Collaborative Agents in Adaptative VLEs: Towards an Interface Agent for Interactivity and Decision-making Improvement

Karize Viecelli[1], Aluizio Haendchen Filho[1], Hércules Antonio do Prado[2], Edilson Ferneda[2], Jeferson Miguel Thalheimer[1] and Anita Maria da Rocha Fernandes[1]
[1]Laboratory of Applied Intelligence, University of the Itajaí Valley (UNIVALI), Rua Uruguay, 458, Itajaí, Brazil
[2]Catholic University of Brasilia (UCB), Brasilia, Brazil

Keywords: Interface Agent, Adaptative e-Learning, Virtual Learning Environments, Microservices.

Abstract: This paper presents an Interface Agent (IAg) in the context of collaborative software agents aiming at improving the interaction, interactivity and decision-making processes in Virtual Learning Environments (VLE). Working collaboratively in a multi-agent system, IAg receives notifications about situations that require interaction with students to assist and motivate them in the processes of navigation and use of VLE. In order to assist decision-making processes, it provides dashboards that enable the human tutor and VLE coordinators to make real-time decisions about non-normal situations. In addition, it monitors the actions of students seeking for clarifying doubts, utilizing a knowledge based on past situations. With this approach it is expected to enable a more attractive environment to students by reducing feelings of demotivation and isolation, and helping to reduce student dropout.

1 INTRODUCTION

The Distance Learning (DL) market is the fastest growing modality in Brazil, and already represents ¼ of enrolments, according to the Brazilian Association of University Education Maintainers (ABMES, 2019). This growth tends to continue. ABMES predicts that distance education will surpass presential university education by 2023. However, there are major challenges.

In the 10th Distance Learning Census, the Brazilian Association of Distance Learning (ABED, 2018), pointed out a high dropout rate and a low graduation completion rate. According to ABMES (2019), the indicators of the completion rate of distance learning students in 2016 was 35% and the dropout rate reached 62%, tending to increase. For Open University (UK), this is a globalized scenario, as internationally graduation rates would be close to 10% and tending to decline (Woodley, Simpson, 2014).

It was observed that the main reasons for high dropout rates are the feeling of demotivation and isolation of students (ABED, 2018). From this perspective, the suggested actions to reduce these rates are similar for both Open University researchers and ABED and ABMES. They are unanimous in recommending the development of proactive motivational support from institutions for student retention.

In contrast to this idea, it is clear that the vast majority of institutions are reactive, that is, they wait for students to contact them for help (Woodley, Simpson, 2014).

In this sense, research seeks to approach the issue of isolation and motivate the student by means of interaction and interactivity actions in the context environment / tutor / student. For achieving quality in student teaching and learning, it is clearly necessary to monitor the Virtual Learning Environment (VLE). However, one cannot ignore that, by supporting large numbers of students and proactive supportive actions, mentoring quality can be compromised.

In this context, Simbine et al. (2018) implement a model of visualization of student interactions based on their learning trajectory. By monitoring the way the student interacts in the VLE, the model generates graphical information with their interaction characteristics.

Viecelli, K., Filho, A., Antonio do Prado, H., Ferneda, E., Thalheimer, J. and Fernandes, A.
Collaborative Agents in Adaptative VLEs: Towards an Interface Agent for Interactivity and Decision-making Improvement.
DOI: 10.5220/0009415006910702
In *Proceedings of the 22nd International Conference on Enterprise Information Systems (ICEIS 2020) - Volume 1*, pages 691-702
ISBN: 978-989-758-423-7

The environment adaptation to the student profile is another important issue. Vaidya and Sajja (2016) proposed an adaptive learning agent platform that produces and assesses learning contents. It applies analytical reasoning about content before and after presenting it to the student in VLE. Student interaction monitoring data is used for planning and organizing the content of the learning environment.

This paper presents an intelligent software agent for making a VLE more interactive, intelligent, enjoyable, and student-adaptive environments. This agent, herein called Interface Agent (IAg), is responsible for interaction and interactivity in the environment. IAg operates in a collaborative organization of agents, playing its role proactively and autonomously to take action and reduce the work of human actors. The goal is to develop proactive motivational support resources to reduce the sense of isolation and abandonment while making the student a protagonist of their learning.

The paper is structured as follows: Section 2 presents the background, describing the agent technologies, data structure, and the Artificial Intelligence technique adopted for modelling IAg, Section 3 discusses some related works found in the literature, Section 4 explains the methodological approach, Section 5 discusses some preliminary results, and Section 6 concludes with some remarks and future work.

2 BACKGROUND

2.1 Microservice-oriented Multi-agent System

Despite the drawbacks reported about Service-Oriented Architecture (SOA), it remains the best option available for system integration and leverage of legacy systems (Alencar et al., 2013) due to its inherent ability to compose applications, processes, and assemble new functionalities from existing services. Inside the industry segment, the SOA principles have evolved in the form of microservices, an architectural paradigm that is based on fine-grained and independent software components that interact to build highly scalable distributed systems (Dragoni et al., 2017).

Although not so spread in the industry realm (Collier et al., 2015), several principles of MAS, as decentralization, distributed environments, amongst others, have been observed in the microservices model (Burkhardt, 2018). Several authors have studied the applications of microservices paradigm as a framework for building modern MAS, in an attempt to shorten this gap between industry and academic efforts (Burkhardt, 2018; Collier et al., 2015; Higashino et al., 2018). To assure that microservices can meet these expectations, multiple specifications and standards have been proposed and created, and middleware products are becoming more robust (Alencar et al., 2013). MIDAS (Haendchen Filho, 2017) is a platform that relies on microservices as the basis for the development of distributed MAS. Its architecture is composed by a front-end server (MIDAS Server) and one or more agent containers (MIDAS Container), as shown in Figure 1.

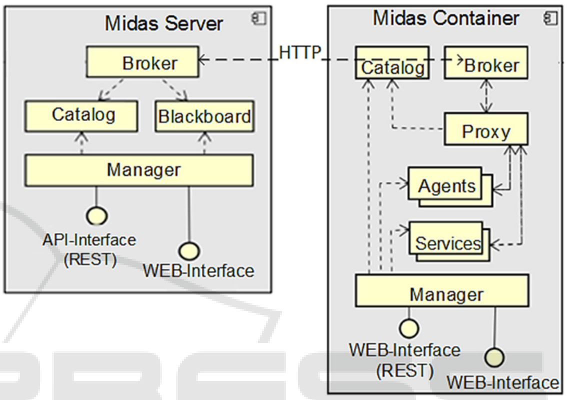

Figure 1: MIDAS Generic Architecture.

MIDAS Server is responsible for the platform integration rules, synchronizing the containers and interoperating with external applications. It contains three main interfaces: *(i)* an HTTP interface for intra-platform communication between MIDAS Server and the MIDAS Containers; *(ii)* a REST interface that allows communication with external applications; and *(iii)* Web interface for human management and configuration.

Each MIDAS Container is a lightweight container that houses software agents and/or microservices. It is capable of cataloguing the interface of its own services, and to break the conversation between its own agents and foreign agents. The containers may register themselves on a MIDAS Server, exposing their services and agents, allowing for distributed collaboration with other containers within the same server domain.

Also, the MIDAS Server performs the integration and discovery on its child containers, eliminating the complexity of service lookup and remote requests between containers. The application agents are instantiated in containers and developed by extending the abstract class, from which specific application behaviour can be implemented.

Placed in the MIDAS Server and also in the MIDAS Containers are middleware agents: Broker, Proxy, Catalog, Blackboard and Manager. They provide infrastructure services, playing in a collaborative and pro-active way the roles defined by the reference architecture. The introduction of the agent concept to play these roles complies with the current tendency and non-functional requirements for microservices-oriented architectures: flexibility, dynamic behaviour, pro-activity, and adaptability. They completely abstract the standard code required to implement those characteristics, such as communication protocols, concurrency control, lifecycle management, and services discovery and interoperability, enabling the developer to focus only in specific characteristics of the application business.

The Broker agent focuses on the architectural aspects related to the message transport: send/receive, pack/unpack, and managing exceptions. It translates agents and services request in HTTP streams.

The Catalog agent is responsible for the relevant aspects related to the resources concept of a resource-oriented model. A resource description is a machine runnable metadata representation that makes possible for a human or software program to locate services and agents within the ecosystem.

The Proxy agent plays the role defined by the service-oriented model, which focuses on the architectural aspects related to the messages processing. It acts as a service provider representative, being responsible for the dynamic configuration and creation of instances. Dynamic configuration focuses on the capacity of redirecting messages to different providers during runtime, whenever the Catalog agent updates the resource model.

The Manager is the most complex agent in the architecture, playing the roles defined at the management and policy levels. It involves a set of tasks that enable the control over the platform, such as the life cycle management, checking activities, statistics, QoS (Quality of Services) reporting, and GUI (Graphical User Interface) wizards.

Finally, the Blackboard takes responsibility for information exchange in symbolic cognitive MAS. Its structure follows the basic blackboard pattern: the knowledge sources represent the agents, the data structure is visible to all agents, and the controller is responsible for notifying the agents about the changes in the environment. When a MIDAS Container is running in stand-alone mode, it has a local Blackboard agent that deals with intra-container communication, and when a Container connects to a MIDAS Server, each local Blackboard of each MIDAS Container synchronizes with the MIDAS Server Blackboard in order to provide transparent communication within the whole ecosystem.

2.2 Data WebHouse

The Web allows recording practically all behavioural actions of the user in a single click (Kimball and Merz, 2000). It means that one can capture not only the page accessed but also navigability information. The recording of all interactions made by anyone via an application or web site, is called a clickstream. Activities carried out by the user such as click capturing, form filling, and others, create conditions for analysis, profile identifications, preferences and trends of each particular user.

Figure 2 shows a very simple example of a dimensional model for a Data Webhouse (DWH) for a VLE. As for Data Warehouse, a DWH is based on an architecture called Dimensional Model. Dimensional modelling is a discipline that seeks to model data for the purposes of understandability and performance.

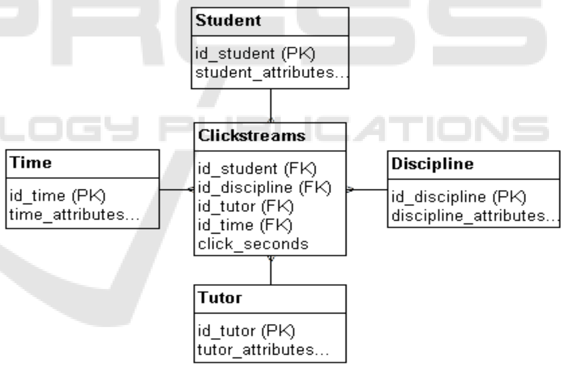

Figure 2: Simple dimensional DWH example for a VLE.

All dimensional models rely on the concept of measured facts. The Facts table, represented by the entity Clickstreams, stores users clicks on the VLE. The dimensions relate to the entities that serve as perspectives of analysis in any subject of the model. In the example, Student, Discipline, Time and Tutor are the dimensions connected to the fact table. Dimensions are rich in descriptions. For example, the Student dimension stores all the student profile data.

Besides manipulating information and discovering knowledge, a VLE needs to be prepared to react immediately to students' actions in the environment, eliminating the time between the occurrence of an event and the execution of an action (Sassi, 2010). This is called Zero Latency Enterprise (ZLE). The idea in a ZLE strategy is to use DWH integrated with other Business Intelligence tools to deliver real-time, zero-latency information for much faster decision making.

2.3 Case-Based Reasoning

Case-Based Reasoning (CBR) is an Artificial Intelligence technique for problem solving and knowledge acquisition based on the principle that “similar problems have similar solutions” (Aamodt and Plaza, 1994). According to Vitorino (2009), the use of CBR methodology and its application in VLE is based on a broad cognitive theory that involves the process of remembering, as a problem-solving phenomenon, and the process of reusing past episodes to solve new problems, that corresponds to a frequent and powerful way of human reasoning.

In CBR approach, knowledge maintenance is simplified by the ability to learn new information in the form of cases. Other advantage is the fast response time and the ability to work in domains that are not completely known. These features enable its application in many types of tasks such as diagnostic systems, help desk systems, evaluation systems, decision support systems, and project systems (Kolodner and Leake, 1996).

The basic elements of a CBR system are: *(i)* knowledge representation, carried out by means of concrete experiences; *(ii)* similarity measure, which looks for similar situations for the current problem in a knowledge base; *(iii)* adaptation, where past situations not identical to the current problem can be adapted to find a suitable solution for the new one; and *(iv)* learning, which occurs every time a case is resolved and a new experience is retained and integrated into the knowledge base.

A conceptual model for the cycle CBR (Figure 3) was proposed by Aamodt and Plaza (1994). It encompasses a continuous cycle of reasoning, consisting of four main tasks: *(i)* recovering the most similar case(s) from the case base, in which the goal is to find a case or a small set of cases in the base that contains a problem description near to the current problem or situation; *(ii)* reusing this case(s) to solve the problem; *(iii)* review the proposed solution in order to transfer it to the present situation; if necessary, the recovered solution can be adapted to fully meet the requirements of the present situation; and *(iv)* retaining the experience represented by the current case (or parts of that experience) for future reuse.

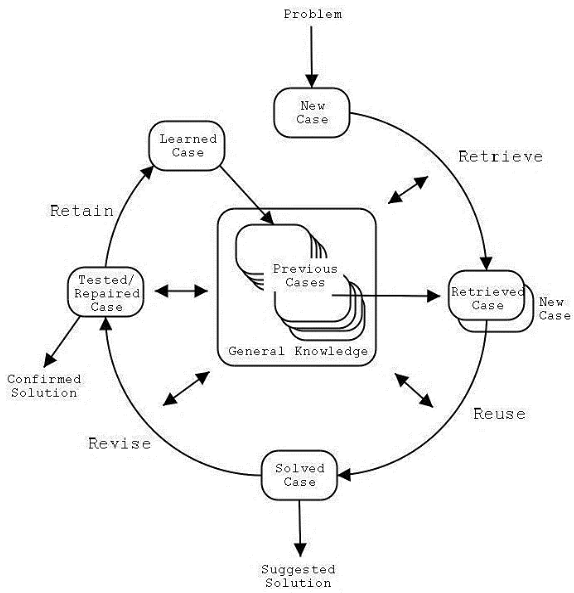

Figure 3: Cycle of Case-Based Reasoning (Aamodt and Plaza,1994).

2.4 Felder-Silverman Model

Felder and Silverman (1988) developed a theory that states there is difference in the way students learn: seeing or hearing; reflecting and acting; reasoning logically and intuitively; memorizing, visualizing, drawing analogies and building mathematical models; steadily or not.

The authors mapped learning styles and created a questionnaire entitled Index of Learning Styles (ILS) base on them. The ILS is an instrument that evaluates seven identified dimensions (Table 1). Based on this assessment, according to the answers provided, an index is calculated that establishes the predominant dimension present in each profile. The index ranges from 1 to 11 and represent the intensity of the categories.

Table 1 presents the proposed dimensions: perception (sensory or intuitive), input (visual or verbal), organization (inductive or deductive), processing (active or reflective) and understanding (sequential or global).

This model has been widely used to classify profiles (Freitas et al., 2006; Aguiar et al., 2014; Trevelin et al., 2013). Some of the features identified by ILS are:

- **Active Students.** Tend to understand and retain information if they can turn that knowledge into action.
- **Reflective Students**. Prefer to think about information before acting and tend to enjoy working alone.

Table 1: Learning dimensions (Felder and Silverman, 1988).

Learning Style preference		Corresponding Teaching Style	
Sensory Intuitive	Perception	Concrete Abstract	Content
Visual Verbal	Fundraising	Visual Verbal	Presentation
Active Reflective	Processing	Active Passive	Student Participation
Sequential Global	Understanding	Sequential Global	Perspective

- **Intuitive Students.** They like innovations, but not repetitions; they may be better at understanding new concepts and tend to be more innovative and work faster than sensory students.
- **Sensory Students.** They like to learn facts and solve problems by established hands-on methods and don't like surprises and complications.
- **Visual Students.** They easily remember what they see such as movies, photos, diagrams and demonstrations.
- **Verbal Students.** Acquire and assimilate knowledge based on written and spoken explanations.
- **Sequential Students.** Tend to gain knowledge in linear, logically interconnected steps, and follow step-by-step ways to find solutions.

3 RELATED WORKS

Simbine et al. (2018) proposed a model for student interaction visualization on the basis of their learning trajectory, along with an interactive visualization system of learning trajectories in VLE. Thus, it generated a model of data collection, visualization and analysis in the form of graphs, according to the characteristics of student interaction. By analysing these data, it was possible to verify the order of access of students' interaction with existing content, which can be used to improve the organization of educational content in the VLE.

Referring to the issue of interaction with students in VLE, Maciel et al. (2014) propose a virtual assistant integrated with the Moodle environment in order to offer daily support to the academic activities of distance learning students. This wizard exposes the content orally through a visual avatar, making it more interesting for students. Besides, it allows the human tutor to contact the student through this avatar, sending messages in BackOffice.

Regarding the adaptability of a VLE, Vaidya and Sajja (2017) proposed an agent-based system for collaborative learning environment in an educational habitat. The approach provides an agent that not only offers the student learning facilities, but also calibrates content and learning outcomes. Dorça (2012) presents a probabilistic approach using reinforcement learning, in which a dynamic, interactive and gradually updated student model is implemented through a stochastic process. Model updating occurs based on information about student performance within the learning environment. This approach adopts the ILS.

Zapparolli et al. (2017) develop a tool called FAG that uses Business Intelligence and Learning Analytics techniques to assist in knowledge management and decision support in a VLE. The tool provides analytical and consolidated reports with cross-sectional and systemic views, considering all virtual classrooms and contexts of a specific teacher. It enables corrective actions to be taken, ensuring quality work and preventing dropout risk.

In the context of knowledge management, Heinzen (2002) presented a tool to assist the teaching of programming logic. Departing from the problem statement, the system retrieves solutions to similar problems previously solved. The work emphasizes problem solving based on analogy, being focused on code. Nascimento et al. (2016) apply CBR to suggest a pedagogical action for a student-learning problem. This system also uses learning objects to support pedagogical actions aimed at facilitating the understanding of complex concepts of the Introduction to Programming discipline.

Although the many efforts to improve interaction, interactivity, and decision-making processes in VLE (CBR, ILS questionnaire, and BI techniques, and so on), none of them provide an integrated platform with the most relevant functionalities, unlikely the proposal presented in next section. It is also important to note that neither approach adopts a proactive procedure, all are just responsive.

4 IAG DESCRIPTION

This section introduces the main methodological procedures for the proposed approach.

4.1 Defining the Generic Architecture

The generic architecture of IAg is shown in Figure 4 including the following components: *(i)* a microservice-oriented platform for MAS development and management (Haendchen Filho et al., 2019), as presented in Section 2.1; *(ii)* a DBW structure for data representation; *(iii)* an organization of collaborative agents, instantiated on the platform; and *(iv)* the Moodle virtual environment, used for the case study. As described in Section 2, the platform facilitates the development of agents, providing communication, management, and database access.

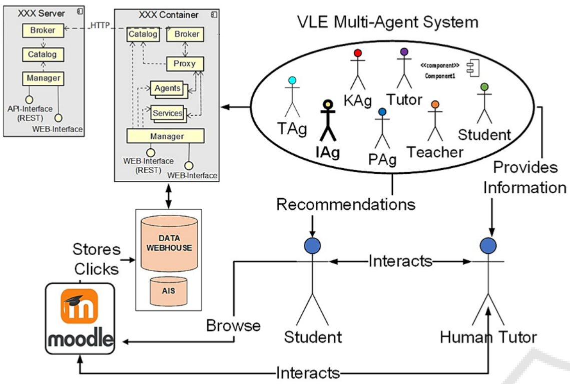

Figure 4: Generic Architecture.

The VLE MAS instantiated in the platform is composed of the following collaborative agents: *(i)* Tracing Agent (TAg), responsible for storing and managing the data structure; *(ii)* Interface Agent (IAg), that perform interaction with human actors; *(iii)* Knowledge Agent (KAg), which manages AI techniques to perform predictions and prescriptions; *(iv)* Pedagogical Agent (PAg), which performs content management, learning objects and trails; and *(v)* Student, Tutor and Professor, which represent virtual instances of these human actors.

The databases are represented by the Academic Information System (AIS) and the DWH. The AIS contains academic data, such as student profile and history, data from tutors, teachers, discipline/courses, and so on. The DWH is a dimensional model composed by a central table of facts, connected with the dimensions.

Listeners are placed in relevant spots in the interface, waiting for the clicks triggering the script to store the information. Locations do not necessarily have to be on the links since data can be stored with simple interactions. Clickstreams should be sent to the webhouse data for storage.

All actions the user take can disclose knowledge about the use of the system. DWH is widely used to process analyses, obtaining information from two main sources: *(i)* communication protocol data, stored in the web services logs; and *(ii)* behaviour seized with site scripts after establishing a session. User behaviour on pages is a critical part because it is not so simple to change a site to capture the information.

4.2 Data Gathering

In order to promote collaboration, the TAg interacts with the Interface Agent for populating the DWH with the necessary data on student interaction in VLE.

Figure 5 shows the data structure stored in the dimensional model from which IAg can obtain the data with the collaboration of TAg. A central Fact table is connected with the dimensions relevant to the context of the VLE.IAg have access to data to turn it into real-time information that may be used to interact with students, tutors, and teachers. Besides, it will use BI tools to generate OLAP dashboards, assisting teachers, and tutors in the decision-making process.

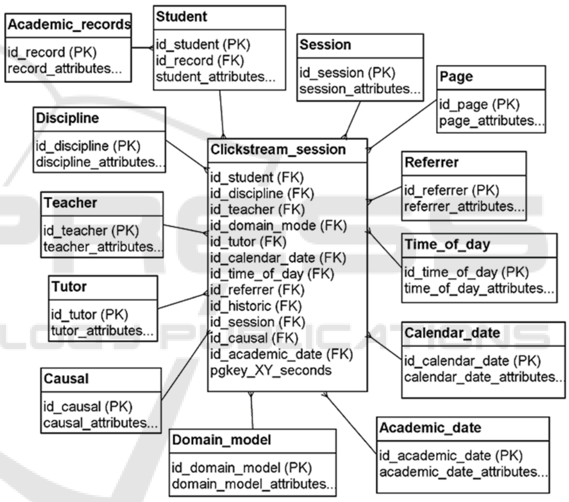

Figure 5: VLE Webhouse Model.

The summary descriptions of dimensions are:

- **Calendar Date.** Attributes may include days of the week, seasons, and holidays, among others.
- **Time_of_day.** Time slots during the day, including hours, minutes, and time slots like lunchtime, class time, and so forth.
- **Academic Date.** Associated with different structures that differ on the number of modules (semesters, four-month periods and trimesters).
- **Page.** The page source (e.g., static, dynamic), function (content, exercise, video, forum) and so forth.
- **Session.** Session is the collection of actions taken by a visitor to a site while it navigates without leaving this site.

- **Causal.** Describes the conditions of the current progress of the subject, such as beginning of the subject, period of tests, etc.
- **Student.** Information about the student profile.
- **Academic Records.** Provides information on the student trajectory.
- **Discipline.** The attributes can include class hours, credits, opening and closing dates, and so forth.
- **Teacher.** Information about teacher profile.
- **Tutor.** Information about the tutor including the degree of training, number of tutored disciplines, etc.
- **Domain Model.** Describes the course schedule of the learning path in the VLE.
- **Referrer.** Brings information about the URL from where the user came from.

4.3 Functionalities Specification

The role model has been used (Gonçalves, 2009, Haendchen Filho, 2017) to provide a summary of software agents. A role can be described by two basic attributes: *(i)* responsibilities are obligations and indicate functionality, and *(ii)* permissions are the rights associated with the role and indicate the resources that the agents can use. Interaction and interactivity are key concepts in the IAg role model.

Interactivity can be defined as the communication human-machine and refers to a mediated environment in where participants can communicate synchronous or asynchronously and participate in reciprocal message exchanges (Kiousis, 2002). Interaction occurs among same nature entities: human-human or machine-machine. Table 2 presents the IAg role model.

The IAg responsibilities are:

- **Login Procedures.** IAg has two responsibilities in the login procedure. The first is to apply the ILS questionnaire when the student's first login occurs. In this procedure, he works in collaboration with the PAg. According to the information collected in the questionnaire, the student is inserted in one of the profiles provided for in the Felder-Silverman model. The second responsibility is to create an interactive message each time the student logs in, containing a set of information about their last access.
- **Knowledge Representation.** In representing knowledge, IAg's responsibilities are to record new questions and allow both the student, the tutor and/or the teacher to consult on cases that have already been resolved. In this case, IAg's main responsibilities are to collect cases and questions and consult via KAg the most similar cases that can be reused for a specific problem.
- **Interactivity with the Human Tutor.** The IAg is also responsible for enabling the tutor to interact with students by sending notifications with relevant information. These include: *(i)* notify busiest time in class, allowing the tutor and teacher to interact in real time with students through chats, forums and any other activity that may motivate interaction with groups; *(ii)* generate notification of absence list, low/high frequency in the discipline; and *(iii)* inform about high or underperformers.
- **Interactivity with the Student.** IAg's interaction with students occurs by sending interactive message at login, invitations to chat and forums with other students and tutors. It also *maintains* a proactive stance, informing the student of their latest actions in the environment. In addition, it sends welcome messages to new students, messages for low-frequency students offering help.
- **Interactivity among Tutors, Teachers and VLE Managers.** The data structure stored in a dimensional model enables IAg to use OLAP tools to provide important information to VLE tutors, teachers, and managers. for decision-making. The tool provides analytical and consolidated dashboards with cross-sectional and

Table 2: Partial IAg Role Model.

ORGANIZATION: VLE MULTIAGENT SYSTEM
AGENT: INTERFACE AGENT
RESPONSIBILITIES
Login Procedures
▪ Applies ILS Questioner
▪ Create interactive message
Knowledge Representation
▪ Registration of Doubts
▪ Look up cases
Interactivity with the Human Tutor
▪ Notify busiest time in class
▪ Report absence, low / high attendance
▪ Inform about high or underperformers
Interactivity with the Student
▪ Send interactive message at login
▪ Send track progress notification
▪ Inform student of last login actions
Interactivity among tutors, teachers and VLE managers
▪ Statistical Reports on the discipline
▪ Generate OLAP Dashboard
Interaction
▪ Send student profile identification to PAg
▪ Receive TAg notification from missing student who signed in

systemic views, considering all dimensions and contexts of a specific classes and students.

- **Interaction.** In this group, microservices are placed. They involve interaction among IAg and other collaborative agents. The microservice *Send student profile identification to PAg* means that IAg collects information from the ILS questionnaire, scores the answers, and identifies the student profile in one of 7 possible Felder-Silverman Model categories. With this information, PAg can define which learning paths best fit this profile. The microservice *Receive TAg notification from missing student who signed in* enables preventive measures to be taken when a student who has spent time without logging in to VLE logs in. In this case, it is important to find out why he is absent and if he needs help with any difficulties. Agent interactions can occur synchronously or asynchronously. In asynchronous mode, Blackboard is used as a mediator, as will be shown in the following section.

4.4 IAg's Responsibilities Specification

In order to specify IAg's responsibilities, the HEFLO (https://app.heflo.com/) tool was used. It has strong adherence to the Business Process Model and Notation (BPMN) for process diagram (orchestration). BPMN diagramming is intuitive and allows the representation of complex process details as a standard language.

Figure 6 presents a workflow of collaboration and interaction among agents and other actors through task and service modelling. Links and messages describe how they are related and how they interact.

According to the service workflow, the procedure starts when the student logs in to the Moodle environment. After logon, the system begins to collect log data, which is stored in a database (DB). At this time TAg, which is responsible for coordinating and maintaining the DWB, performs the task of identifying the student profile.

If it identifies linked profile, TAg runs a new service for retrieving information from its last login and sends this data to IAg. The IAg upon receiving this information initiates an interactivity service by creating a welcome message with information from the student's last login and where they left off.

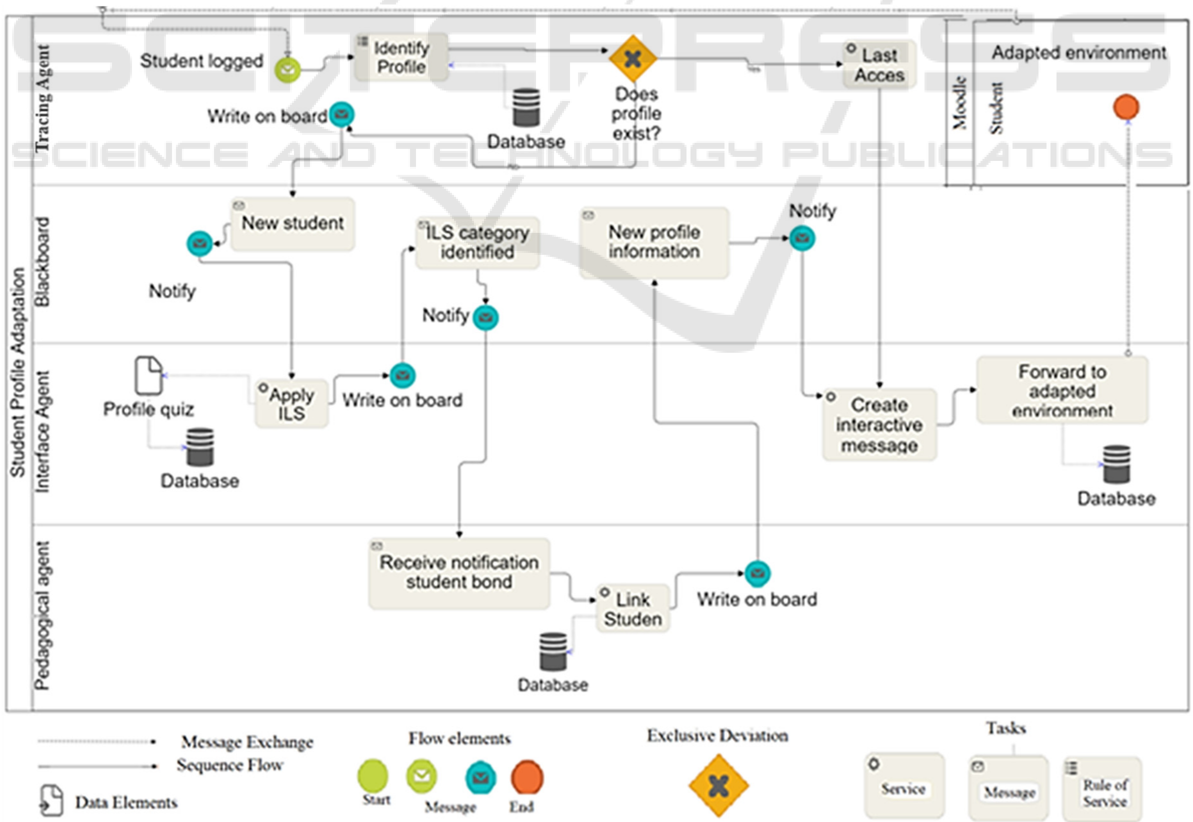

Figure 6: Workflow adaptation - login procedure.

However, if the student does not have a profile linked to his/her account, TAg writes on Blackboard (BB) that it is necessary to apply the questionnaire for this student. Upon receiving the message, BB generates a notification for IAg to apply the questionnaire. So, the IAg begins the service of applying ILS questionnaire to rate the student profile. When it finishes, IAg posts a message on the BB, notifying TAg for storing profile data. After reading the message on BB and storing the profile data, TAg writes on the BB a message for the PAg informing that there is a new student with a defined profile.

Thus, PAg will be able to offer this new student the layout and the most appropriate track for his/her learning. The IAg now performs an interactive welcoming message for the student and forwards his/her to its adapted environment. environment.

5 IMPLEMENTATION

As mentioned in previous sections, the VLE system was instantiated and implemented on the MIDAS platform. On the platform, the S-Manager infrastructure component provides a GUI wizard to assist in global management tasks, as shown in Figure 7.

In the panel on the right side of the figure, all agents that make up the VLE can be seen. The panel shows a navigable resource hierarchy organized by agents. When an agent is selected, details are displayed in the Details panel on the left side of the window. At the bottom of the window, the Server Log panel shows details of all transactions being executed. In the Containers panel on the top left, you can see the two containers registered on the platform and used in this domain: the instantiated VLE and the Academic Information System, which works integrated with the VLE, from which the data for the DW is extracted.

The details of the Interface Agent in the right-hand panel show the services it provides. The + sign in front of the services indicates that these services are broken down into microservices at a lower level of detail, as previously shown (Figure 6).

For the implementation of graphical BI interfaces, the API of the Power BI tool (Microsoft, 2019) was used. Once the data is loaded in the tool, it is possible to transform them by means of the Query Editor option. This function offers several data preparation functions such as dividing and grouping columns, creating calculated columns, applying filters and even building relationships

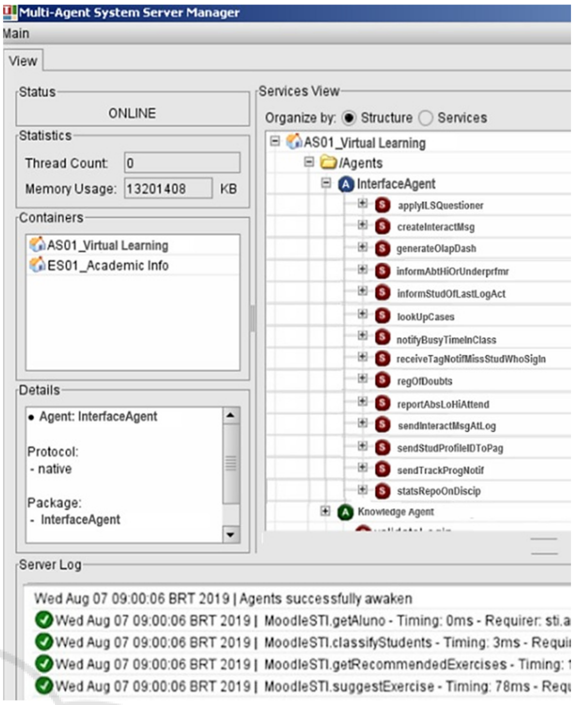

Figure 7: GUI to view instantiated agents and its services.

between tables.

The home screen allows the creation of visualizations in an intuitive way with drag-drop functions. The visualization of the dashboard or ready-made report with drill-up and drill-down operations, as well as the application of filters for data analysis and knowledge acquisition was carried out successfully. Many ways for visualizing are available in the side menu and the columns can be selected, as shown in Figure 8.

The Power BI JavaScript API provides bidirectional communication between Power BI reports and the application. The JavaScript API enables to more easily embed reports into applications and to programmatically interact with those reports so that the applications and the reports are more integrated.

The software also has the attractive feature that the reports created in the tool can be accessed from mobile devices through Power BI Mobil, being made available free of charge for operating systems, Android, IOS and Windows Mobile.

Knowledge representation was implemented with the development of a question base, previously fed, that allows the student to carry out research. The prototype was developed using CBR techniques and rules, applying similarity based recover procedure for answers retrieval. Figure 9 shows the results of a

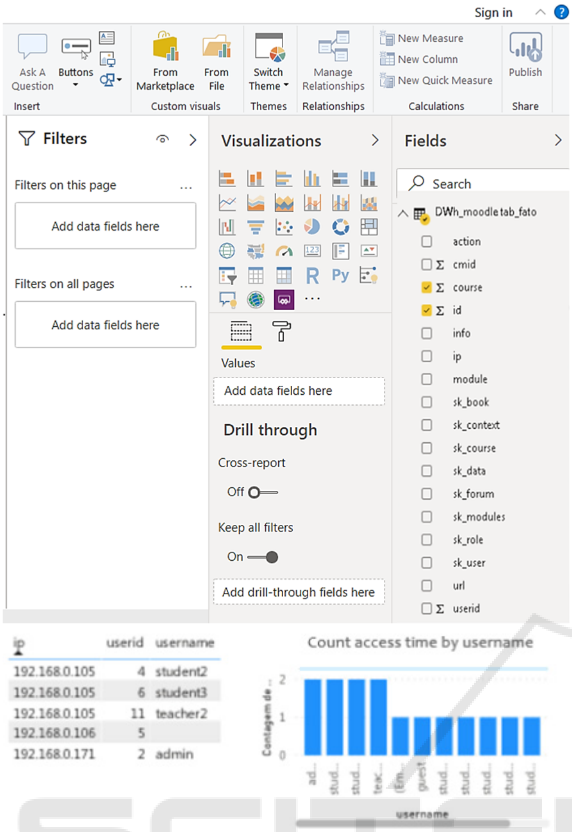

Figure 8: Power BI filters and the partial view of a query.

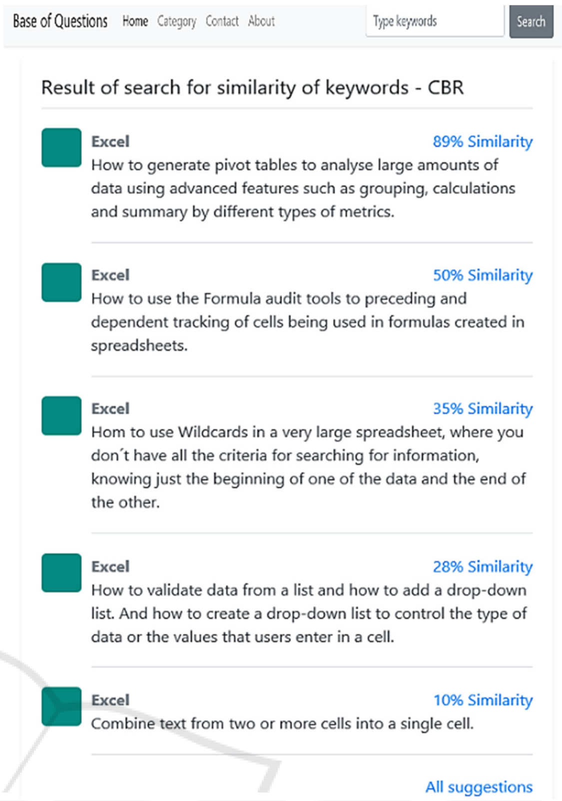

Figure 9: CBR Similarity Search Result.

search, presenting several questions organized by degree of similarity to the searched subject.

For developing this part of the solution, the NetBeans IDE 8.1 development environment was used. Bootstrap was chosen as a framework for developing the web interface, and MySQLi was used as the database management system.

The results achieved so far have been shown to be adequate to what was proposed in this work. In addition to its proactive nature, the solution offers yet another tool for student learning, providing a knowledge base that will assist the course with several classes of students. The recovery, adaptation and learning of registered cases are still under development.

6 DISCUSSION

The work presented in this paper is primarily based on suggestions from Open University (UK), Brazilian Association of Distance Learning Maintainers (ABMES, 2019), Brazilian Association of Distance Learning (ABED, 2018), and the previously cited authors, beyond other researchers (Choi et al., 2018; García-Álvarez et al., 2018) that unanimously recommend the development of proactive solutions for VLEs. The literature shows that the vast majority of existing solutions do not meet this requirement, or only partially.

As previously mentioned, the following works focuses in the same problem: *(i)* Zapparolli et al. (2017), which provide analytical and consolidated BI reports; *(ii)* Dorça (2012), that uses a dynamic, interactive, and gradually updated student model based on ILS profiles; *(iii)* Maciel et al. (2014), that use an avatar for facilitating interaction; *(iv)* Vaidya and Sajja (2017), that provides an agent that not only offers student learning facilities, but also calibrates content; *(v)* Simbine et al. (2018), that focus on the analysis and adaptation of learning trails; *(vi)* Nascimento et al. (2016), that apply CBR to suggest a pedagogical action for a student-learning problem.

The solution presented in this paper uses all these techniques in one approach, offering a complete and comprehensive proposal. In addition, it offers an environment composed of a set of collaborative agents with the proposal to create a virtual environment for proactive learning, in contrast to all approaches that act in a reactive way. As it is still a work in progress, it was not possible to show in this article all the potentialities that are in the development phase, mainly the proactive procedures of the solution. But the preliminary results achieved were beyond our initial expectations.

7 CONCLUSIONS AND FUTURE WORKS

Preliminary studies show that current VLEs are mostly reactive. This characteristic is a source of demotivation and feeling of abandonment for students, leading to high dropout rates and low graduation rates. More adaptable environments to student profile, with large-scale interactivity, and proactivity, can promote the expected benefits of VLEs. It is well-known that when technology expectations are unrealistically high and subsequently not met in practice, the result can be dissonance and dissatisfaction among stakeholders, especially students (García-Álvarez et al., 2018).

The main contribution of this work is a solution based on the needs of the educational market, aiming to guarantee the student expectation meeting and generation of adequate levels of motivation and satisfaction. For this, proactive characteristics such as adaptation, interactivity, and interaction were included to obtain a strong sense of satisfaction, possibly reducing dropout rates and increasing graduation rates.

For implementing the solution, a platform developed in the Applied Intelligence Laboratory at University of Vale do Itajaí was used. The platform has been already applied to successfully implement other collaborative agents of the system (Haendchen Filho et al., 2019).

As future works it is ongoing the implementation of proactive procedures and adaptative interfaces. Proactive procedures of social skill and autonomous behavior are being developed in the agent's workflow. Autonomy refers to the agent property of running without interacting with humans, and social ability indicates that they are able to interact by sending and receiving messages and not by explicit task invocation. For implementing interactivity, two approaches are being applied: *(i)* the specification of pre-defined rules (eg, welcoming students who are absent for *x* days), and *(ii)* the use of a knowledge base acquired by means of a machine learning process. In certain circumstances, the agent must have autonomy to communicate with the human actors Student or Tutor. Interactivity based on machine learning must consider that the knowledge acquired by the KAg can be used collaboratively by the IAg to assist the student in a proactive way.

Adaptative interfaces aim at providing an interface design with learning objects appropriate to the student's profile defined by the ILS questionnaire. A user model will be created in order to represent the way the developer will build the system based on computational thinking. That is, according to a logical sequence, observing the requirements, tasks, and user experiences and capabilities. The VLE graphical interface refers to the environment in which the user effectively interacts to accomplish a task. This user model will be managed by IAg, according to the student's profile, the learning trails, and the content objects, handled by the Pedagogical Agent.

REFERENCES

Aamodt, A. and Plaza, E. (1994). Case-based reasoning: Foundational issues, methodological variations, and system approaches. *AI communications*, 7(1), 39-59.

ABED - Associação Brasileira do Ensino a Distância (2018). *Brazilian e-Learning Census 2017*. http://www.abed.org.br/site/pt/midiateca/censo_ead/1554/2018/10/censoeadbr_-_2017/2018. *In Portuguese.*

ABMES - Associação Brasileira de Mantenedoras do Ensino Superior, 2019. *Brazilian Higher Education Census* http://www.abmes.org.br/documentos/detalhe/720/censo-da-educacao-superior-2018-principais-resultados. *In Portuguese.*

Aguiar, J. J. B., Fechine, J. M., and Costa, E. B. (2014). Cognitive Styles and Learning Styles in Informatics in Education: a systematic mapping focused on SBIE, WIE and RBIE. *Brazilian Symposium on Computers in Education*. *In Portuguese.*

Alencar, M. and Netto, J. M. (2013). Facilitating e-learning tutoring using SISACAD. *19º Congresso Internacional ABED de Educação a Distância*, Salvador/BA, Brazil.

Burkhardt, M. (2018). Towards a Pragmatic Approach to Engineer Cognitive, Agent-Oriented, Distributed Applications. *IEEE 3rd International Workshops on Foundations and Applications of Self Systems*.

Choi, S., Lam, S., Li, K., and Wong, B. (2018). Learning Analytics at Low Cost: At-risk Student Prediction with Clicker Data and Systematic Proactive Interventions. *Journal of Educational Technology & Society*, 21(2):273-290.

Collier, R. W., Russel, S., and Lillis. D. (2015). Reflecting an agent programming with AgentSpeak. *LNCS*, 9387.

Dorça, F. A., 2012. *A stochastic approach based on reinforcement learning for automatic and dynamic modeling of student learning styles in adaptive and intelligent systems for distance education*. PhD Thesis.

Universidade Federal de Uberlândia, Brazil. *In Portuguese.*

Dragoni, N., Giallorenzo, S., Lafuente, A. L., Mazzara, M., Montesi, F., Mustafin, R., and Safina, L. (2017). Microservices: Yesterday, Today, and Tomorrow. In: Mazzara, M., Meyer, B. (eds) *Present and Ulterior Software Engineering*. Springer.

Felder, R. M. and Silverman, L. K. (1988). Learning and teaching styles in engineering education. *Engineering Education*, 78 (7), 674-681.

Freitas, A. A., Dornellas, D. V., and Belhot, R. V. (2006). Production engineering student professional requirements: a view through learning styles. *Revista GEPROS*, 1. *In Portuguese.*

García-Álvarez M.T., Novo-Corti I., and Varela-Candamio, L. (2018). The effects of social networks on the assessment of virtual learning environments: A study for social sciences degrees. *Telematics and Informatics*, 35(4), 1005-1017.

Gonçalves E. J. T. (2009). *Modeling Software Agent Internal Architectures Using the MAS-ML Language*. MSc Thesis, Universidade Estadual do Ceará, Brazil. *In Portuguese.*

Haendchen Filho, A. (2017). *Multiagent Systems on the Internet: A service-oriented architecture for the development of intelligent software agents on the Internet*. Novas Edições Acadêmicas, 2017.

Haendchen Filho, A., Thalheimer, J. M., Dazzi, R. L. S., Santos, V., and Koehntopp, P. I. (2019). Improving Decision-making in Virtual Learning Environments using a Tracing Tutor Agent. *21nd ICEIS.*

Heinzen, L. A. (2002). *An Application of Case Based Reasoning to support programming logic teaching*, Regional University of Blumenau. *In Portuguese.*

Higashino, M., T., Kawato, T., and Kawamura. T. (2018). A Design with Mobile Agent Architecture for Refactoring a Monolithic Service into Microservices. *Journal of Computers*, 13(10).

Homayouni H., S. and Ghosh, I., Ray (2018). Data Warehouse Testing. *Advances in Computers*.

Kimball, R. and Merz, R. (2000). *The Data Webhouse toolkit*. John Wiley & Sons.

Kiousis, S. (2002). Interactivity: a concept explication. *New Media & Society*. 4.

Kolodner, J. L. and Leake, D. (1996). A tutorial introduction to case-based reasoning. In: Leake, D. (ed.). *Case-Based: Experiences, Lessons and Future Directions*. MIT Press, p.31-65.

Maciel, A. M. A., Rodrigues, R. L., and Carvalho, E. C. B. (2015). Development of an integrated Moodle virtual assistant to support online learning. *XXV Simpósio Brasileiro de Informática na Educação* (SBIE 2014). *In Portuguese.*

Microsoft, (2019). Introduction to Power BI Desktop. https://docs.microsoft.com/pt-br/power-bi/desktop-getting-started. *In Portuguese.*

Nascimento, P. B., Oliveira, E. H. T. de, Silva, F. S. da, Ramos, I. M. M. and Ramos, D. B. (2016). Case-Based Reasoning in Virtual Learning Environments: A Systematic Literature Review. In: Sánchez, J. (ed). *Nuevas Ideas en Informática Educativa*, 12, 551-556. *In Portuguese*.

Sassi, R. J. (2010). Data Webhouse and operational Business Intelligence: revisiting technology and analyzing Data Warehouse Trends. *XXX Encontro Nacional de Engenharia de Produção*. São Carlos, SP, Brasil. *In Portuguese.*

Simbine, F. B., Lima, J. V., Torre, M. R., and Chiguvo, S. S. (2018). Analysis of Learning Paths in Virtual Learning Environments through Information Visualization. *InfoDesign - Revista Brasileira de Design da Informação*, 15(2), 183-196. *In Portuguese.*

Trevelin, A. T. C., Pereira, M. A. A., and Oliveira Neto, J. D. (2013). The use of “inverted classroom” in technology courses: a comparison between the traditional model and the flipped classroom inverted model adapted to learning styles. *Revista de estilos de aprendizaje*, 6(12). *In Portuguese.*

Vaidya, N. M. and Sajja, P. S. (2016). Agent based system for collaborative learning environment in an educational habitat. *International Conference on ICT in Business, Industry, and Government*, p. 1–5.

Vitorino, T. A. S. (2009). *Case-Based Reasoning: Concepts and Applications*. MSc Thesis. UFMG, Brazil. *In Portuguese.*

Woodley, A. and Simpson, O. (2014). The elephant in the room. In: Anderson, T. and Zawacki-Richter, O. (Eds.). *Online distance education: Towards a research agenda*, 459-485.

Zapparolli, L., Stiubiener, I., Pimentel, E., and Braga, J. C. (2017). Applying Business Intelligence and Learning Analytics technics in Distance Learning Environments. *XXVIII Simpósio Brasileiro de Informática na Educação* (SBIE 2017).