proper aggregation of materials. This will lead to an augmented attribution in the meaning of the materials as shown in Rossi (2009).

The proposed approach will be used as a guideline to define the environment. The overall environment can be seen as the overlapping of three different networks:

1. the network of tools and materials
2. the network of writings
3. the interaction network between users in the environment

A possible way to relate the three networks defined above can be to highlight fragments of materials (wikis) or interactions (forums) related to the learning process inside the platform. The learning environment must support the actors of this process (mostly the tutors) and has to automate the process itself as much as possible. Some examples of the type of actions required are the definition of aggregate data from a forum analysis, tracking students' browsing inside the didactical materials, and deciding whether or not to moderate a forum about a particular topic. The overall complexity level of the environment which is no more a collection of simple Learning Objects requires an agent architecture in order to provide a high level of flexibility (Rossi *et al.*, 2009a; Rossi *et al.*, 2009b).

This work presents the integration of two new agents developed by some authors in the general framework. The mentioned agents deal with the conversation management and with the acquisition of new information in the system's knowledge base. The followed approach is the Self-Regulated Learning Theory (Zimmerman, 2001; Pintrich, 2000; Azevedo, 2008). The agents facilitate a self-regulated behavior in students, which take control of their own learning

process (Prensky, 2008). The same approach is followed w.r.t. tutors having to monitor and sustain the learning processes.

The article is organized as follows: The next section describes the reference architecture, while section III describes the two new modules. Section IV presents some conclusions about the topics.

2 System Architecture

The idea of designing and developing a new software architecture for tracking and monitoring learning activities derives from the two main problems we observed about current implementations. There is a considerable amount of data to be collected for each user session in a LMS, while the market presents solutions that are too heterogeneous in this regards. At present, there is no implementation which takes into account the possible sources of data. A modular system is needed to cope with the problems mentioned above, and we claim it can be implemented as an extensible software architecture (Fig. 1).

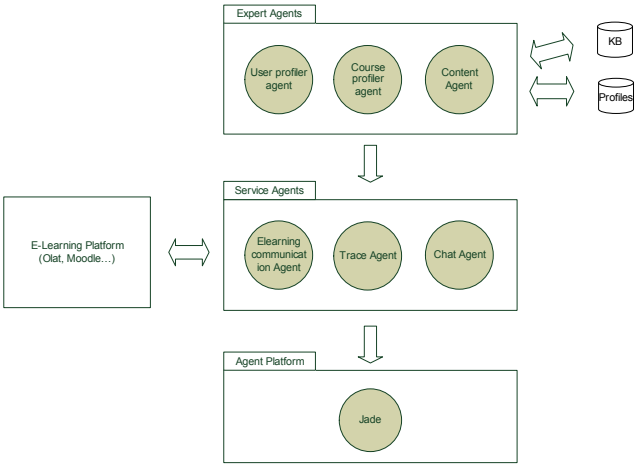

Fig. 1 - The multi-agent architecture from a computational point of view.

The system is hence made up of four main components:

- a logger adapter based on Messaging techniques;
- a Multi Agent System (MAS);
- a Jabber, XMPP client;
- different web services as interfaces.

The first component of the software architecture, the Message Broker, has been implemented through ActiveMQ, a software service for message rou-

ting able to exchange messages decoupling the sender (LMS) and the receiver (Tracking and Monitoring system). This solution has many advantages, first of all, its simplicity. A set of instructions is actually needed to develop a simple adapter for each different LMS system. Second, the adapter is just an extension of the logger or audit manager that is present on every LMS platform so the introduction of that new component has no effect on the existing systems. This way, we can catch and send every single user's action in a standard, XML-based format. The messaging technology is highly reliable and scalable for our environments, making it possible to exchange a high volume of messages per second. These messages contain different information about platforms, users, roles, and actions to be carried out i.e. posting a new message in a forum. This information is very useful for the analysis and monitoring of students' progress.

We are currently running our first adapter written for the OLAT LMS Platform. It extends the existing audit manager and makes it possible to track every single user's action on the course and specific information for particular course elements such as forums and chat.

When the Message Broker receives a new message, the information is routed to our Multi Agent System. This MAS application has been developed using JADE (Java Agent Development framework) and is composed by different agents that interoperate in order to carry out profile analysis and give real-time feedback channeled toward the LMS.

In the presented architecture the MAS is composed of three different software layers matching three abstraction layers: the bottom level is the middleware software supporting the execution of the several agents coded in the system. In the middle layer there are agents communicating with the LMS and providing services to higher-level agents. In the top layer we have "expert agents" carrying out specific tasks such as user profiling by processing data through Fuzzy logic and Bayesian networks.

- Agent platform: we have adopted JADE as middleware. This framework is FIPA compliant[1] and supports standard ontologies (RDF/OWL). Moreover, it supports communication among agents and towards external systems.
- Service Agents: the agents at this level have to communicate with the eLearning platform through web-services and messages queues. They receive data related to users tracking or communicate in real-time with the user by chat – using, for instance, the XMPP protocol. All communications between LMS and the MAS are initiated and handled by agents in this layer.

[1] Specifications of FIPA standard are produced by the Foundation for Intelligent Physical Agents and are published at http://www.fipa.org/specs/fipa00023/index.html

- Expert Agents: these agents are specific to support learning; in this layer the agents can use KB and ontologies in order to run semantic processing algorithms and make decisions based upon user profile analysis and course-specific information stored into the LMS. These processes may be supported in the future by inference engines (Drools), Bayesian networks and fuzzy logic.

The multi-level architecture described above makes it possible to process data both synchronously and asynchronously; for instance, through the service-level agents it is possible to communicate in real-time with the student while he/she is opening a forum and reminding him/her the importance of collaborative work i.e. via the chat interface. We have already implemented different agents behaviors based on AIML[2] standards (Artificial Intelligence Markup Language) with a generic knowledge repository. We have planned to build a dedicated web interface for e-tutors to support the building of a knowledge repository. This way the e-tutor will have the possibility to add course-specific information to the knowledge repository.

At the same time Expert Agents support intensive computational analysis by scheduling CPU-intensive calculations during server off-peak hours. Such agents are in charge of calculating users' and course profiles in addition to collective qualitative and quantitative indicators (for instance the behavior of a student community in a forum). More precisely, they build and update user profiles, course profiles and special course element profiles considering completed user sessions; the agents elaborate user navigation paths and make it possible to compare information related to different dates.

In order to make collective statistics, a set of indices is daily computed to guess the current status of the forum according to a model of forum evolution (Bentivoglio, 2009). Such states depend on the behavior of the community. They show, for instance, whether a specific discussion is engaged or they are responding to the tutor seminal post. This way it is possible to make a situated discussion with the users improving the interaction with the chatbot.

All the extracted information is stored and made available externally through special interfaces based on Web 2.0 standards. All the retrieved information data from our tracking and monitoring platform is available via open web services. We have deployed different web services to make possible further interaction between external analysis tools, and to support service orchestration techniques. These WS provide also data to purpose-built web 2.0 interfaces that are designed for teachers and tutors, who can hence evaluate students' profiles and progress. This architecture also makes it possible to analyze the collected data with external data mining and business intelligence tools.

[2] AIML Home Page available at: http://www.alicebot.org/aiml.html

3 The implemented agents

The so-called "Chat Agent" and "Content Agent" belong to the TutorJ framework presented by some of the authors (Pirrone *et al.*, 2008). TutorJ is an Intelligent Tutoring System (ITS) whose first version was developed to support students training on the Java programming language. The last version of TutorJ is focused on Arts.

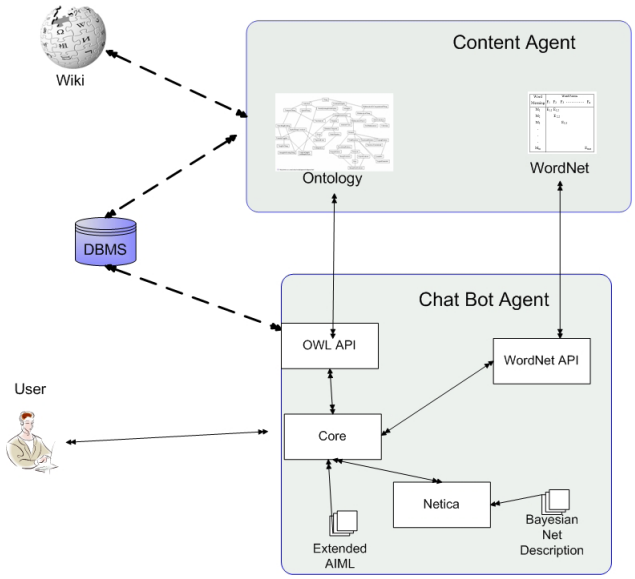

Fig. 2 - The interaction between the user, the chat agent and the content agent modules

TutorJ has been designed as a cognitive system, on the basis of the Human Information Processor Model (HIPM). According to this model a cognitive agent is made up up three kinds of modules: perceptive modules, sense-motor modules, and strictly cognitive modules.

The most important tasks of the modules are: natural language interaction, learning content management, learning path planning, gathering of new knowledge. Fig. 2 shows a simplified version of the TutorJ architecture focused only on the two agents presented in this work. The Chat Agent interacts with the user. To reach this goal, it plans a mixed modality conversation either using natural language or through graphical components for a visual interaction.

The Content Agent manages the domain knowledge through a knowledge base that codifies its symbolic memory about the domain as an OWL ontology. Moreover, we have developed a Latent Semantic Analysis module (Landauer

& Laham, 1998). It manages a sub-symbolic memory, which merges text documents, main domain terms, and concepts into a common space, building a unified representation of the conceptual domain. This space joins the symbolic description of the domain with the corpora.

Finally, the agent is able to obtain new knowledge from the Internet when it perceives the student is unsatisfied with the material obtained as the interaction result. We use WordNet (Fellbaum, 1998) for terms disambiguation.

3.1 The Chat Agent

The Chat Agent architecture relies on classic conversational agents. It has to interact with the user in natural language or through a graphical interface. A conversational agent is a software artifact that makes possible natural language interaction with the computer.

Typical examples of such systems are chatterbots, or chatbots. Eliza and A.L.I.C.E.[3] are probably some of the most famous examples. Both of them are simple stimulus-reply systems, without state management, or interaction planning. Moreover, they were designed only to interact linguistically. The proposed chat agent, on the other hand, combines a stimulus-reply state automaton and a goal-driven probabilistic agent, defined as a Partially Observable Markov Decision Process. It can react to the stimuli from the user in a predefined manner or through a conversation planning process. The agent owns a library of actions. Each action can be joined to a goal. There are two types of goals: immediate goals, and global ones that are intended for the whole conversation. An immediate goal can be reached in a single dialogue step, while achieving a global goal spans over the whole conversation. This way, we have inserted elements of Pragmatics in the description of dialogs. Each sentence is considered as a linguistic act with its own goals. In Pragmatics many goals have been defined for interaction acts. Each action can also be described through a generic attribute of type that is useful in unforeseen situations. It could be used, for instance, to modify the chatbot's behavior according to its mood or to distinguish between different interaction modes that is graphical or linguistic one. This modality uses a part of the agent's long-term memory, which stores the procedural knowledge of the system. In fact, this knowledge includes the interaction structures too.

Moreover, the working memory of the Chat Agent stores the stimuli from the environment together with the state of the user-system conversation.

3.2 The Content Agent

One of the issues of the conversation between the system and the users is

[3] A.L.I.C.E. Home Page available at: http://alicebot.org/about.html

that the conversation itself is not explicitly defined by its topics. To solve this problem, an extension of the flow control model of the Chat Agent has been proposed: the extension starts from the relations between the topics of the treated domain.

The chosen artifact for the domain model is the ontological model: for this reason, the model of the conversation must be able to navigate the domain ontology autonomously, browsing from concept to concept through the relations between them. This browsing is coded in AIML, so specific tags that allow loading an OWL ontology have been defined in order to retrieve the properties of a concept in the ontology and to move along its relations. These tags are compliant to the OWL API.

The Content Agent implements all the main functionalities to retrieve information from the ontology.

Each node in the ontology may be related also to external contents, and the learner can express the intention to retrieve such materials. In general, this intention is not explicit, being rather a disappointment for the contents already obtained from the system.

The Content Agent implements a strategy to retrieve new knowledge from external information sources. The task to expand automatically the knowledge base of a system from external resources consists of two steps: first, semi-structured knowledge sources (we use wikis) have to be transformed in structured knowledge sources, which can be represented in ontological form. In our implementation, we make use of semi-structured parts of the wiki page (i.e. the TOC) to locate the most interesting sections. Then we parse text, and search for some suitable linguistic patterns that are used in relation to the topic that needs deepening. Finally, new knowledge has to be integrated in the existent knowledge, and validation has to be performed. This problem is stated as "Ontology matching" that is to define multiple correspondences between some concepts in the base ontology with some other concepts in the incoming one. Our implementation of the matching procedure is based on projecting new documents in the LSA space, while their distance form some "known topics" is evaluated. The entire procedure is described detailed in (Pirrone *et al.*, 2008b; Russo *et al.*, 2009).

Conclusions

The main advantage of an open modular LMS architecture is the interoperability of its components that may be used to some extent in learning platforms based on heterogeneous systems. Components reuse is an efficient design strategy both in experimental, and commercial platforms.

However, the real innovation in this work is the pedagogic-didactical para-

digm of self-regulated learning, whose main goal is to encourage a process of conscious and reflective learning by SRL based tools running on the presented architecture.

Looking forward to extending these tools to facilitate the task of tutors, we are expected to merge statistical analysis, performance indicators – both quantitative and qualitative - and feedbacks within a "Pedagogical Dashboard" that allow teachers and tutors:

- to monitor the activities of the course and the participation;
- to schedule the routine interventions (e.g. send automatic messages to encourage participation).

Moreover, the Dashboard allows teachers and tutors to explicit the educational interventions with the highest relevance such as user profiling, and then to customize the learning process through the use of more specific agents.

REFERENCES

Azevedo, R. (2008), *The role of self-regulation in learning about science with hypermedia*. In D. Robinson & G. Schraw (Eds.), Recent innovations in educational technology that facilitate student learning, 127-156. Charlotte, NC: Information Age Publishing.

Bentivoglio C.A. (2009), *Recognizing Community Interaction States in Discussion Forum Evolution*. In Cognitive and Metacognitive Educational Systems AAAI Fall Symposium, 20-25.

Fellbaum, C. (1998, ed.), *WordNet: An Electronic Lexical Database*. Cambridge, MA: MIT Press.

Pilato G., Pirrone R., Rizzo R. (2008), *A KST-Based System for Student Tutoring*. Applied Artificial Intelligence, 22(4), 283-308, ISSN: 0883-9514.

Pintrich, P. R. (2000), *The role of goal orientation in selfregulated learning*. In Boekaerts, M.; Pintrich, P. R.; and Zeinder, M., eds., Handbook of self-regulation. San Diego, CA: Academic Press. 451–502.

Pirrone R., Cannella V., and Russo G. (2008), *Awareness Mechanisms for an Intelligent Tutoring System*. Biologically Inspired Cognitive Architectures. AAAI Fall Symposium (FS-08-01) ISBN 978-1-57735-396-6, 146-151.

Prensky, M. (2008), *The Role of Technology in teaching and the classroom*, Educational Technology, Vol. 48(6), 1-3.

Rossi, P.G. (2006), *Design and ongoing monitoring systems for online education*. In Proceedings of On Line Educa, Berlin.

Rossi, P.G. (2009), *Ambiente di apprendimento con elementi di artificial intelligence*. JE-LKS. Journal of E-Learning and Knowledge Society, vol. 5(1), 65-75.

Rossi P.G., Carletti S., Bonura D. (2009a), *A Platform-Independent Tracking and Monitoring Toolkit*, AAAI 2009 Fall Symposium on MCES, Arlington VA, USA, November 2009, 76-80.

Rossi P.G., Carletti S., Bonura D., Bentivoglio C.A. (2009b), *A Multi-Agent Environment for Tracking and Monitoring Learning Activity*, AI*IA 2009, Reggio Emilia, Italy, December 2009, 71-78.

Russo G., Pipitone A., and Pirrone R. (2009), *Acquisition Of New Knowledge In TutorJ. Cognitive and Metacognitive Educational Systems*. AAAI Fall Symposium (FS-09-02) ISBN 978-1-57735-436-9, 81-86.

Landauer, T., and Laham, P. F. D. (1998), *An Introduction to Latent Semantic Analysis*. Discours Processes 25, 259–284.

Zimmermann, B. J. (2001), *Theories of Self-Regulated Learning and Academic Achievement: An Overview and Analysis*. Lawrence Erlbaum Associates. chapter 1, 1–37.