International Journal of Advanced Information Science and Technology (IJAIST) ISSN: 2319:2682
Vol.4,No.7, July 2015 DOI:10.15693/ijaist/2015.v4i7.36-49

A Smart Learning Environment based on Cloud Learning

MSc. M. Sánchez
AggregateProfessor/Universidad Nacional Experimental del Táchira, San Cristóbal, Venezuela

Dr. J. Aguilar
Titular Professor/Universidad de Los Andes, Mérida, Venezuela
Prometeo researcher/Universidad Técnica Particular de Loja, Ecuador

MSc. J. Cordero
Titular Professor/Universidad Técnica Particular de Loja, Ecuador

MSc.P. Valdiviezo
AggregateProfessor/Universidad Técnica Particular de Loja, Ecuador

***Abstract*—In this paper is presentedan architecture of a Reflective Middlewarebased in Cloud Learning, for Intelligent Learning Environments.The middleware is defined using theMultiagent Systemsparadigm, and propose academic services on the cloud based on the current context.That is, the middlewaremanages educational services in the cloud to enhance the learning experience of students, either collaboratively or individually.In that sense, in this paper is detailed the middleware components, which enable the process of management of the cloud computing. The paper also presents examples of the utilization of the middleware to provide services on the cloud about tasks of learning analytics, which allow processing of data of students and learning environment in order to understand and optimize their learning processes.**

Index terms -Intelligent Learning Environment; Cloud Learning; Reflective middleware; Ambient Intelligence

I. INTRODUCTION

The new advances in information technology, in domains like cloud and ubiquitous computing, allow us exploit all computation tools (devices, software, etc.) as a whole, in order to define systems in a high abstraction level. In this way, it is possible the developing of new domains like Ambient Intelligence (AmI). In particular, the AmI for education has been used as space where the technologies of the clouds and ubiquitous, help the learning processes in an unobtrusive manner, to improve their scalability and integration capabilities[1, 2, 3].

In educational environments, the AmI allows to follow the dynamic needs of teaching and learning of users[4, 5, 6, 7], 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 services in the cloud, using the mechanisms and tools that provides cloud computing. For that, the resources are stored in a virtual environment, which can be accessed in different ways, times and locations.

Combining C-Learning with AmI brings great benefits to the teaching-learning process, as it will not only count on the services provided by C-Learning, but the environment will be able to determine when it is appropriate for the user to make use of these services, and with what devices or smart objects available in the environment must be integrated. Thus, we propose a new concept called AmI for C-Learning (AmICL), defined as:AmICL combines educational services available in the cloud with objects (which can be intelligent or not) found in an educational environment, in order to achieve a learning process adapted to the student's learning style.

Thisarticleaims to presentthe detailed design ofthe implementationof the paradigm ofcloud computinginAmICL,as well asverify their operationina use case. Basically, the use caseis linkedto providingcloud serviceslinked toLearningAnalytics (LA) tasks. LArefers to theprocessing of dataofstudents andlearning environments, in order to understand andimprovetheir learning processes [8].

This articleis organized asfollows:we begin bypresenting thetheoretical framework, then we presentthe detailed specificationof the components ofAmICL responsible for the management of the cloud computing. Finally,we describethe use case, detailing boththe design and utilization of the services of LAin the context of the AmI where they are used.

II. THEORETICAL FRAMEWORK

A. AmI for C-Learning (AmICL)

In the literature there are several definitions about AmI, for example, in[9] an AmI is defined as "one which uses computer technology to control its operation automatically, so as to optimize user comfort, consumption of resources (energy for example), safety and efficiency of work". The main idea of this definition is the automation of an ambience, in order to optimize a set of aspects around of its utilization.

A C-Learning is a learning model that uses all digital resources existing on the Internet, in order to improve the training process of the students. In this type of learning, the set of tools and services in the cloud, that contribute to the student learning process, is incorporated, without necessity for

students and teachers to be physically in the same classroom.

The combination of C-Learning with AmIcan give large benefits to the teaching-learning process, because it will not only count on the services provided by C-Learning, but the environment will be able to determine when it is appropriate to make use of these services, and with what devices or smart objects available in the environment must be integrated. Thus, we propose a new concept called AmI for C-Learning (AmICL), defined as:

An AmICLis an Intelligent Learning Environment that combines educational services available in the cloud with objects (which can be intelligent or not) in the educational ambiance, in order to adapt the learning process to the student's learning style.

B. Autonomic Reflective Middleware

A classic Reflective Middleware is a distributed system that acts as an intermediate layer between applications and services[10], with the possibility of change the behavior of the applications, through self-awareness and self-reference. For that, a Reflective Middleware executes two processes:

- Introspection: The program's ability to observe and reason about their status [11, 12].
- Intersection: The program's ability to change its own execution state [11, 12].

A Reflective Middleware is defined by two or more levels: the **Base level** is where the applications run, and the **Meta level** implements the reflectivity and verifies that the system operation (application) is the expected or required. At this level, introspection processes are executed, while intersection requires of the two levels [10, 11, 12].

An Autonomic Reflective Middleware incorporates sensors and actuators for observing the environment and acting accordingly [12]. The Autonomic Reflective Middleware fuses in the Reflective Middleware architecture, the different levels that composed the autonomic computing paradigm:

Resources Management: It can be any type of resources (hardware or software) that can be managed, and may have embedded self-managing attributes.

Touch Point: Link to the sensors and/or actuators required for resources management.

Autonomic Manager: Implements intelligent control ties that automate the tasks of self-regulation/self-management.

Orchestrating Autonomic Managers: Because an autonomic system can have several autonomic managers that need to work together to ensure the proper functioning of resources, this level provides the communication channel for the coordination between them.

Manual Handler: Allows humans to configure the autonomic managers to perform its task of self-management, providing for this a man-machine interface.

Knowledge Resource: Provide access to the knowledge required toperform autonomic management of the system.

The main element of the Autonomic Reflective Middleware is the Autonomic Manager, because it performsan intelligent control loop known as MAPE[12].The MAPE control loop consists of four phases:

Monitor: Gathers through sensors events/details of what is happening in the managed resources, making them a set of patterns or symptoms that can be analyzed by the analyzers component.

Analyzer: Provides mechanisms to analyze patterns/symptoms provided by the monitor, to determine the reasons of the problems in the managed resources.

Planner: Generates a set of operations to perform on the managed resources, in order to achieve the goals and objectives proposed.

Executer: Interact with the actuators to perform the necessary set of actions contained in the Plan. Provides an interface for communication between the autonomic manager and the managed resources.

III. AMICL SPECIFICATION

A. Conceptualization of AmICL

The main difference of AmICLwith similar researchesis that it combines the AmI with academic services provides by 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 optimize the teaching and learning process of users. This enables to make available to users intelligently services deemed appropriate to perform some activity at a specific time.

AmICL is an Autonomic Reflective Middleware that enables the integration of objects (intelligent or not) in the educational ambiance, with educational services 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, using the educational services(see Fig. 1, the multi-layer architecture for AmICL).

In particular, the new aspect introduced by AmICL in order to fuse an AmI with the cloud computing paradigm is the Services Management Layer (SML). This layer has some agents which are an extension of the FIPA standard proposed by [1], because the layer not only handles the resources and applications, but also manages the educational services (locals and in the cloud), to take advantage of learning services available, in order to adapt the AmI to current learning conditions required by its users.

B. AmICL multilayer architecture.

1) Physical Layer (PL):contains the physical and virtual components present in the AmI(how this is an educational environment, in principle, they would be objects of a classroom), such as: a) Sensors, b) Intelligent Objects (for examples, a StudentBoard, where students and teachers connect virtually to the environment); c) Other Objects that lack intelligence and cannot communicate with other elements in the environment d) Software Objects, like the educational applications for managing educational contents, virtual learning environments, among others (for examples, Moodle,

EVA, Joomla, etc.).

2) MAS Management Layer (MMAL): This layer is adapted from FIPA standard proposed by[13], which defines the rules and agents that allow a society of agents to exist, operate and be managed. AMA manages the agents on the MAS, CCA the communication among them, and DMA the data on the MAS.

3) Services Management Layer (SML): AsMMAL, this layer is an adaptation of FIPA standard proposed by[13], by noting that in this case it is extended because the layer not only handles the resources and applications,but also managesthe educational services (locals and in the cloud), to take advantage of learning resources available, to adapt the AmI to current learning conditions required by its users.

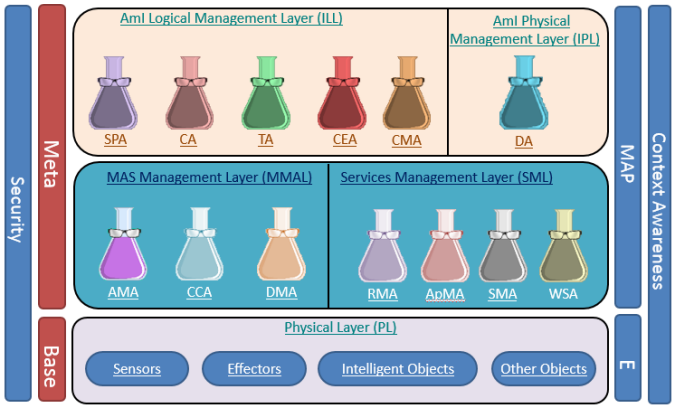

Figure 1. AmICL Proposed architecture.

The MMAL and SML layers provide operational services to ILL and IPL layers, which will allow agents of these two layers to meet the objectives for which they were designed. Some of theirservices are based on [13], but others are derived from the extension ofSML to support the cloud learning paradigm (particularly, the SMA and WSA agents).

4) AmILogical Management Layer (ILL): It is the responsible for providing intelligence to theAmICL. This layer is where all the applications (Software Objects) and persons present in the AmI are characterized. Basically, each application/person is characterized by an agent that contains metadata that defines its properties. In this layer there are different types of agents (CMA, SPA, TA, CA and CEA) as element types that can appear in the AmI, they are later explained.Some of these applications are smart, so that the properties of learning, reasoning, among other, are defined for them.

5) AmIPhysical Management Layer (IPL): This layer has characterized the physical elements of the environment, to facilitate the use and communication with each of them. Thus, each physical device is characterized by an agent that contains metadata that defines its properties. Some of these physical devices are smart (Intelligent Objects), so that the properties of learning, autonomy, reasoning, among other, are crucial to characterize them. This layer communicates with the distributed operating system, as it is through it that has access to physical hardware devices.

C. Characteristics of the Cloud Management Agents.

The MMAL has three agents, whose general characteristics are detailed in [13]. At theSML coexist four agents, two classics of FIPA, such as RMA and ApMA(they manage the resources and applications on the MAS[13]), and two new for theAmICL (they are the bridge between AmIand the cloud):

1) Services Management Agent (SMA): Controls, records and manages web services available on the system (whether they are in the cloud or locally); so that when an agent requires a specific serviceitshould contact this agent to locate the Web Service Agent(WSA) that characterizes that service in the MAS. Additionally, it must discover new leaning services on the cloud. The SMAwould characterize what in web services is known as UDDI (Universal Description, Discovery and Integration), which is a standard on web services.

2) Web Service Agent (WSA):it isthe logical representation of a web service. This agent characterizes the web services, for that reason, it is who knows how to invoke (it has its interface) and which are the requirements necessaryto access the web service. The WSA has access to the services description file (WSDL). There is one instance of this agent for each web service on the SMA.

Fig. 2 shows how agents in AmICL can use SaaS (Software as a Service) model of the Cloud. First, web services will be registered on SMA, this action allows to know which web services are available to be used by the AmICL's agents.

Second, when an agent requires a web service, it sends a query to SMA, SMA starts a bidding, to select the WSA that can meet the requirements of the agent. Once the WSA is selected, the agent can send a message to it to consume the web service; WSA will use the web service proxy created using the WSDL file to invoke the web service endpoint. WSA will receive the web service result and will send it back to the agent.

This model enables the system scalability because each agent that requires invoke a web service does not need to know its WSDL file neither create a proxy to invoke the same. Also, this model allows use FIPA protocols (bidding) to determine the web service that meets the requirements of a request in the best manner.

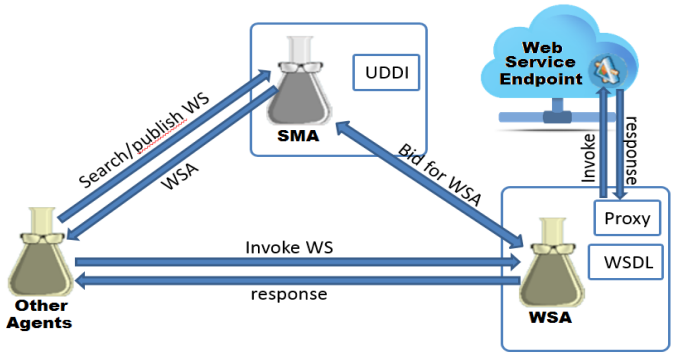

Figure 2. Agents and Cloud Services Integration.

D. Characteristics of the highest Layers

The agents of the AmILogical Management Layer are:

1) Student's Profile Agent (SPA): This agent is an abstract representation of students, it models both student activity in the environment and their personal data, such as academic profile and usage history of objects in the environment. There is an agent of this type for each student in the environment.

2) Collaborative Agent (CA): Its function is essential to promote collaborative learning, because this agent is responsible for locating students studying the same subject or performing similar activities, to put them in communication through applications of collaborative environments.It must execute tasks like Social Network Analysis (SNA), web mining, etc. as novel methods for creating academic similarity networks.

3) Tutor Agent (TA): This agent is a representation (abstraction) of the Professor, or of a manager learning processes (Intelligent Tutorial System[14], Virtual Learning Environment[15]) which guides the learning processes. There is one instance of this agent per each tutor in the environment.

4) Content ManagementSystem Agent (CMA): Model the characteristics of different types of content management systems (CMS[16, 17]), learning processes (LMS, CLMS)[16, 17], etc. in an educational environment. There is an instance of this agent for each one in theAmICL.

5) Collaborative Environments Agent (CEA): represents applications of collaborative work and collective learning, so that they can be accessed by the Collaborative Agent when is required. For each application of a collaborative environment that is required in the AmICL, there is an instance of this agent.

The AmIPhysical Management Layer is composed of a single type of agent, and its detailed description is as follows:

1) Device Agent (DA): Represents a logical abstraction of a device in the real environment, which allows modeling the interaction with these objects. There is an instance of this agent for each device in the environment.

E. AmICL Architecture based on Autonomic ReflectiveMiddleware.

The levels that make theAmI reflective middleware are:i) Base level: At this levelare the applications with thesystem functionalities and the physical devices in the AmI (the elements of the Physical Layer of AmICL). ii)Meta Level: is where the middleware has its ability to reflect, generally, it observes, reasons and adapts its internal structure and behavior according to the necessities of the environment. In AmICL,this level is composed by theILL, IPL, MMAL and SML layers.

Regarding the autonomic capacity, according to the MAPE model, it is specified in AmICL as:

1) Monitor: This occurs in the ILL and IPL layers. This task is carried out specifically by DA, TA and SPA agents, which are in constant communication with the sensors and smart objects in the environment, to know the current state of the environment and the activities that users are developing in it. It answers questions like: how?, and when? To establish communication among the smart objects, the ontological component plays a critical role (explained below).

2) Analyzer: This is carried out on the ILL and IPL layers, for the agents TA, SPA, CA, CMA and DA agents (when they are intelligent objects). This component analyzes the data arriving through these agents to determine, among other things, changes in the learning process, inclusion of new items, incorporation of new learning goals, etc. Tasks such as pattern recognition, trend analysis, inference,etc., are critical.

3) Planner: In this step, the agents evaluate actions to be taken in respond to the situation detected by the analyze phase. Specifically, the tasks/services to be performed, or applications to be run, are determined,with the sequence between them, and who must performed it (agents, educational services in the cloud, etc.). The TA, CMA,SPA and CA agents are the planners of the solution.

4) Executer:the plans done in the previous phase aresent to the various abstractions of agents present inthe AmI (IPL and ILL layers),or are invoked some WSAs for the specific services involved in the plans. The abstractions invoke the components of the physical layer for theexecution (for example, intelligent object or software).

Regarding the ontological component, AmICL requires 3 ontologies: *Component ontology* defines the users and entities that make up the environment. In addition, each user or entity should be described semantically in a*domain ontology*. Finally, it is important to maintain the contextual information in a *context ontology* that allows characterizing the current contextual properties for the entities of the environment(see[10]).

IV. SPECIFICATION OF THE CLOUD MANAGEMENT SUSBSYSTEM OF AMICL.

In this paper, we are going to define the agents that compose the *Services Management Layer (SML).* For that, we define a general type of service agent (framework) to represent the different services on the cloud,which can be invoked by AmICL. Additionally, we need to 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. In this way, in this layer are specified two agents (they are the bridge between the AmI and the cloud):

Services Management Agent (SMA): controls, records, discovers and manages web services available on the system (whether they are in the cloud or locally); so that when an agent requires a specific service, it should contact this agent to locate the Web Service Agent (WSA) that characterizes that service in the MAS. The SMA is a hybrid that characterizes what in web services is known as UDDI (Universal Description, Discovery and Integration), which is a platform-independent framework to describe, discover, and integrate web services; with an enterprise service bus (ESB) (software architecture model used for designing and implementing communication between applications based in the SOA

paradigm). The main differences between SMA and UDDI, is that the first has the capacity of localize/discover a web service, by category, which is essential to ensure integration with other system agents, and accomplishes tasks of an ESB.

Web Service Agent (WSA): it is the logical representation of a web service. This agent characterizes the web services, for that reason, it is who knows how to invoke (it has its interface) and which are the requirements necessary to access the web service. The WSA has access to the services description file (WSDL), and by using it, WSA can create a local proxy to consume the service's methods. There is one instance of this agent for each web service on the SMA.

In this section,we will use the MASINAmodels [18] to specify these agents.Basically, the models that interest usare model ofagents (for detail eachagent, its objectives, its services provided, etc.), of tasks (define the activities to carry outto achieve theirobjectivesor to provideservices)and of intelligencefor caseswhere there is suchbehavior in an agent.

A. Phase 1: Conceptualization.

1) Services Management Agent: In Table I is described the "Services Management" use case, and Fig. 3 presents its activity diagram.

TABLE I. SERVICE MANAGEMENT USE CASE

Use Case: **Service Management**
Description: This use case allow to manage (registering, searching and removing) web services to and from the SMA's web services directory, such that those web services can be accessed by other system agents. Pre-condition: request from a system agent Actors: system agents, SMA Failure Condition: failure in the service management. Success Condition: service managed without errors.

The activities diagram of Fig. 3**Error! Reference source not found.** shows in details the services provided by the SMA. This agent offers three services; the first one adds a web service into the SMA's web services directory; the second one locates/discovers a web service, and finally, the third one removes a web service from the SMA's web services directory.

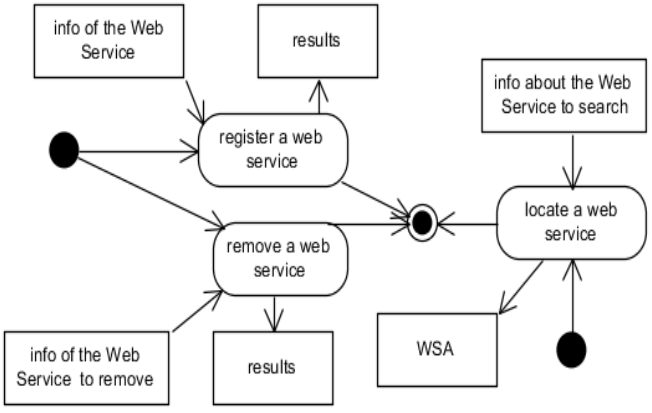

Figure 3. Activities Diagram for the "Service Management" use case.

2) Web Service Agent: Table II shows the description of the "Invoke Web Service" use case. In general, when a system agent requires invoking a Web Service method, it does not need to know the messages formats that will be sent to the Web Service to consume it, the requesting agent only need to know the data that the Web Service require to give response to its request; this information can be demanded to WSA. The activities diagram for this use case is shown in Fig 4.

TABLE II. INVOKE WEB SERVICE USE CASE

Use Case: **Invoke Web Service**
Description: This use case permits that system agents can consume the web service in a transparently manner. Pre-condition: request from a system agent. Actors: System agents Failure Condition: communication failure. Success Condition: Web Service results.

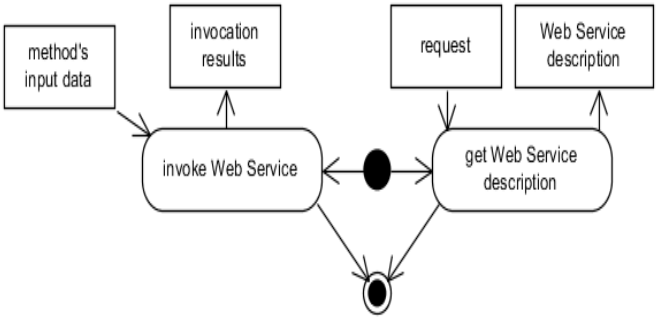

Figure 4. Activities Diagram for the "Invoke Web Service" Use Case.

WSA offers two services: the first one invokes a Web Service method by providing the input data, while the second one permits to obtain the Web Service description, which can be used to know what type of information is needed to invoke the service.

B. Phase 2: Analysis.

1) Agent Model for SMA: The Agent Model for SMA is shown in TablesIII, IV, V, yVI. Table III describes SMA: its role, type, etc. Particularly, SMA is a goal-based intelligent agent which needs to learn how to localize web services based on past experiences.

TABLE III. SMA DESCRIPTION

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

The objective of the SMA is to facilitate: access, location (service discovery) and consumption of the Web Services in the Cloud, such that other agents can make use of these services in a transparent manner, In Table IV, the objective of

the SMA is described.

TABLE IV. SMA'S OBJETIVES

SMA's Objetives
Facilitate the access, location and consumption of the Web Services in the Cloud. Type: On-condition objetive Input parameters: Web Service Information Output parameters: Web Service Requested Activation Condition: Web Service information received from system agents. End Condition: Web Service located or error. Success Condition: Web Service located. Failure Condition: Web Services not found. Representation language: natural language. Description: The main objective of SMA is to locate the web service that other agents require.

As mentioned above, the SMA offers 3 services:

- Register a Web Service; allowregistering a Web service in the SMA's service directory. In order to complete the tasks of this service, it is necessary: the description file of the Web Service (WSDL), and a file describing the operation types of its methods (create, request, update, delete, etc.).
- Locate a Web Service; this service allow to locate a Web Service that meets the requirements of category, input and output required by the requesting agent.
- Remove a Web Service; allowunregistering a web service.

Table Vshows the description of the SMA Service "Locate a Web Service". The other services of SMA are described in [19].

TABLE V. SMA SERVICE

SMA Service
Name: Locate a Web Service Type: Dual Input parameters: Web Service Category, input parameters, output parameters. Output parameters: WSA which characterizes the Web Service required or error when the Web Service is not found. Representation language: Natural Language

The SMA will have the ability to localize/discoverWeb services which are being required by other system agents (see TableVI), based on past experiences. The SMA should combine its ability with the abilities of WSA to perform the location of the Web Service that accomplishes with the requirements ordered by the agents.

TABLE VI. SMA CAPABILITY

SMA Capability
Ability: Localization of Web Services, based on past experiences. Knowledge representation: OWL Communication language: ACL

2) SMA Tasks Model: The relationship between the SMA services and the tasks executed by this agent for each of their services is shown in TableVII.

TABLE VII. SMA TASKS AND SERVICES

SMA-S1. Register a Web Service
T1. Receive the Information of the Web Service (WSDL, service name, web service methods description, category, etc.). T2. Check the connection to the Web Service. T3. Register the Web Service in the directory.
SMA-S2. Locate a Web Service
T1. Receive data of the Web Service to localize. T2. Locate/Discover the WSA requested.
SMA-S3. Remove a Web Service
T1. Receive data of the Web Service to remove. T2. Remove the WSA that characterizes that Web Service.

The first task to be executed by the "Register Web Service" service (SMA-S1) is to receive the data of the web services to be registered (see Table VIII).

TABLE VIII. RECEIVING THE INFORMATION OF THE WEB SERVICE

Task: Receive 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.

Then, the SMA-S1 will perform a task that will check whether the Web Service that is being registered is on line 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 Web Service (The task model of this process is in [10]). The next task that SMA-S1 performs is to register the Web Service in the knowledge base of the SMA, so that it will be found in a new search (The task model of this process is in [10]).

The tasks carried out to meet the service "Locate Web Service" (SMA-S2) of SMA are described next. First, there is the task "Receive data of the Web Service to localize", which receives the information that is needed to localize the Web Service that has the ability to meet the requirements requested (The task model of this process is in [19]).

Once the data are received, the next task (see TableIX) determines through a bidding process, which WSA has the ability to meet the requirements requested. This task is performed by a bid process, because each WSA knows its category and its function, so only those WSA who can meet the requirements will make a proposal.

TABLE IX. LOCATING THE WSA REQUESTED

Task: Locate the WSA requested
Objective: Determine which WSA has the ability to meet the requirements requested. Pre-condition: Information of the Web Service received with not errors. Frequency: Each time that a Web Service is required. Description: This task should localize the Web Service that has the ability to meet the requirements requested by other system agents. For this task a bid process to select the WSA will be started.

The third service offered by SMA is responsible for "Remove a Web Service" (SMA-S3). This service performs two tasks. The first one will receive data of the Web Service to remove, and the second one will update SMA's knowledge base (The task models of this service are in [19]).

3) SMA's Intelligent Model: Tables X, XI and XII describe the learning and reasoning mechanisms and the storage of the experience (knowledge) for the SMA.

TABLE X. SMA'S LEARNING MECHANISM

SMA's Learning Mechanism
Name: Web Service discovering Type: Adaptive Representation techniques: Rules Learning Sources: Success or failure during WSA localization. Update mechanism: Experience Feedback

TABLE XI. SMA'S REASONING MECHANISM

SMA's Reasoning Mechanism
Information source: system agents Activation source: SMA-S2 Tasks Inference technique: based on rules Knowledge representation language: OWL Task-inference relationship: It decides whether or not localization rules are adequate for localize a Web Service Agent. Reasoning strategies: Deductive

TABLE XII. SMA'S EXPERIENCE

SMA's Experience
Description: localize Web Services based on past experience. Characterization: Web Services Source: Rules Default values: 0 Max and min values: 100 – 0

4) Agent Model for WSA: The Agent model for WSA is divided into TablesXIII, XIV, XV y XVI. Table XVIII shows the description of this agent. The agent is of type "reflex simple", because its fundamental role is to invoke the Web Service that characterizes.

TABLE XIII. WSA DESCRIPTION

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

The goal of the WSA (Table XIV) is to serve as a local proxy to access the remote Web Service (endpoint), so that an agent can consume the Web Service.

TABLE XIV. WSA'S OBJETIVES

WSA's Objetives
Serves as a local proxy to access the remote Web Services (endpoint) Input parameters: method type, input values to send to the Web Service Output parameters: Web Service results Activation condition: activated when a system agent requires invoke a Web Service. End condition: Web Service invocation finished. Success condition: Web Service finished with not errors. Failure condition: failure in Web Service invocation. Representation language: Natural Description: The WSA is responsible for invoking a method in the web service, according to the input parameters and the type of invocation.

The WSA offers two services. The first one, "Invoke Web Service", allows to consuming the web service in question given the input parameters to the service (Table XV). The "get Web Service Description" service returns information,which is necessary to know about the input and output parameters of the Web Service (The model of this service is in [19]).

TABLE XV. WSA SERVICE

WSA Service
Name: Invoke Web Service Type: Dual Input parameters: input values to the Web Service method. Output parameters: Web Service invocation results Representation Language: Natural language

WSA has the ability to recognize requirements and data under which the Web Service can be used. Thus, in cooperation with SMA, it can facilitate the process of locating Web services (see Table XVI).

5) Task model for WSA: Table XVIIshows the relationship between services and tasks that are carried out to fulfill the WSA's services. WSA-S1 service performs 4 tasks. The first one is responsible for receiving the input data to the Web service and verifies that the data have the appropriate format (The task model of this process is in [19]). The second task (see Table XVIII) infers the Web service method that will be called based on the input parameters and the type of operation to be performed (creating, requesting, updating or deleting), and so for.

TABLE XVI. WSA'S CAPABILITY

WSA's Capability
Ability: To know requirements under which the Web Service that it characterizes is used. Knowledge Representation: OWL Communication Language: ACL

TABLE XVII. WSA TASK AND SERVICES

WSA-S1. Invoke Web Service
T1. Receive input data to invoke the Web Service. T2. Infer method's name that will be invoked. T3. Invoke the Web Service 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.

TABLE XVIII. INFERRING METHOD'S NAME THAT WILL BE INVOKED

Task: Infer method's name that will be invoked
Objective: Infers the Web Service method to invoke. Pre-condition: input values received correctly. Frequency: For each Web Service invocation. Description: The task is responsible for inferring the Web Service method that should be invoked based on the input parameters and the type of operation that should be performed: create, request, update or

delete.

In Table XIX, the task to invoke the web service method for the WSA-S1 service is described. The result should correspond to the result expected by the requesting agent.

TABLE XIX. INVOKING THE WEB SERVICE METHOD

Task:Invoke the Web Service Method
Objective: Invoke the web service method. Pre-condition: T2 for WSA-S1 completed without errors. Frequency: When the Web Service is going to be consumed. Description: To Invoke the Web service method to obtaining the result expected for the requesting agent.

Finally, the WSA-S1 service performs the task "Send execution result to the requesting agent" to send the results of the invocation of the Web Service requested by the agent (The task model of this process is in [19]).

The WSA-S2 service must perform two tasks: The first task (see Table XX) analyzes the WSDL file to determine the input and output parameters of the Web Service method.The second task WSA-S1 sends the data extracted by the first task to the agent (The task model of this process is in [19]).

TABLE XX. ANALYZE WSDL FILE

Task: Analyze WSDL file
Objective: Analyze the WSDL file to obtain information about the Web Services methods. Pre-condition: N/A Frequency: When description of the Web Service is required. Description: This task is responsible for analyzing the WSDL file of the Web Service to obtain information about the input and output parameters of the Web Service methods.

C. Registration of a Service on AmICL.

In general, to use a service in AmCL, it must be registered in the SMA. To register a Web Service in the SMA, so that it can be discovered and located, the following information is required:

- **Service Name:**specifies the name of the new Web Service.
- **Business Name:**specifies the business entity that will offers the new Web Service.
- **WSDL Location:**specifies a URI that points to a WSDL document that contains a description of the service.
- **Web Service Description:**specifies a XML document to describe the operation type of each Web Service Method.
- **Category:**specifies the category into which the service is located. The category helps to identify a Web Service into the SMA's directory: Academic (Recommender System (Affective, Learning Style), Quiz Generator, Clustering, descriptive learning model, etc.), Recognition (Speech, Face, Gestures, etc.). Thus, if a specific agent needs to locate a recommender system it would have to tell to the SMA its category, which, in this case would be: Academic/Recommender System/Learning Style.

An example of the possible structure of the XML document would be:

```
<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 elements in the XML file are used to know which service WSA should call according to what want to perform. This information does not come within the WSDL file, it is an extension in the SMA in order to take decisions when a WSA is being locating.

V. EXPERIMENT

A. Case Study

We analyze the case study of an online tutoring process in AmICL. AmICL adapts the online tutoring process to the 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 which 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 he develops certain activities proposed by the teacher. The computer is used to monitor the student activity. Similarly, the environment has a VLE (MOODLE) application and web content recommendation services that use AmICL to help students to develop their academic activities.

This conversation is described in Fig. 5**Error! Reference source not found.**, it is here where the learning environment begins, identifying users that enter to the VLE with a context aware service which will identify the current student carrying out a recognition (face). In this case is called the SMA for the invocation of this service. The conversation is activated when the user try to input to VLE (message 1). At that time, VLE identifies the user that is entering to the system, for which it uses a recognition service of face located through SMA (messages 1.1 to 1.3). Once VLE has the UID of the different users of this season, it instantiates the corresponding agents (message 1.5 to create a TA, message 1.8 to create a SPA) for each user. AmICL determines that it must prepare the smart board available in the intelligent classroom (message 1.6). Thus, TA locates the course data and plans the class (message 2); with that information, it locates today class' slides

(messages 3 and 3.1) through a storage service (in the course plan, it is indicates where the slides are stored, and in this way it can call the RMA).

The slides file (managed through the RMA) is sent to the smart board (messages 5), then the DA (Smart Board) is prepared to begin class when teacher ordered. If the user is a student, the academic data is retrieved (message 4) to determine the learning style and usage history of the environment for the group of students in the AmI. For that, VLE invokes a clustering service (messages 6 to 7.3) 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 (messages 8 to 8.2). In this case, each group is a pattern or style of learning to be exploited by the recommender system, to search more accurate information. That is, 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 (messages 9 to 10.2 and 12 to 12.1). Then, the students interact with these learning resources via the smart board (message 13), and TA monitors (message 14) the work of the students (normally, the learning resources have an evaluation phase before to finish with their utilization, that VLE must guarantee the students carry out). 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) for the next session (messages 15 to 15.1.3). 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 respecting agents of SML (messages 16 to 17.3).

In the conversation are involved different types of agents: one device type (the smart board), and the rest of software type: VLE, students, etc. Additionally, are invoked several services on the cloud for specific requirements during the session, by the agents of the AmICL.These services invoked solve specific LA tasks, as the definition of the patterns of style of learning of the students (clustering task) or the online tutoring model to learn about the activities, tools, etc., used normally. With this information, AmICL can optimize it.

B. LA Services: 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 involves gathering 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, video collaboration, among others.

We have tested our approach in an Ecuadorian university where one of the main activities is the distant education. Particularly, the Universidad Técnica Particular de Loja (UTPL) has a competency-based teaching model. In this model, the student is the central actor in the educational process, and the process is mediated by a teaching team, tutorials, resources learning and new technologies. For online tutoring, UTPL uses: a VLE based on MOODLE platform, a Video Conference System, e-mail, phone tutoring, etc. The studentsthrough the forums, chats and video conferencing in the VLE, can get points. Therefore, the teacher must meet certain activities that enable the communication and evaluation of teaching-learning process, such as:i) propose at least a forum, a chat and videoconferencing every two months, ii)answer the question of the students through the email, iii) upload educational resources.

The information about the interaction of the users of VLE is collected in a file. Weuse this file to determine a descriptive model that characterizes the use of these tools on the platform during the online tutoring process.To obtain the descriptive model, the data was first preprocessed, and then several experiments were performed with association rules, in order to obtain the model with the best results. The R tool was used (its “arules” library) [20].

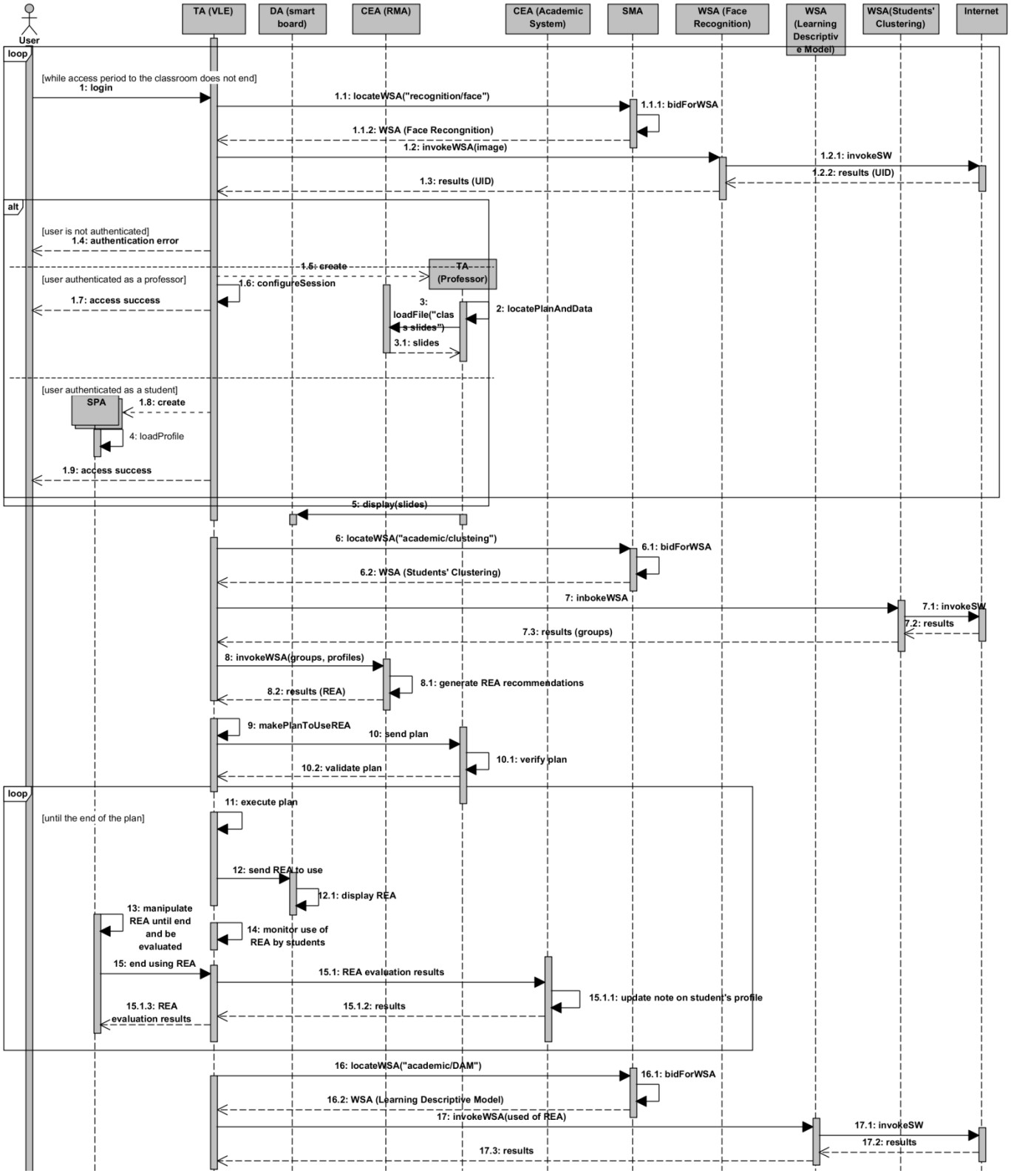

Figure 5. Conversation "online tutoring process"

The results are:

- 8 attributes are used: Ads (NumCommentsProfessorStudent), Learning Resources (NumRessources load), Forums NumForumsDefined), Tasks (NumTaksDefined), REAS (NumREAS), Chats (NumChatDefined), NumMssEnv (NumMessRep), NumParticipantesChat (NumParticChat).
- The best results were obtained for the experiment of the Fig. 6 (with 24 rules).

According to the results of the Fig. 6, the first 6 rules are considered very obvious, or of little interest, which is reflected in the measures of support and interest. In the case of rules 4, 5 and 6, they represent an interest of 68% approximately. In particular, the rules 12, 20 and 21 are the more significant. They describe very well the relationship between the attributes involved.

```
## 1  {}          => {Recursos}   0.950      0.950 :
## 2  {}          => {Foros}      0.975      0.975 :
## 3  {}          => {Anuncios}   0.983      0.983 :
## 4  {Chats}     => {Recursos}   0.681      0.988 :
## 5  {Chats}     => {Foros}      0.681      0.988 :
## 6  {Chats}     => {Anuncios}   0.681      0.988 :
## 7  {Recursos}  => {Foros}      0.933      0.982 :
## 8  {Foros}     => {Recursos}   0.933      0.957 :
## 9  {Recursos}  => {Anuncios}   0.933      0.982 (
## 10 {Anuncios}  => {Recursos}   0.933      0.949 (
## 11 {Foros}     => {Anuncios}   0.958      0.983 :
## 12 {Anuncios}  => {Foros}      0.958      0.974 :
## 13 {Recursos,
##     Chats}     => {Foros}      0.672      0.988 :
## 14 {Foros,
##     Chats}     => {Recursos}   0.672      0.988 :
## 15 {Recursos,
##     Chats}     => {Anuncios}   0.672      0.988 :
## 16 {Anuncios,
##     Chats}     => {Recursos}   0.672      0.988 :
## 17 {Foros,
##     Chats}     => {Anuncios}   0.672      0.988 :
## 18 {Anuncios,
##     Chats}     => {Foros}      0.672      0.988 :
## 19 {Recursos,
##     Foros}     => {Anuncios}   0.916      0.982 (
## 20 {Anuncios,
##     Recursos}  => {Foros}      0.916      0.982 :
## 21 {Anuncios,
##     Foros}     => {Recursos}   0.916      0.956 :
## 22 {Recursos,
##     Foros,
##     Chats}     => {Anuncios}   0.664      0.988 :
## 23 {Anuncios,
##     Recursos,
##     Chats}     => {Foros}      0.664      0.988 :
## 24 {Anuncios,
##     Foros,
##     Chats}     => {Recursos}   0.664      0.988 :
```

Figure 6. Best rules obtained.

Specification asWSA: The registration of this service on AmICL is shown in Table XXI:

TABLE XXI. 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>
```

Analysis of results returned by the service: We observe a significant level of support and confidence in rules 12, 20 and 21; that highlights the dependence between four variables mainly: Ads, Forums, Chat and Learning REAs we can see the percentage of correctly classified instances is of 91.7736% and incorrectly classified is 8.2264%. This can be corroborated with the Fig. 7that determines the attributes with more relationship.

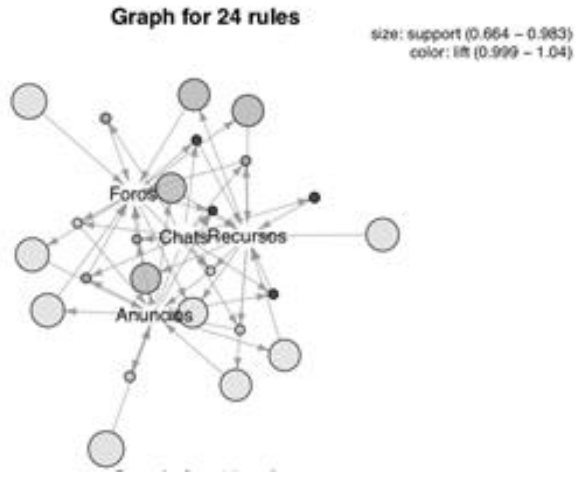

Figure 7. Relationship between the attributes.

Service of students clustering: We used the Kmedias clustering algorithm, because it is one the most commonly used to identify groups of objects with similar characteristics. Additionally, we use the same data from UTPL. For training, the number of cluster is defined as 3 by the amount of homogeneous groups in the data. We divide the dataset with 80% for training and 20% for the test. We used the tool WEKA and the algorithm SimpleKMeans (see Fig. 8).

As is noted in Fig. 8, the three groups have 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.

Specification as WSA: In general the registration of this service (see Table XXII) on AmICL is:

TABLE XXII. 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*

Attribute	Full Data (95)	Cluster# 0 (24)	1 (21)	2 (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

Figure 8. Results of the clustering problem.

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>
```

Analysis of results returned by the service: Group 0: one can see that in this group are students with greater participation in forums, queries sent to the teacher, but have a median level for share resources and sent messages. On the other tools, presents a similar behavior to the other groups.

Group 1: a group having greater contribution to the forum (NumForosEnvEst), a large number of participants in the chat, and more resources available to the student in the course. It's a more participatory group than the other groups.

Group 2: this is a group having a lot of performance on tasks and has less participation in chat.

The number of instances grouped into each cluster is shown in Fig. 9.

```
Clustered Instances

0       8 ( 33%)
1       5 ( 21%)
2      11 ( 46%)
```

Figure 9. Instances into each cluster

VI. CONCLUSIONS

In this work is proposed a Reflective Middleware for Intelligent Environmentin Cloud Learning, called AmICL. This Middleware allowsa better use of resources and academic services in a learning environment, whether physical, virtual or hybrid, and moreover allows adapting the environment to the academic necessities of users.

The functionality of the cloud component of AmICL guarantee a Cloud Learning Environment. AmICL mixes the benefits and capabilities such as reflective, autonomous and context aware of AmI, with capabilities that provide a learning environment in the cloud, which provides academic services that can be accessed from anywhere. This results in a better use of resources and academic services in a learning environment, whether physical, virtual or hybrid, and moreover allows adapting the environment to the academic needs of users.

In particular, our system allows the invocation of educational web services, and therefore be executed by smart objects in the AmI, in transparent manner to users, to improve and adapt their operations to the requirements of the users. In addition, the reflective and contextual awareness capabilities of AmICL allow the system to adapt itself to the necessities of the environment, acting intelligently to every situation that arises. More precisely, these capabilities allow AmICL meeting the academic necessities of users by making efficient use of objects in the environment, improving the teaching-learning process extending these objects with cloud services.

Thus, AmICL exploits intelligent capabilities related to reflection and analysis of the context of middleware, using the LA paradigm, to act appropriately to situations and requirements that occur in the AmI. That is, we have tested LA services on the cloud, for complex task like processing of Big Data, patterns recognition, etc., 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. Next work must analyze the emergence and the self-organization of AmICL, and the specific problems of implementation (multi-agent platform, communication protocols, etc.).

VII. ACKNOWLEDGEMENTS

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] M. El-Bishouty, H. Ogata, S. Rahman y Y. Yano, «Social Knowledge Awareness Map for Computer Supported Ubiquitous Learning Environment,» *Educational Technology & Society,* vol. 13, nº 4, p. 27–37, 2010.

[2] S. Garruzzo, D. Rosaci y G. Sarné, «Isabel: A multi agent e-learning system that supports multiple devices,» de *Proceedings of the IEEE/WIC/ACM International Conference on Intelligent Agent Technology*, November 2007.

[3] G. Margetis, X. Zabulis, P. Koutlemanis, M. Antona y C. Stephanidis, «Augmented interaction with physical books in an Ambient Intelligence learning environment,» *Multimedia tools and applications*, vol. 67, nº 2, pp. 473-495, 2013.

[4] F. Mhiri y S. Ratté, «AARTIC: development of an intelligent environment for human learning,» *SIGCSE Bulletin*, vol. 41, nº 3, pp. 359-359, 2009.

[5] P. Mikulecký, «Smart environments for smart learning,» de *Proceedings of the 9th International Scientific Conference on Distance Learning in Applied Informatics DIVAI*, Sturovo, Slovakia, 2012.

[6] Y. Shi, W. Qin, Y. Suo y X. Xiao, «Smart classroom: Bringing pervasive computing into distance learning,» de *Handbook of Ambient Intelligence and Smart Environments*, Springer, 2010, p. 881–910.

[7] S. Stenvall-Virtanen y K. Nordell, Smart Environments: Technology, Protocols and Applications, Finland, Tech. Rep.: University of Turku, 2014.

[8] G. Wolfgang y D. Hendrik, «Translating Learning into Numbers:A Generic Framework for Learning Analytics,» *Educational Technology*, vol. 15, nº 3, pp. 42-57, 2012.

[9] H. Hagras, V. Callaghan y M. Colley, «A Hierarchical Fuzzy Genetic Multi-Agent Architecture for Intelligent Buildings Sensing and Control,» de *Developments in Soft Computing*, 2001.

[10] M. Mendonça, J. Aguilar y N. Perozo, «Middleware Reflexivo Semántico para Ambientes Inteligentes,» de *CoNCISa 2014*, Caracas, 2014.

[11] A. Zoghbi y B. Teolinda, «Arquitectura Inteligente de Desarrollo de Software,» Mérida, Venezuela, 2010.

[12] J. Vizcarrondo, J. Aguilar, E. Exposito y A. Subias, «ARMISCOM: Autonomic Reflective MIddleware for management Service COMposition,» de *GIIS 2012*, Choroni, Venezuela, 2012.

[13] J. Aguilar, A. Rios, F. Hidrobo y M. Cerrada, Sistemas MultiAgentes y sus aplicaciones en Automatización Industrial, Mérida: Universidad de Los Andes, 2013.

[14] D. A. Ovalle y J. A. Jiménez, «Ambiente Inteligente Distribuido de Aprendizaje: Integración de ITS y CSCL por medio de Agentes Pedagógicos,» *Revista Escuela de Ingeniería de Antioquía*, pp. 89-104, 2006.

[15] W. F. McComas, «Virtual Learning Environment,» de *The Language of Science Education*, Springer, 2014, pp. 110-110.

[16] A. Cañellas Mayor, «CMS, LMS y LCMS. Definición y diferencias,» Centro de Comunicación y Pedagogía, [En línea]. Available: http://www.centrocp.com/cms-lms-y-lcms-definicion-y-diferencias/. [Último acceso: 17 02 2015].

[17] N. M. García, «Diferencias entre CMS, LMS, LCMS y EVA (elearning),» E-Ducación, 27 11 2013. [En línea]. Available: http://e-ducacion.info/e-learning/diferencias-entre-cms-lms-lcms-y-eva-elearning/. [Último acceso: 17 02 2015].

[18] J. Aguilar, I. Besembel, M. Cerrada, F. Hidrobo y F. Narciso, «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.

[19] M. Sánchez, J. Aguilar, J. Cordero y P. Valdiviezo, «Basic features of a Reflective Middleware for Intelligent Learning Environment in the Cloud (IECL),» de *Asia-Pacific Conference on Computer Aided System Engineering*, Quito, Ecuador, 2015.

[20] R. Jafri y H. R. Arabnia, «A Survey of Face Recognition Techniques,» *Journal of Information Processing Systems*, vol. 5, nº 2, pp. 41-68, 2009.

Authors Profile

M. Sanchez received the**B.S.** degree in System Engineering in 2002 from Universidad de los Andes, Venezuela. The**M.S.c.**degree in Computer Science from Universidad de los Andes, Venezuela, in 2012.Currently doing **Dr.**degree in applied science (Ambient Intelligence) in Universidad de los Andes, Venezuela. His research interest includes Artificial Intelligence, Learning Environments, Graphic computing, Programming languages.

J. Aguilar received the **B.S.** degree in System Engineering in 1987 (Universidad de los Andes-Venezuela), the **M.Sc.** degree in Computer Sciences in 1991 (Universite Paul Sabatier-France), and the **Ph. D** degree in Computer Sciences in 1995 (Universite Rene Descartes-France). He was a Postdoctoral Research Fellow in the Department of Computer Sciences at the University of Houston (1999-2000). He is a Titular Professor in the Department of Computer Science at the Universidad de los Andes, Mérida, Venezuela, and Prometeo Research at the Universidad

Técnica Particular de Loja. He is member of the Mérida Science Academy and of the IEEE CIS Technical Committee on Neural Networks. He has published more than 400 papers and 10 books, in the field of parallel and distributed systems, computational intelligence, science and technology management, etc. His research interest includes Artificial Intelligence, Semantic Mining, Big Data, Emergent System, and Smart Environment.

J. Cordero received the **Eng.** degree in Engineering in Computer Systems and Computation from the Universidad TécnicaParticula de Loja, Ecuador, in 2006. Received the **MSc.** in Science and Technologies of the Computer in Universidad Politécnica de Madrid, Spain, 2014. He is working as a university teacher and researcher since 2007 of Computer Science Departament at the Universidad Técnica Particular de Loja. Her research interest includes artificial intelligence, ambient intelligent (AmI), social robotics.

P. Valdiviezo received her Master **degree** inDistance Education from Universidad Técnica Particular de Loja, Ecuador. Diplomaof Advanced Studies (**DEA**) inArtificialIntelligence(UNED, Spain). Engineering in ComputerScience (Universidad Técnica Particular de Loja, Ecuador). Professorof Computer ScienceDepartmentat the Universidad Técnica Particular de Loja. Research interests: Artificial Intelligence appliedto education,Technologies appliedto Education,SemanticWeb. E-mail: pmvaldiviezo@utpl.edu.ec