Cloud Computing in Smart Educational Environments: application in Learning Analytics as Service

Manuel Sánchez[1], Jose Aguilar[2], Jorge Cordero, Priscila Valdiviezo-Díaz, Luis Barba-Guamán, Luis Chamba-Eras[3]

[1] Universidad Nacional Experimental del Táchira, San Cristóbal, Venezuela
mbsanchez@unet.edu.ve
[2] Universidad de Los Andes, Mérida, Venezuela
Prometeo researcher, Universidad Técnica Particular de Loja, Loja, Ecuador
aguilar@ula.ve
[3] Universidad Técnica Particular de Loja, Loja, Ecuador
{jmcordero, pmvaldiviezo, lrbarba, lachamba6}@utpl.edu.ec

Abstract. In this paper, we present an extension of a Middleware for Smart Educational Environments based in agents, using the paradigm of Cloud Computing. In that sense, we detail the Middleware components, which enable the process of management of the Cloud Computing. We also present the utilization of this Middleware to provide services on the cloud about task of Learning Analytics that allow processing of data of students and learning environments, to understand and optimize the learning processes.

Keywords: cloud computing, smart educational environment, learning analytics.

1 Introduction

In educational environments, the Ambient Intelligence (AmI) allows to follow the dynamics of teaching and learning of users [1, 2, 3, 4], such as: supervise the student performances, provide digital contents according to the student's learning style, link students studying similar topics, etc. On the other hand, cloud learning (C-Learning) allows a reuse of learning resources in a distributed manner. C-Learning provides available educational services in the cloud, using the mechanisms and tools that provides cloud computing. In particular, the AmI for education based on the cloud could be used as a space, where the technologies of the cloud and ubiquitous, help the learning processes in an unobtrusive manner, to improve their scalability and integration capabilities [5, 6, 7].

In previous work, we have developed a middleware based on multi-agent system, called AmICL, to support smart educational environments [8, 9, 10]. This Reflective Middleware allows to manage an Intelligent Environment (IE) of Learning. This middleware proposes five levels, one for the management of the multi-agents community, other to manage the access to services, applications, etc., and the last one to characterize the different components (software and hardware) of AmI for education. This article aims to present the detailed design of the implementation of

Á. Rocha et al. (eds.), *New Advances in Information Systems and Technologies*,
Advances in Intelligent Systems and Computing 444,
DOI 10.1007/978-3-319-31232-3_94

the paradigm of cloud computing in AmICL, as well as verify their operation in a use case. Basically, the use case is linked to providing cloud services linked to Learning Analytics (LA) tasks. LA refers to the processing of data of students and learning environments, in order to understand and optimize their learning processes [11].

This paper is organized as follows: Section 2 provides detailed specification of the components in AmICL. Section 3 presents the components in AmICL responsible for the management of the cloud computing. Section 4 describes the use case, design and utilization of the services and the context of the AmI where they are used. Finally, Section 5 has some conclusions and acknowledgments.

2 Middleware for intelligent cloud learning environments (AmICL)

In this work, we are going to use the middleware AmICL proposed in [8, 9, 10]. This Middleware proposes five levels (see Fig. 1). The IE physical Management layer represents the different devices in the environment, defined as agents. MMAL is composed by a multi-agent community to support the execution of multi-agents applications. This level follows the FIPA standard (see [12]). The Services Management Layer (SML) has the responsibility of finding, searching, etc., services required by the applications. ILL represents the different software components in the educational platform. Particularly, it has two agents: the first is a profile agent to represent each student (SPA), and the second is the tutor agent (TA) to represent the professor. Finally, IPL is where are deployed the different devices and software of a smart classroom (SaCI). AmICL is an Autonomic Reflective Middleware that enables the integration of objects in the educational environment, with educational resources in the cloud, in a flexible an adaptive way, so that objects can adjust their behavior according to the context, and the requirements of the users.

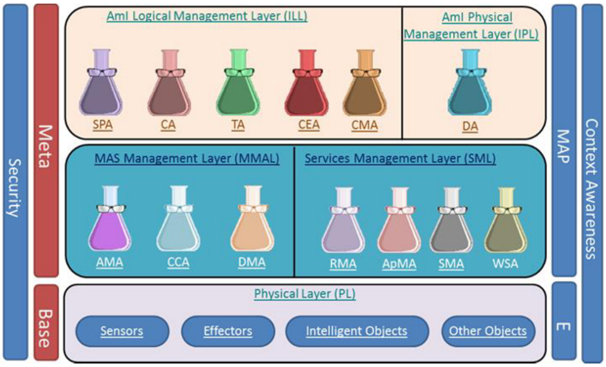

Fig. 1. AmICL architecture.

The difference between AmICL and other similar researches is in the way that combines the AmI with academic services that provides the cloud learning paradigm (C-Learning). In AmICL, academic cloud services are combined with intelligent and non-intelligent objects in the environment, to adapt and respond to the teaching and learning requirements of users. In particular, the new aspect that fuses an AmI with the cloud computing paradigm is SML. This layer has some agents which are an extension of FIPA standard, because the layer not only handles the resources and applications, but also educational services (locals and in the cloud), to adapt the AmI to current learning conditions required by its users.

In general, the agents of the middleware are, in MMAL: Agent Management of Agents (AMA), CCA (Communication Control Agent) and DMA (Data Management Agent, they are defined in [12]; SML is defined by the Services Management Agent (SMA), the Web Service Agent (WSA), the RMA (Resource Management Agent) and the ApMA (Applications Management Agent). The agents of ILL are: the Student's Profile Agent (SPA), the Collaborative Agent (CA), the Tutor Agent (TA), the Content Management System Agent (CMA), and the Collaborative Environments Agent (CEA). Finally, the IE Physical Management Layer is composed of a single type of agent, the Device Agent (DA).

3 Specification of the Cloud Management Subsystem of AmICL

In this paper, we define the agents that compose SML. Therefore, we define a general type of service agent (framework) to represent the different services on the cloud, which can be invoked by the educational platform. Additionally, we define a subsystem to manage the services provides on the cloud for the other components of AmICL via communications protocol based on the service-oriented architecture (SOA) paradigm. Specifically, this layer defines two agents (they are the bridge between the AmI and the cloud):

- *SMA*: controls, records and manages Web Services (WS) available on the system (whether they are in the cloud or locally); so when an agent requires a specific service it should contact this agent to locate the WSA that characterizes that service. The SMA is a hybrid between what is known in WS as UDDI (Universal Description, Discovery and Integration), which is a platform-independent framework to describe, discover, and integrate WS; with an enterprise service bus (ESB) (this software architecture model is used in the design and implement communication between applications based in the SOA paradigm).
- *WSA*: it is the logical representation of a Web Service. It characterizes the WS, knows how to invoke them (it has their interfaces) and what are the requirements necessary to access the WS. The WSA has access to the services description file (WSDL), and creates a local proxy to consume the services' methods. There is one instance of it for each Web Service on the SMA.

In this section, we will use the MASINA models [13] to specify the agents. Basically, we are interested in the models of agents (for detail each agent, etc.), and their tasks (define the activities to achieve their objectives or provide services).

SMA Agent Model

Table 1 shows the SMA Agent model. Particularly, SMA is a goal-based intelligent agent, who learns how to localize Web Services based on past experiences.

The goals of the SMA are facilitate the: access, location (service discovery) and consumption of the WS in the cloud, such that other agents can make use of these services in a transparent manner. This agent offers three services; the first is adds a WS into the SMA's WS directory; the second is locate a WS by its category, and finally, to remove a WS from the SMA's WS directory.

Table 1. SMA description.

Service Management Agent
Type: Goal-Based Intelligent Software Agent
Roles: Manage Web Services that are available in the IE.
Description: This agent manages (registering, locating, removing) the WS that will be used by other system agents. This agent characterizes what is known in SOA as UDDI, providing to the agents community, location and consumption of the Web Services required.

SMA Tasks Model

Table 2 shows the relationship between SMA services and its tasks.

Table 2. SMA Tasks and Services.

SMA-S1. Register a Web Service
Receiving the Information of the Web Service (WSDL, service name, web service methods description, category, etc.).
Checking the connection to the Web Service.
Registering the Web Service in the directory.
SMA-S2. Locate a Web Service
Receiving data of the Web Service to localize.
Locating the WSA requested.
SMA-S3. Remove a Web Service
Receiving data of the Web Service to remove.
Removing the WSA that characterizes that Web Service.

The first task, “Register Web Service” (SMA-S1), records the data of a new WS (see Table 3).

Table 3. Receiving the Information of the Web Service.

Task Receiving the Information of the Web Service
Objective: Receive data from the Web Service that will be registered in the SMA's directory.
Pre-condition: N/A.
Frequency: When SMA-S1 starts.
Description: This task is needed to receive the information of the new Web Service that is going to be registered in the SMA's directory.

So, the SMA-S1 will perform a task that will check whether the WS that is being registered is online or not. To do that, SMA will use the WSA's capabilities, which in turn will use the WSDL file to check the status of the WS. The next task that SMA-S1 performs is to register the WS in the knowledge base of the SMA (The rest of task models are in [10]).

The tasks of the service “Locate Web Service” (SMA-S2) of SMA are described in [10]. First, the task “Receiving data of the Web Service to localize” receives the information that is needed to localize the WS that has the ability to meet the requirements requested. Once the data is received, the next task is responsible for determining through a bidding process, which WSA has the ability to meet the requirements requested. A bid process performs this task because each WSA knows its category and its function, so only those WSA that can meet the requirements will make a proposal. The third service offered by SMA is responsible for "Remove a Web Service" (SMA-S3). This service performs two tasks. The first will receive data of the WS to remove, and the second will update SMA's knowledge base (their task models are in [10]).

WSA Agent Model

Table 4 shows the WSA agent model. This agent is of type reflex simple, because its fundamental role is to invoke the WS that characterizes.

Table 4. WSA Description.

Web Service Agent
Type: reflex simple software agent.
Roles: invoke the WS that characterizes.
Description: This agent is responsible for invoking the method to consume the WS, according to the information requested.

The goal of the WSA is serves as a local proxy to access the remote Web Service (endpoint), so can consume the WS. The WSA offers two services: the first invokes a WS method by providing the method input data, while the second permits to obtain the WS description, which can be used to know what type of information is needed to invoke the service.

WSA Task model.

Table 5 shows the relationship between the services and tasks of WSA.

Table 5. WSA Task and Services.

WSA-S1. Invoke Web Service	
T1.	Receiving input data to invoke the WS.
T2.	Inferring method's name that will be invoked.
T3.	Invoking the WS Method.
T4.	Send execution result to the requesting agent.
WSA-S2. Get Web Service Description	
T1.	Analyze WSDL file.
T2.	Send Web Service's description to the requesting agent.

WSA-S1 service performs four tasks (Their task models are in [10]). The first receives the input data to the WS and verifies that the data have the appropriate format. The second task infers the method called of the WS, based on the input parameters and the type of operation to be performed (creating, requesting, updating or deleting). The next task calls the WS method. Finally, the WSA-S1 service performs the task "Send execution result to the requesting agent" to send the results of the invocation of the WS requested. The WSA-S2 service must perform two tasks: The first task analyzes the WSDL file to determine the input and output parameters of the WS method. The second task WSA-S1 sends the data extracted by the first task.

Registration of a Service on AmICL.

In general, to use a WS in AmICL, it must be registered in the SMA. To register a WS in the SMA, which means can be discovered and located, the following information is required: i) **Service Name:** specifies the name of the WS. ii) **Business**

Name: specifies the business entity that will offer the WS. iii) **WSDL Location:** specifies a URI that points to a WSDL document that contains a description of the WS. iv) **Web Service Description:** specifies a XML document to describe the operation type of each WS Method. v) **Category:** specifies the category into which the WS is located (Academic, Recognition, etc.) An example of the XML document is:

```
<Service name="Affective Recommender System">
  <description> Content Recommender System based on the student's affective state
  </description>
  <method name="search">
    <description>
      obtains recommendations of contents
    </description>
    <type>request<type>
  </method>
  <method name="register">
    <description>
      Register a new content in the system
    </description>
    <type>create</type>
  </method>
</Service>
```

The “type” element in the XML file is used to know which method WSA should call according to what want to perform. It can take the following values: create, request, update or delete (CRUD). This information does not come within the WSDL file, but it is necessary to take decisions when a WSA is being located or invoked.

4 Experiment

4.1 Case Study

We analyze the case study of an online tutoring process in AmICL. This process adapts the online tutoring requirements of a specific session. In this case, the teacher gives a class and proposes some practical activities to students, to be developed in class. AmICL must be able to act proactively to help users to develop their activities, and access to academic resources available in the environment, combining cloud services with AmI. For this case, it is assumed that in the intelligent classroom there is a smart board to project slides, a student board can display the image of students virtually connected, cameras and microphones to capture images and sounds. In addition, each student has access to a computer where developed certain activities proposed by the teacher. The computer is used to monitor the student activity. Similarly, the environment has a Virtual Learning Environment (VLE) application and a Web content recommendation service.

The online tutoring process begins when the users enter to the VLE. Once that VLE has the UID of the different users in this season, it instantiates the corresponding agents for each user. AmICL determines that it must prepare the smart board in the intelligent classroom. Thus, TA locates the course data and plans the class; with that information, it locates today class’ slides through a storage service (in the course plan, it is indicates where the slides are stored, and this way it can call the RMA).

The slides file (managed through the RMA) is sent to the smart board, and then the DA (Smart Board) is prepared to begin class when the teacher gives the order. If the user is a student, the academic data is retrieved to determine the learning style and usage history of the environment for the group of students in the AmI. Thus, VLE invokes a clustering service to define the groups of students (each group is a pattern). With this data, VLE asks the recommender system (it is inside of the RMA) for activities, digital contents, etc. to each group of students. In this case, each group is a pattern or style of learning to be exploited by the recommender system, to search more accurate information. The clustering process allows an intelligent search (that is carried out by the recommender system) of learning resources, which are shown into the environment by the smart board according to the planning defined by VLE. Then, the students interact with these learning resources via the smart board, and TA monitors the work of the students. This is a cyclical process that is done in each tutoring session. At the end, VLE establishes a student score (evaluation), and updates the learning profile of the students in function of these results (learn). Additionally, VLE establishes an online tutoring model of the process to analyze the elements (chat, email, etc.), activities, etc., used during the section. For that, it invokes a data mining service, which is going to determine a descriptive model using the respective agents of SML.

Examples of the LA Services

In the conversation are invoked two LA services: to define the patterns of style learning of the students (clustering task) and to build the online tutoring process model. To test the LA services in AmICL, they have been instanced in the Universidad Técnica Particular de Loja (UTPL), where one of the main activities is the distant education. In the UTPL the student is the central actor in the educational process, and a teaching team, the tutorials, the resources learning and new technologies, mediate the learning process. UTPL uses for online tutoring: a VLE based on MOODLE platform, a Video Conference System, e-mail, phone tutoring, etc. The students in distance modality, through the forums, chats and video conferencing in the VLE, can get points. The information about the interaction of the users of VLE is collected in a file.

Building Service of the Pattern of the Online tutoring process.

Service description

This service builds a descriptive model based on association rules, to identify patterns of the interactions in the tools involved in the online tutoring process through the VLE. This service support the data on student interaction during the process tutoring online through VLE, to understand the learning process that is developing in this environment and optimize the use of tools used as forum, chat, among others. We use this file to determine a descriptive model that characterizes the use of the tools on the platform during the online tutoring process. To obtain the descriptive model, The R tool was used (its “arules” library), and 8 attributes were used: Ads, Learning Resources, Forums, Tasks, REAS, Chats, NumMssEnv, NumParticipantesChat. The best results obtained are shown on the Fig. 2 (10 rules).

```
Best rules found:

 1. Chats='(-inf-0.5]' 37 ==> NumParticipantesChat='(-inf-6.6]' 37    conf:(1)
 2. REASdoc='(-inf-1]' Chats='(-inf-0.5]' 28 ==> NumParticipantesChat='(-inf-6.6]' 28    conf:(1)
 3. Chats='(-inf-0.5]' NumMensajesEnvEst='(-inf-7.2]' 27 ==> NumParticipantesChat='(-inf-6.6]' 27    conf:(1)
 4. Tareas='(-inf-2]' Chats='(-inf-0.5]' 25 ==> NumParticipantesChat='(-inf-6.6]' 25    conf:(1)
 5. Chats='(1.5-2]' 34 ==> Tareas='(-inf-2]' 33    conf:(0.97)
 6. Chats='(1.5-2]' 34 ==> NumMensajesEnvEst='(-inf-7.2]' 33    conf:(0.97)
 7. Chats='(1.5-2]' NumMensajesEnvEst='(-inf-7.2]' 33 ==> Tareas='(-inf-2]' 32    conf:(0.97)
 8. Tareas='(-inf-2]' Chats='(1.5-2]' 33 ==> NumMensajesEnvEst='(-inf-7.2]' 32    conf:(0.97)
 9. Chats='(1.5-2]' 34 ==> Tareas='(-inf-2]' NumMensajesEnvEst='(-inf-7.2]' 32    conf:(0.94)
10. Recursos='(-inf-5]' REASdoc='(-inf-1]' NumMensajesEnvEst='(-inf-7.2]' 26 ==> Tareas='(-inf-2]' 24    conf:(0.92)
```

Fig. 2. Best rules obtained.

Specification as WSA

The registration of this service in AmICL is:

Table 6. Building Service of the Pattern of the online tutoring process.

Service
Name: *Service of the Pattern of the Online tutoring process*
Business Name*: NA*
WSDL location: *defined by the SOA platform*
Description: *Service of obtaining patterns in online tutoring process based on models descriptive*
Category: *Data Mining*

And the XLM document is:

```
<Service name="Service of the Pattern of the Online tutoring process">
  <description> Service of obtaining patterns in online tutoring process based on models descriptive.
  </description>
  <method name="build of the descriptive model">
    <description>
      obtain patterns of user interaction in the VLE
    </description>
    <type>search<type>
  </method>
  <method name=" Analysis of rules">
    <description>
      interpretation of rules
    </description>
    <type>identify</type>
  </method>
</Service>
```

Service of students clustering

Service description

We used the K-means clustering algorithm and the tool WEKA, to identify groups of objects with similar characteristics. Additionally, we use the same data from UTPL. For training, we divide the dataset with 80% for training and 20% for the test. As is noted in Fig. 3, there are three groups with a large supply of ads, but there is a clear distinction between them in relation with the utilization of the tools (chats, messages, etc.) by the students in their interactions.

Attribute	Full Data	Cluster# 0	1	2
	(95)	(24)	(21)	(50)
Anuncios	22.6105	25.1667	21.5238	21.84
Recursos	12.6	8.75	14.4762	13.66
Foros	3.9789	3.7917	3.4762	4.28
ComentProfEst	0 a 0	14 a 65	5 a 78	0 a 0
Tareas	1.8421	0.8333	0.8095	2.76
REASdoc	1.1368	0.7083	1.1905	1.32
Chats	1.1263	2.2917	1.5238	0.4
NumForumSentEstudent	4.2316	2.75	8.9524	2.96
NumMensajesEnvEst	5	3.2083	5.1429	5.8
NumParticipantesChat	13.3474	18	38.0952	0.72

Fig. 3. Results of the clustering problem.

Specification as WSA

The registration of this service in AmICL is:

Table 7. Building Service of the Pattern of the online tutoring process.

Service
Name: *Student's Clustering*
Business Name: *NA*
WSDL location: *defined by the SOA platform*
Description: *Service of obtaining student patterns in the process of online tutoring.*
Category: *Data Mining*

And the XLM document is:

```
<Service name="Service of the Pattern of the Online tutoring process">
  <description> Service of obtaining similar behavior patterns in the process of online tutorial.
  </description>
  <method name="Clustering model">
    <description>
      Obtaining of patterns behavior similar in the process of online tutorial.      </description>
    <type>search<type>
  </method>
  <method name="identify user profile">
    <description>
      Obtaining of profiles based on the interaction in the VLE      </description>
    <type>cluster</type>
  </method>
</Service>
```

5 Conclusions

In this work, we have shown the functionality of the cloud component of AmICL, to guarantee a cloud learning. AmICL mixes the benefits and capabilities such as reflective, autonomous and context aware of AmI, with capabilities that provides a learning environment in the cloud with academic services, which can be accessed from anywhere. This result provides improving resources and academic services in a learning environment. Moreover, it allows adapting the environment to the academic needs of users.

In particular, the system allows the invocation of educational WS to be executed by smart objects in the AmI, in transparent manner to users, to improve and adapt their operations to the requirements of these users. In addition, the reflective and contextual awareness capabilities of AmICL allow the system to adapt itself to the needs of the environment, in order to act intelligently to each situation that arises. Thus, AmICL used the intelligent capabilities related with reflection and analysis of the context in the middleware, using the LA paradigm, to adapt appropriately to situations and requirements that occur in the AmI.

We have tested LA services on the cloud for complex tasks, which are used by AmICL to improve the quality of the learning process. In general, the scalability and flexibility properties of an AmI are very good solved in our model. The next work must analyze the emergence and the self-organization of AmICL, and the specific problems of implementation (multi-agent platform, communication protocols, etc.).

Acknowledgments. Dr Aguilar has been partially supported by the Prometeo Project of the Ministry of Higher Education, Science, Technology and Innovation of the Republic of Ecuador. This work has been partially supported by the UTPL Project entitled: "Medios de Gestión de Servicios (Middleware) Inteligentes para Entornos de Aprendizaje Virtual".

References

1. Mhiri, F., Ratté, S.: AARTIC: development of an intelligent environment for human learning. SIGCSE Bulletin, vol. 41, nº 3, pp. 359-359 (2009)
2. Mikulecký, P.: Smart environments for smart learning. In 9th International Scientific Conference on Distance Learning in Applied Informatics DIVAI, Sturovo, Slovakia (2012)
3. Shi, Y., Qin, W., Suo, Y., Xiao, X.: Smart classroom: Bringing pervasive computing into distance learning. Handbook of Ambient Intelligence and Smart Environments, Springer, p. 881–910 (2010)
4. Stenvall-Virtanen, S., Nordell, K.: Smart Environments: Technology, Protocols and Applications. Finland, Tech. Rep.: University of Turku (2014)
5. El-Bishouty, M., Ogata, H., Rahman, S., Yano, Y.: Social Knowledge Awareness Map for Computer Supported Ubiquitous Learning Environment. Educational Technology & Society, vol. 13, nº 4, p. 27–37 (2010)
6. Garruzzo, S., Rosaci, D., Sarné, G.: Isabel: A multi agent e-learning system that supports multiple devices. In IEEE/WIC/ACM International Conference on Intelligent Agent Technology (2007)
7. George Margetis, X. Z., Koutlemanis, P., Antona, M., Stephanidis, C.: Augmented interaction with physical books in an Ambient Intelligence learning environment. Multimedia tools and applications, vol. 67, nº 2, p. 473–495 (2013)
8. Valdiviezo, P., Cordero, J., Aguilar, J., Sánchez, M.: Conceptual Design of a Smart Classroom Base on Multiagent System. In 2015 International Conference on Artificial Intelligence (ICAI'15), Las Vegas, USA (2015)
9. Aguilar, J., Sánchez, M., Valdiviezo, P., Cordero, J.: Mecanismos de Coordinación en un Salón Inteligente. In 6to Congreso Iberoamericano de Estudiante de Ingeniería Eléctrica (VI CIBELEC 2015), Mérida, Venezuela (2015)
10. Sánchez, M., Aguilar, J., Cordero, J., Valdiviezo, P.: Basic features of a Reflective Middleware for Intelligent Learning Environment in the Cloud (IECL). In Asia-Pacific Conference on Computer Aided System Engineering (APCASE'15), Quito, Ecuador (2015)
11. Wolfgang, G., Hendrik, D.: Translating Learning into Numbers: A Generic Framework for Learning Analytics. Educational Technology, vol. 15, nº 3, pp. 42-57 (2012)
12. Aguilar, J., Ríos, A., Hidrobo, F., Cerrada, M.: Sistemas Multiagentes y sus aplicaciones en Automatización Industrial. 2nd ed., Mérida: Talleres Gráficos, Universidad de Los Andes (2013)
13. Aguilar, J., Besembel, I., Cerrada, M., Hidrobo, F., Narciso, F.: Una metodología para el Modelado de Sistemas de Ingeniería Orientado a Agentes. Inteligencia Artificial: revista iberoamericana de inteligencia artificial, vol. 12, nº 38, p. 39–60 (2008)